

`HTTP` 是一個「無狀態協議 Stateless Protocol」，也就是說，每次從客戶端（Client）對伺服器（Server）發出的請求都是獨立的 — 這一次的請求無法得知上一次請求的內容與資訊。也因此，最近開始在學習將認證系統加入餐廳清單的應用程式當中時，就遇到了有趣的問題：既然 `HTTP` 是一個無狀態協議「那伺服器該如何辨認不同的請求是來自同個瀏覽器？」或「使用者登入後，伺服器如何在往後的請求中，辨認使用者其實是已經通過驗證（已登入）的狀態？」 — 透過運用 **Cookie** 和 **Session** 可以解決這些問題。

Cookie 是伺服器（Server）傳送給瀏覽器（Client）的一小片段資料，並請瀏覽器保存起來，以便往後向相同的伺服器發送請求時，附上這 **Cookie** 的資料。

Cookie 常見用途

1. 儲存和追蹤使用者行為
2. 儲存用戶登入、購物車等伺服器所需的資訊
3. 儲存使用者設定和偏好等

基本用戶登入流程範例

1. 使用者透過瀏覽器登入畫面登入
2. 伺服器端通過驗證後，可以將使用者「已經登入」的資訊附在 **Cookie** 中回傳，並請瀏覽器保存起來
3. 往後，每當瀏覽器對伺服器發出請求時，會一併附上存有使用者「已經登入」狀態資訊的 **Cookie** 紿伺服器
4. 伺服器透過 **Cookie** 就能辨識這位使用者已經通過驗證了

使用 Cookie 傳送重要資訊的安全性隱憂

透過 Cookie, 我們的確達成目標, 讓伺服器可以在往後透過客戶端發出的請求, 辨識使用者及其登入狀態, 非常方便。然而, 有趣的是, 這些資訊其實用戶是有機會可以在瀏覽器中修改的, 因此使用者能透過串改 Cookie 上的內容, 讓伺服器收到不正確的訊息 — 也就是說, 以登入的例子來看, 使用者可以在串改 `sLoggedIn` 的值, 讓伺服器誤以為使用者已經通過認證

瀏覽器和WEB服務(或是Server)之間, 我們知道是透過HTTP協議來進行通訊, 而這樣的協議是一種無狀態的協議, 簡言之當我們操作某些功能時, 例如搜索, WEB服務應用端就會接收到為因應我們的需求所發出來的Request, 並作出相對應的響應, 完成響應之後就會結束與該用戶的連接, 例如傳統飲料店我們點餐後, 老闆娘只能憑藉著記憶來幫你作出你點的飲料, 而這個老闆娘的記憶力相當不好, 並不會因為你是熟客就記錄你每天的點餐習慣, 每天都會當成你是第一次來買飲料的人, 而當你需要發出下一次Request, 例如取飲料時, 他才會反應到說喔你的飲料好了(或沒好), 在處理上是相當的不便捷的。所以我們可以比擬成我們就是使用服務的Client端, 而老闆娘的腦袋就是所謂的Server端, 那Cookie是什麼呢?

Cookie 技術是網路時代的快速變遷下所誕生的產物, 隨著網路的快速發展, 人們需要更複雜的網上交互活動, 讓伺服器與客戶端可以保持長久的互連與活動狀態, 而這項技術初衷也在瀏覽器的發展時, 得到了基礎上的硬用, 在1993年時, Lou Montulli 為了讓用戶在訪問網站時, 可以提高訪問速度, 並減少現有的人工填寫及個人話應用, 而發明了現在被大家廣為人知的Cookie.

早期的概念上Cookie是由Web server端產生的, 作為發送給給User-Agent (一般是瀏覽器端) , 而當瀏覽器接收到Cookie後, 會將其中的key/value, 保存到某個路徑內的文本文件之中, 讓下次於造訪同一網站時, 就可以將Cookie自動發送給Web Server端。

從這樣的概念可以知道, Cookie(複數形態為Cookies)是屬於一種小型的文字檔案, 透過加密的方式儲存在用戶端 (Client Side)上的資料, 一般來說 cookie 會紀錄用戶的資訊, 比較常見的做法是應用在購物車、會員登入或瀏覽紀錄、停留時間等等的, 使Web service 可以透過辨別用戶身分, 來取得相關的資訊。

