《前端開發資安入門》附錄
安全性檢查表

這張檢查表都彙整了書中所有提及的漏洞相關防範策略。歡迎各位在實現 Web 應用程式時可以多加利用。
每項檢查內容都有介紹，不過更詳盡的內容還是請移駕本書主要篇章進行閱讀。

策略目的	講解章節	檢查內容	
加密通訊（HTTPS）	3.3.1	使用 HTTPS 傳輸資源。	
	3.3.1	受信任的授權單位（CA）所發行的憑證。	
	3.3.5	確認沒有發生 Mixed Content 的情況。	
	3.3.6	設定當被使用 HSTS 存取時會重新導向至 HTTPS。	
	3.3.6	啟用 HSTS 時，將 Strict-Transport-Security 標頭附加到網頁標頭中。	
	3.3.6	想讓子網域的 HSTS 也生效時，需要在 Strict-Transport-Security 標頭附加 includeSubdomain。	
	3.3.6	使用 HSTS Preload 時，需要在 Strict-Transport-Security 標頭附加 preload。	
	3.3.6	使用 HSTS Preload 時，需要在 Strict-Transport-Security 標頭附加 includeSubdomain。	
	3.3.6	使用 HSTS Preload 時，需註冊到 Preload List。	
	3.3.6	為 HSTS 設定有效時間時，需要在 Strict-Transport-Security 標頭附加帶有合適的值的 max-age。	
跨來源通訊	4.4.2	Access-Control-Allow-Origin 的值只允許想要存取的來源。	
	4.4.2	只有當允許所有來源存取也沒問題時，才將 Access-Control-Allow-Origin 的值設定為 *（萬用字元）。	
	4.4.3	Access-Control-Allow-Method 的值不可包含不必要的 HTTP 方法。	
	4.4.3	Access-Control-Allow-Headers 的值不可包含不必要的 HTTP 標頭。	
	4.4.3	Access-Control-Max-Age 的值需配合資源特性進行適切的設定。	
	4.4.4	使用 fetch 函式傳送帶有 Cookie 的請求到跨來源時，需設定合適的 credentials。	
	4.4.4	使用 XMLHttpRequest 傳送帶有 Cookie 的請求到跨來源時，要設定 withCredentials=true。	

跨來源通訊	4.4.4	欲允許來自跨來源、帶有 Cookie 的請求時，回應當中需設定為 Access-Control-Allow-Credentials: true。	
	4.4.4	附加 Access-Control-Allow-Credentials 標頭時，Access-Control-Allow-Origin 的值不可為 *（萬用字元）。	
	4.4.5	想要限制有使用 fetchc 函式的請求的接收端必須是相同來源時，需在 fetch 函式的 option 引數指定 mode: same-origin、或是 mode: no-cors。	
	4.4.6	要讓 <canvas> 讀取來自跨來源的圖像時，需要設定 <img> 元素的 crossorigin 屬性。	
	4.4.6	為控制 Cookie 的傳送，HTML 元素的 crossorigin 屬性需要設定合適的值。	
	4.6	跨越 iframe 傳輸資料時，接收端的 iframe 須要檢查傳送端希望允許存取的來源。	
	4.7.2	使用 SharedArrayBuffer 時、或是不希望降低 Performance 系列的 API 時，需啟用 Cross Origin Isolation（COOP+COEP）。	
	4.7.3	只有 self.crossOriginIsolated 的值為 true 時，才能使用 SharedArrayBuffer。	
跨站腳本攻擊（XSS）防範策略	5.2.5	要將使用者輸入的字串插入 HTML 時，該字串需要執行跳脫處理。	
	5.2.5	要將使用者輸入的字串作為 HTML 元素的屬性值來插入時，該字串需要用加上引號。	
	5.2.5	要將使用者輸入的字串插入 <a> 元素的 herf 屬性時，需判斷該字串是否為 http: 或 https: 起首的 URL 字串。	
	5.2.5	要將使用者輸入的字串插入 HTML 時，建議使用 appendChild 函式、來取代像是 innerHTML 這類的接收器（sink）。	
	5.2.5	包含機密資訊的 Cookie 需要加上 HttpOnly 屬性。	
跨站腳本攻擊（XSS）防範策略	5.2.5	要將使用者輸入的字串插入 HTML 時，需要排除 DOMPurify 等用來防範 XSS 的函式庫、或是因為使用 Sanitize API 而可能導致 XSS 的字串。	
	5.4.1	套用 Content-Security-Policy（CSP）。	
	5.4.1	CSP 的值不可指定為 unsafe-inline。	
	5.4.1	CSP 的值不可指定為 unsafe-eval。	
	5.4.1	CSP 的值不可指定為 unsafe-hashes，與 HTML 元素串接的事件處理器需使用 JavaScript 的 addEventListener 進行設定。	
	5.4.1	針對需要控制的執行跟讀取的資源，設定必要的 CSP 指引。	

