MMDays - 網路, 資訊, 觀察, 生活

網路, 產業, 資訊, 觀察, 生活, 電影, 技術, 新知, 科技, 媒體, 趨勢, Web 2.0

資訊安全雜談：OWASP TOP 10 2013

Dec 11th, 2013 by Mr. March

Posted by Mr. March

以前在學校上課的時候曾經聽一名資安專家的演講者打過一個比喻，資訊安全的防護就好比像一條鍊子，這條鍊子的耐用強度取決於鍊子上最弱的那一個環節。資訊安全的問題不單單是使用者本身習慣不良的問題，網頁應用程式的開發者也是有一定的責任的。

關於資安的議題有非常多可以探討的空間，但今天筆者想要介紹的是 OWASP 這個組織，和他們今年發表的OWASP TOP 10 2013。

圖：OWASP組織的Logo

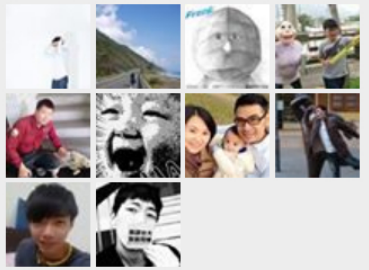
OWASP的全名是Open Web Application Security Project，關於這個團體的成立宗旨，他們是這樣說的：

> OWASP(開放Web軟體安全計畫 – Open Web Application Security Project)是一個開放社群、非營利性組織，其主要目標是研議協助解決Web軟體安全之標準、工具與技術文件，長期致力於協助政府或企業瞭解並改善網頁應用程式與網頁服務的安全性。

而OWASP TOP 10是他們目前手頭上眾多計劃中最知名的一個，內容為他們認為網頁應用程式上最重要且嚴重的10大弱點且進行排名，在今年他們發布了最新的一個版本，詳細的內容如下(以下由第10名介紹到第1名)：

> A10-Unvalidated Redirects and Forwards (未經驗證的重新導向與轉送)：

如果你的網站有在網址列裡使用重新導向功能的話那可能要特別小心了。



最新文章

- 網路公司如何行銷「有機產品」
- 網路公司如何開發「有機產品」
- 台灣新生代企業的面貌：網路公司
- 從服貿官員發言談網路，看政府失能的癥結
- VoiceTube：看 YouTube 學英文
- 免費看電影軟體 Popcorn Time 將顛覆電影產業
- 吃不到的大餅？由服貿前的經濟發展開始看
- 從封殺五月天與蔡康永看服貿協議的荒謬性
- 一個科技部的誕生，事小；一個沒有美感的政府，事大。
- 創新是一條艱苦的路 (至少在台灣)

範例：http://www.example.com/redirect.jsp?url=evil.com

第一個範例所造成的問題，是有心人士可以利用這個功能來製作釣魚網站，將網址掛在你的網域下，實際上是連到其他網站去。如果是有高知名度且受到民眾信任的網域，那麼就非常有可能造成一般民眾的受害。

範例：http://www.example.com/boring.jsp?fwd=admin.jsp

第二個範例則是會跳到這個網站內的其他位址，去進行一些使用者未授權進行的行為。

防範方式：不要用這個功能，一定要用的話一定要對傳進來的參數進行驗證。

> A9-Using Components with Known Vulnerabilities(使用已知漏洞元件)：

現在的網站很常使用第三方的元件，但是有些時候這些元件或是函式庫其實是有問題的，就有可能會受到攻擊。

防範方式：最簡單的方法是所有的元件都自己寫，但是這是不可能的，而基本上所有的第三方元件都會在最新版本裡修復漏洞，所以就隨時關注最新消息和更新到最新版本吧！

> A8 – Cross Site Request Forgery (CSRF) (跨站冒名請求)：

惡意的HTTP指令被當成合法的指令來執行，例如

這樣子一行的HTTP指令就會去執行上述的網址，進而執行惡意的行為，此手法常見於各種Web2.0的互動網站，也就是在留言板等地方留言時塞入HTML碼。

防範方式：在使用者在input欄位裡輸入資料時加上token，來避免使用者偷渡惡意的指令進入網站；並且使用CAPTCHA等驗證方式來防止人為的大量散布惡意連結。

> A7 – Missing Function Level Access Control (缺少功能級別的存取控制)：

惡意的入侵者可能會嘗試著去試你的網站上其他網頁的網址，而如果你不對每個頁面進行存取控制權限的要求的話，如果直接被攻擊者嘗試到你的網址，那他可以直接癱瘓你的整台主機。根據筆者的架站經驗，這種try網址的攻擊方式在網路上非常的常見，以前在學校寫專題自己架網站的時候，每天翻log都可以看到至少十幾筆以上的惡意嘗試網址的紀錄。

防範方式：可以先預設每個網頁都要辨識身分後才能存取，並且做好認證管制，並確認每個頁面都有做好權限的控制，沒有按鈕可以連結過去的頁面並不代表攻擊者不會手動連過去，要特別注意這一點。

> A6 – Sensitive Data Exposure (敏感資料暴露)：

這部分主要有兩個問題，一個是網站在與使用者傳輸敏感資料時未使用SSL加密連線，導致傳輸有可能被攔截竊聽造成使用者的帳密被盜用；第二個問題是儲存使用者的敏感資料例如帳密時未加密後在儲存。這當然是一個大問題，因為這表示如果這個網站的資料庫被攻破時，將會直接洩漏出你的帳密出去 (而且網站的