而正因為他是儲存在我們用戶的本機端，通常可以儲存的地方有兩個：記憶體或硬體內，記憶體是由Browser(瀏覽器)來維護的，通常會在瀏覽器關閉後清除，而各個瀏覽器之間的Cookie是無法相互使用，也就是說對於在同一台電腦上使用Chorme或是Firefox，儘管操作的是同一個人，卻是會認成兩個不一樣的角色，而硬體的Cookie則會有一個保存期限，除非過期或是手動刪除，不然他的儲存時間會較瀏覽器來的長。以電子商務網站舉例，因為HTTP的無狀態性，伺服器並不知道使用者在每個頁面跳轉時到底帶入了什麼資訊，而Cookie就是用來繞開HTTP的無狀態性的「額外手段」之一，讓伺服器可以設定或讀取Cookies中所包含資訊，藉此維護使用者在使用服務時，在背景完成並可以持續跟伺服器發送請求以及對談中的狀態。

而像是登入的應用為例，就是使用者登入一個網站時，伺服器端往往會請求用戶輸入使用者帳號及密碼，並且用戶可以勾選「下次自動登入」，這就是觸發使用Cookie的開關了，如果勾選了，在使用者前一次登入時，伺服器就會傳送了包含登入憑據(使用者名稱加密碼的某種加密形式)的Cookie到使用者的硬碟(或記憶體上)，在之後登入時，只要Cookie尚未到期，瀏覽器會傳送該Cookie給伺服器作驗證憑據，來減少重複登入的輸入行為。而Cookie属性除了name(名)和value(值)之外，還有以下四種可選擇的屬性，來控制cookie的保存期限，作用網域及安全性：1.expires: 表示Cookie的保存期限，在默認的情況下為暫時性的cookie，只要關閉瀏覽器就會消失2.path: 指定與cookie關連在一起的網頁，默認的狀況下為和當前網頁同一目錄的網頁中有效。3.domain: 設定cookie有效的網域名稱，可以和path一同設定，讓相同/類似的domain可以享有同樣的cookie4.secure:算是cookie的安全值，在默認的情況cookie的傳輸上是不安全的，可以通過一個不安全且一般的http，若設置為安全的狀況下，可以讓cookie只在安全的http上進行傳輸

但是其實我們都知道，像是瀏覽記錄也可以算是cookie的一種，許多人員在使用cookie的狀況下，常會造訪一些"安全性"不夠，或是一些不太合法的網站(例如希爾頓娛樂大酒店、澳門最大線上賭場開幕啦~等等)，而這些不僅會有資訊安全上的風險，更是讓隱私曝光，而若使用的又是公司電腦更是滋事體大，而有些廣告公司更事會把一些資訊寫入cookie來達到發送垃圾訊息的目的。

除上述議題之外，若你的電腦裡有多個瀏覽器，個別在不同空間存放cookie，而若你使用單一瀏覽器卻在不同的電腦上作操作，也會得到不同的cookie訊息，但cookie本身無法準確的區分身分，需要結合Session來作到同步的辨識，才可以有效的作身分上的區分；例如以飲料店點餐為例，Cookie就像是你取號時領到的單據，上面應該還要記載什麼資訊，才能讓領號碼牌叫號、點餐並且取餐的機制作得更為完備，就讓我們在下一章為大家介紹吧！

HTTP (超文本傳輸安全協定) cookie (小餅乾或小型文字檔案) 是一段由伺服器送給使用者瀏覽器的一小塊資料。

瀏覽器會儲存它並且在瀏覽器下一次發送要求的時候將它送回原本送來的伺服器。基本上，它是用來區分兩個要求是來自同一個瀏覽器 — 以此去保持使用者的登入狀態。例如，它提供了保存狀態資訊的功能，來幫助HTTP這個無法紀錄狀態的通訊協定。

雖然HTTP廣泛的運用並且用來做很多神奇的事情，大家都習慣了。但是HTTP這個通訊協定其實是沒有辦法保留狀態的(這要由TCP開始說起，本文不太探討這個)。所以光使用HTTP使無法紀錄使用者登入的，必須要外加其他功能才有辦法，其中一個廣泛使用的方式就是使用Cookie。但是因為Cookie太廣泛使用了，現在只要是使用HTTP就一定會使用Cookie，所以才會造成新手有一種錯覺：“Cookie是HTTP通訊協定所包含的一種功能”。實際上，比較準確的說法應該是：“Cookie是一種可以跟HTTP一起使用的功能，並且被廣泛使用”。它多半是使用HTTP Request的Header區塊來實作。