跨站腳本攻擊（XSS）防範策略	5.4.1	不在 CSP 的各個指引的值當中去指定不必要的主機名稱。	
	5.4.2	需要為 CSP 的值允許 Inline Script 時，建議使用 nonce-source 或 hash-source，避免使用 unsafe-inline。	
	5.4.2	不為 CSP 的各個指引的值指定主機名稱，而是使用 nonce-source 或 hash-source。	
	5.4.2	運用設定了 nonce-source 的 CSP 時，nonce 的值需為難以猜測的值、且必須每次請求時都更改。	
	5.4.2	倘若 HTML 無法配合每次的請求來動態產生，就需要透過使用了 hash-source 的 CSP 來進行設定。	
	5.4.2	想要動態讀取 Script 時，運用 strict-dynamic 來動態產生 <script> 元素。	
	5.4.2	將 CSP 的值指定為 object-src ‘none’。	
	5.4.2	將 CSP 的值指定為 base-uri ‘none’。	
	5.4.3	允許插入 Script 或允許讀取外部 Script 時，需要使用 Trusted Types。	
	5.4.3	不建立不需要的 Trusted Types 的 policy 函式，也別在 CSP 標頭指定不必要的 policy。	
	5.4.4	使用 CSP 跟 Trusted Types 之前，需運用 Report-Only 模式進行充分地確認。	
	5.4.4	啟用 CSP 跟 Trusted Types 後，也須持續蒐集相關報告。	
跨站請求偽造（CSRF）防範策略	6.1.2	使用一次性權杖防範 CSRF 時，需確認嵌入表單內的 CSRF 權杖跟伺服器所持有的權杖是否一致。	
	6.1.2	當表單內有嵌入 CSRF 權杖時，該權杖必須以 type=hidden 隱藏起來。	
	6.1.2	CSRF 權杖必須是難以猜到的字串、且會依據每個 Session 來進行變更。	
	6.1.3	使用 Double Summit Cookie 防範 CSRF 時，要發行持有 CSRF 權杖值的 Cookie、並將該權杖放入請求標頭（或請求主體）來送出。伺服器端需要驗證 Cookie 與請求雙方的權杖是否一致。	
	6.1.4	將 Cookie 的 SameSite 屬性值設定為 Lax 或 Strict（有些瀏覽器會預設為 Lax）。	
	6.1.5	為了防範 CSRF，當伺服器端可以確認來源時，除了接收已允許的請求來源之外，拒絕其他來源的請求。	

點擊劫持防範策略	6.3.2	將不打算使用 iframe 嵌入到其他來源頁面的回應標頭設定為 X-Frame-Options: SAMEORIGIN 或 X-Frame-Options: DENY（或是使用 CSP frame-ancestors 也行，擇一即可）。	
	6.3.2	將不打算使用 iframe 嵌入到其他來源頁面的回應標頭設定為包含 frame-ancestore ‘none’ 或 frame-ancestors ‘self’ 的 CSP（或是使用 X-Frame-Options 也行，擇一即可）。	
開放重定向防範策略	6.5.2	運用使用者可輸入的 URL 字串來轉址時，限制轉址來源必須為特定 URL 或來源才可執行。	
驗證遭受攻擊時的因應策略	7.2.3	在進行開發時如果使用 GitHub 等服務，務必要啟用多重要素驗證。	
	7.2.3	搭載多重要素驗證。	
	7.2.3	使用者輸入密碼錯誤達一定次數時，鎖定帳號。	
	7.2.3	確認（Validation）密碼欄位的輸入內容。	
	7.3.4	搭載輔助輸入密碼的功能。	
	7.3.4	在輸入密碼的畫面上確認資料有無被傳送到 Web 分析服務去。	
	7.3.4	確認機密資訊是否適合儲存在網頁儲存服務上。	
使用安全的函式庫	8.3.1	檢查正在使用的函式庫有無漏洞存在。	
	8.3.2	避免使用已經沒在維護的函式庫。	
	8.3.3	使用最新版本、或者漏洞已經解決的函式庫。	
	8.3.4	讀取 CDN 等外部函式庫時，驗證有無遭竄改。	
	8.3.5	讀取 CDN 等外部函式庫時，指定無漏洞的版本。	