掃描結果詳細資料

Potential Clickjacking on Legacy Browsers

查詢路徑:

JavaScript\Cx\JavaScript Low Visibility\Potential Clickjacking on Legacy Browsers 版本:7

類別

CWE top 25: CWE top 25
MOIS(KISA) Secure Coding 2021: MOIS(KISA) Verification and representation of input data
OWASP ASVS: V05 Validation, Sanitization and Encoding
OWASP Top 10 2021: A8-Software and Data Integrity Failures
SANS top 25: SANS top 25
ASD STIG 5.2: APSC-DV-002330 - CAT II The application must protect the confidentiality and integrity of stored information when required by DoD policy or the information owner.

描述

Potential Clickjacking on Legacy Browsers\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://10.10.2.164/CxWebClient/ViewerMain.aspx?scanid=85703&projectid=31418&pathid=1
狀態	反覆出現的問題
Detection Date	6/13/2024 10:45:28 AM

應用程式未使用 framebusting 腳本保護 cicd-form-frontend-checkmars/src/app/app.component.html網頁免受舊版瀏覽器中的 clickjacking 攻擊。

	來源	目的地
檔案	cicd-form-frontend-checkmars/src/app/app.component.html	cicd-form-frontend-checkmars/src/app/app.component.html
行	1	1
物件	<	<

代碼片斷

檔案名稱 cicd-form-frontend-checkmars/src/app/app.component.html

方法 <app-navbar [logoutTime]="logoutTime"> </app-navbar>

```
....
1.  <app-navbar [logoutTime]="logoutTime"></app-navbar>
```

Potential Clickjacking on Legacy Browsers

風險

可能發生什麼問題

Clickjacking 攻擊允許攻擊者構築無形的應用程式並將其疊加在偽冒網站上，藉此「劫持」使用者在網頁上的滑鼠點擊。使用者以為自己在點擊網站的連結或按鈕，但實際上點擊的是一個看不見的惡意網頁。
這使攻擊者能夠設計覆蓋層，點擊後會導致在脆弱的應用程式中執行非預期操作，例如啟用使用者的網路攝影機、刪除所有使用者紀錄、更改使用者設置或導致 clickfraud。

原因

如何發生

Clickjacking 漏洞的根本原因是應用程式網頁可以被載入到另一個網站的框架中。應用程式未實作可以防止頁面被載入到其他框架的 framebusting 腳本。注意還有很多類型的簡化重新導向腳本會使應用程式容易受到 Clickjacking 技術的影響，因此建議不要使用。

大部分現代瀏覽器可以支援應用程式所發佈的 Content-Security-Policy 或 X-Frame-Options header，從而禁止框架，避免此漏洞。但舊版瀏覽器不支援此功能，因此需要在 Javascript 中手動實作一個 framebusting 腳本，已保證在舊版本瀏覽器中也不會遭遇攻擊。

一般建議

如何避免

通用指南：

- 在伺服器端定義並實作內容安全策略 (CSP)，包括 frame-ancestors 指令。在所有相關網頁上實作 CSP。
- 如果需要將某些網頁載入到框架中，請定義具體的白名單目標 URL。
- 也可在所有 HTTP responses上返回一個 "X-Frame-Options" header。如果需要允許將特定網頁載入到框架中，可定義具體的白名單目標 URL。
- 對於舊版本瀏覽器的支援，可使用 Javascript 和 CSS 實作 framebusting 程式，確保頁面被框架化後不會顯示，並嘗試導航到框架以防止攻擊。即使無法導航，頁面也不會顯示，因此沒有交互性，也可以減少受到 Clickjacking 攻擊的機會。

具體建議：

- 在客戶端上實作不容易受到 frame-buster-busting 攻擊的 framebuster 腳本。
  - 代碼應該先禁用 UI，這樣即使成功繞過防框架代碼，也無法點擊 UI。這可以通過在 "body" 或 "html" 標籤中，將 "display" 屬性的 CSS 值設為 "none" 來禁用UI。這樣做是因為，如果框架嘗試重新導向並成為主視窗，仍然可以通過各種技術阻止惡意主視窗重新導向。
  - 然後程式應通過比較 self === top 來確定是否沒有框架發生；如果結果為 true，則可以啟用 UI。如果為 false，可將 top.location 屬性設置為 self.location 來嘗試離開框架頁面。

程式碼範例

JavaScript

Clickjackable Webpage

```
<html>
    <body>
     <button onclick="clicked();">
            Click here if you love ducks
        </button>
    </body>
</html>
```

Bustable Framebuster

```
<html>
```

```
    <head>
     <script>
            if ( window.self.location != window.top.location ) {
                  window.top.location = window.self.location;
            }
        </script>
    </head>

    <body>
     <button onclick="clicked();">
            Click here if you love ducks
        </button>
    </body>
</html>
```

Proper Framebusterbusterbusting

```
<html>
    <head>
     <style> html {display : none; } </style>
        <script>
            if ( self === top ) {
                  document.documentElement.style.display = 'block';
            }
            else {
                  top.location = self.location;
            }
        </script>
    </head>

    <body>
     <button onclick="clicked();">
            Click here if you love ducks
        </button>
    </body>
</html>
```

檢測的語言

語言	HASH值	變更的日期
JavaScript	5693733879119650	2024/6/11
VbScript	0386000544005133	2023/4/6
Common	1330881790325397	2024/6/11