Cross-site WebSocket hijacking

Executive Summary

This report documents the discovery and successful exploitation of a **Cross-site WebSocket Hijacking (CSWSH)** vulnerability in the chat feature of a target web application. The vulnerability allowed an attacker to establish a WebSocket connection on behalf of an authenticated victim and retrieve sensitive data such as usernames and passwords. The severity of this vulnerability is **critical**, as it enables full account takeover without requiring user interaction beyond visiting a malicious webpage.

Introduction

The objective of this security assessment was to evaluate the security of the live chat feature implemented using WebSockets. The focus was to determine whether the WebSocket implementation could be exploited from a cross-site context to gain unauthorized access to sensitive user data.

Methodology

1. Interacted with the live chat feature to analyze message flow.

2. Inspected WebSocket communication using Burp Suite's Proxy > WebSockets and HTTP History tabs.

3. Identified the use of the `READY` message to retrieve historical chat messages.

4. Observed that the WebSocket handshake lacked any form of CSRF token or origin verification.

5. Crafted a malicious HTML/JavaScript payload to initiate a WebSocket connection from an external origin.

6. Leveraged Burp Collaborator to receive exfiltrated chat data.

7. Extracted sensitive information (credentials) from the victim's chat history and used it to log into their account.

Vulnerability Findings

- **Type:** Cross-site WebSocket Hijacking (CSWSH)

- **Location**: WebSocket live chat endpoint

- **Severity:** Critical

Description

The WebSocket handshake used by the chat system does not validate the origin of incoming connections. This enabled an attacker to establish a connection from a malicious website visited by an authenticated victim. Once connected, the attacker could send the `READY` message, which causes the server to return the victim's entire chat history. This data is then exfiltrated to an attacker-controlled server.

Proof of Concept (PoC)

html

CopyEdit

Steps

1. Captured the WebSocket handshake URL from Burp Suite.

2. Crafted and hosted the exploit script on the lab's exploit server.

3. Verified functionality by sending the exploit to self and observing exfiltration via Burp Collaborator.

4. Delivered the same exploit to the victim.

5. Extracted the victim's chat history containing login credentials.

6. Used the credentials to log in as the victim and confirm access.

Impact Assessment

- Unauthorized access to sensitive information including login credentials.

- Complete compromise of user accounts.

- No user interaction required beyond visiting a malicious link.

- Could be chained with additional attacks such as XSS or privilege escalation.

Recommendations

1. Implement strict origin checks in the WebSocket server to reject unauthorized connections.
2. Use CSRF tokens or session validation for WebSocket connections.
3. Consider enforcing SameSite cookies and `Origin`/`Referer` header validation.
4. Educate developers on the risks of using WebSockets without origin validation.
5. Perform regular testing on real-time features using security tools and manual techniques.

Conclusion

This assessment revealed a critical CSWSH vulnerability in the application's chat feature. Proper origin validation and session control mechanisms are essential in securing WebSocket implementations. Addressing this issue is crucial to prevent unauthorized access and maintain the integrity and confidentiality of user data.