Clickjacking with form input data prefilled from a URL parameter(clickjacking)

Executive Summary

This report presents the successful exploitation of a **Clickjacking vulnerability** in the target web application. By leveraging a form that pre-fills user input from a URL parameter, the attacker was able to deceive the user into unknowingly clicking a hidden “Update email” button resulting in unauthorized modification of account data. The vulnerability poses a **moderate-to-high risk** to user privacy and control.

Introduction

This engagement aimed to identify UI redressing issues (clickjacking) within the user account management interface. The specific objective was to determine whether user data could be altered without consent via pre-populated form inputs combined with deceptive visual elements.

Methodology

1. Logged into the target application to examine the "Update email" feature.

2. Observed that the email input field is prefilled via a URL parameter.

3. Constructed a malicious HTML page using `iframe` and overlaid a decoy “Click me” button aligned with the hidden “Update email” button.

4. Applied CSS styling to reduce iframe opacity and ensure seamless alignment.

5. Verified successful exploitation by testing the flow in Chrome, which mirrors the victim's environment.

6. Delivered the exploit through the exploit server and confirmed that the target’s email was changed without their knowledge.

Vulnerability Findings

- **Type:** Clickjacking with prefilled input

- **Location:** `/my-account?email=...`

- **Severity:** Medium–High

Description

The vulnerability came from a combination of:

- Absence of anti-clickjacking headers (`X-Frame-Options`, `Content-Security-Policy`)

- Form input fields that accept values directly from URL parameters

- Lack of user verification before applying changes

These factors allowed the construction of a transparent iframe-based exploit, tricking the user into unknowingly changing their email address by clicking on an overlaid decoy element.

Proof of Concept (PoC)

html

CopyEdit

This payload was hosted on the exploit server and delivered to the victim.

Impact Assessment

- User’s email address was changed without consent.

- Could lead to account takeover if email-based authentication is used.

- May undermine user trust and violate privacy regulations.

- Attack requires only one click, making it easy to exploit in real-world scenarios via social engineering.

Recommendations

1. Implement anti-clickjacking headers such as:

   - `X-Frame-Options: DENY`

   - `Content-Security-Policy: frame-ancestors 'none';`

2. Avoid pre-filling sensitive form fields from URL parameters.

3. Require explicit user interaction or re-authentication for sensitive actions.

4. Use JavaScript frame-busting scripts for defense in depth.

5. Educate developers on secure UI practices and perform regular code reviews.

Conclusion

This vulnerability demonstrated how poor UI design and lack of frame protection can be exploited to manipulate user behaviour. Though the attack requires user interaction, the risk of unauthorized data modification is significant. Addressing this issue with proper headers and form-handling logic will greatly enhance the application's resilience.