講到跨來源資料共用CORS (Cross-Origin Resource Sharing)就也會提到另一個同源政策 (Same Origin Policy)，同源政策是網站安全的基礎。 https://domain-a.com 只能存取自己網站裡的資源 (圖片、影片、程式碼等)，不允許網站 https://domain-b.com 來存取。想要存取跨來源資源必須在某些特定情況下才被允許。畫清楚界線我的網站我自己管，別人讀取不到也改不了。這很合理，不然若我的網站被別人任意修改、被惡意人士讀取機密資訊那就糟糕了！若沒有這層防護，壞人就可以任意新增刪除我們 Facebook 裡面留言了、也可以輕易登入我們銀行帳戶。要判斷同不同源只要 scheme、domain、port 一樣就會被視為同源 (same-origin)，其他則是不同源，例如https://domain-a.com:20/tom跟https://domain-a.com/mike是同源的跟https://domain-a.com:30是不同源的。但在某些情況下跨來源是被允許的，不受同源策略限制，像是跨來源嵌入通常被允許 (embed)、跨來源寫入通常被允許 (writes)、像<script src=”…”></script>、<link rel=”stylesheet”href=”…”>、<iframe>、圖片<img>、<video>、<embed>.等等都是跨來源嵌入。可以在藉由<form>在domain-a.com發request給 domain-b.com，當然透過連結links或直接redirect到別的網站也是被允許的。可是跨來源讀取通常被禁止 (reads)，domain-a.com不能讀取domain-b.com的 cookie、XMLHttpRequest，Fetch API也都無法被讀取，會回報錯誤。

Same Origin Policy雖然不錯，因為他防止了一些惡意的script攻擊，但總不會每一個跨網域都是惡意的; 也不可能一間公司擁有所有的資源，有時還是必須串接第三方資源，例如Facebook API、Google Map、政府釋出的公開API等。跨來源資源共用標準的運作方式是藉由加入新的HTTP標頭讓伺服器能夠描述來源資訊以提供予瀏覽器讀取。另外，針對會造成副作用的HTTP請求方法（特別是GET以外的HTTP方法，或搭配某些MIME types的POST方法），規範要求瀏覽器必須要請求傳送「預檢（preflight）」請求，以HTTP的OPTIONS (en-US)方法之請求從伺服器取得其支援的方法。當伺服器許可後，再傳送HTTP請求方法送出實際的請求。伺服器也可以通知客戶端是否要連同安全性資料（包括Cookies和HTTP 認證（Authentication）資料）一併隨請求送出。當然後端可以把權限開到最大讓任何人都可以讀取，不受同源政策的限制，但跨來源還是有許多風險的，所以還是不要直接 \* 開好開滿。

而且就算有同源政策的保護，網站也不一定是安全的，Same Origin Policy 只是最基本的保護而已，實際上 attackers 還是可以聰明地找到漏洞攻擊你的網站。例如 Cross-site scripting (XSS) 可以欺騙網站來繞過同源政策的保護，這是一個很大的問題，因為同源政策下瀏覽器信任底下所有資料存取都是安全的，但這樣被惡意注入利用很有可能就讓你網站掛掉，更嚴重是用戶敏感資料被洩漏，應該被保密的資料被壞人利用。要確保自己網站的安全，還是需要下功夫的，