管理員可以直接看到你的密碼，有沒有覺得很可怕？)，而我們自己都很清楚，我們每個人常用的帳密通常也就是那幾組 (如果讀者真的能做到一網站一帳密，那麼筆者對於你對帳密的記憶能力真的是深深佩服)，一旦一個網站被攻破，那麼這表示許多網站都會遭受池魚之殃，關於有那些網站密碼沒加密和如何調查的問題可以參考這個網站：我的密碼沒加密

防範方式：通通使用SSL安全連線來傳輸；密碼必須使用不可逆的演算法加密後再存在資料庫裡，登入時只需比對加密後的結果是否相同。

> A5 – Security Misconfiguration (不當的安全組態設定)：

關於自己的網站安全設定沒有設定好，常見的有以下幾種問題：第一，未刪除或更改所使用套件的預設帳密，攻擊者可以輕而易舉地透過嘗試法直接入侵；第二，Directory listing未關閉，攻擊者可以透過此功能輕易地找出所有網站上的檔案，並且獲知你的原始碼；第三，錯誤訊息直接回傳在使用者頁面上，此舉會透漏許多額外的訊息給予攻擊者；第四，未刪除套件所附的範例應用程式，有許多的範例都是有漏洞的。

防範方式：軟體和作業系統更新至最新的patch；不需要的port與頁面和服務，通通關閉；預設的帳密務必要進行更改。

> A4 – Insecure Direct Object References (不安全的物件參考)：

攻擊者利用網站自身的檔案讀取功能，去任意的讀取敏感資料或重要檔案，進而分析這些檔案後，達到攻破網站的目的，這個問題主要的部分在於網頁編寫時所使用的原始碼裡沒有去驗證使用者所投入的字串是否合法，如以下的範例：

>

> String query = “SELECT * FROM accts WHERE account = ?”;
>
> PreparedStatement pstmt = connection.prepareStatement(query , …);
>
> pstmt.setString(1, request.getParameter(“acct”));
>
> ResultSet results = pstmt.executeQuery( );

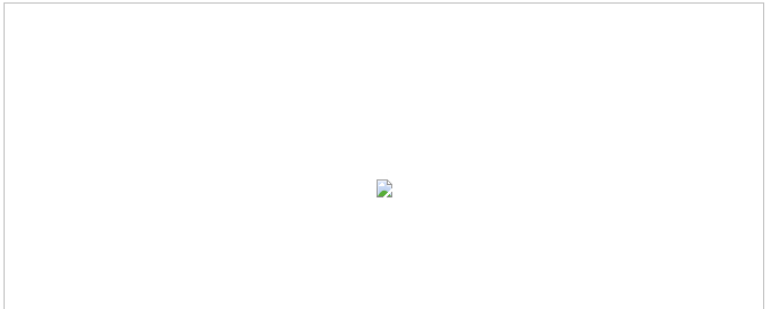
這是一段有問題的程式碼，攻擊者可能可以透過修改下列那一行網址中紅字的部分，來任意取得他人帳戶的資料。

> http://example.com/app/accountInfo?acct=notmyacct

防範方法：針對網頁裡的每個物件設定正確的存取權限限制，並且在存取時確認使用者是否真的有權限存取。

> A3 – Cross-Site Scripting(XSS) (跨站腳本程式攻擊)：

這是目前最流行的攻擊方式之一，容我借用一張網路上的圖來說明



(圖源: http://www.chmag.in/article/aug2010/advance-xss-attacks-dom-based)

使用者先與伺服器進行密碼的驗證拿到了cookie，這時執行了攻擊者埋入在頁面中有問題的script(一樣是透過Web2.0型的網頁上的互動留言等等功能)就可以順利竊取使用者的cookie然後就可以假冒他人身分做任何事情啦！

防範方法：不管是使用者輸入還是頁面輸出都應該要有檢查的機制；可以使用白名單機制；過濾掉有問題的字串(例如PHP可以使用htmlentities)

> A2 – Broken Authentication and Session Management (失效的驗證與連線管理)：

這一點中有些部分與前面的部分是有重複的，主要的部分有：將使用者的SESSION ID曝露在URL中，我們不能保證使用者不會進行螢幕截圖或是將URL傳給任何他所認識的人，所以這是很危險的。還有未將SESSION設定Timeout時間，如果使用者在公用電腦登入卻未登出，那麼下一個使用者將可以直接以上一位使用者的身分登入。其他的部分還有未使用SSL加密連線登入，密碼儲存時未加密等等前面幾點已經提到過的問題。

防範方法：使用SSL加密連線機制，不要將SESSION ID暴露在URL中，要有完善的SESSION保護的機制，設定一個Timeout的機制，密碼儲存一定要加密。

> A1 – Injection (注入攻擊)：

SQL Injection可以說是最常見也最有名的問題了，在這裡筆者先請大家看一條新

聞：駭客界林志炫 盜改1.2萬筆個資

這條新聞裡就是使用了SQL Injection攻擊，輕鬆的就毀滅了一個網站的資料庫。還好該公司資料庫有定期備份，不然後果就不堪設想。

這邊舉維基上SQL Injection條目裡的範例做說明：

某個網站的登入驗證的SQL查詢代碼為

```
strSQL = "SELECT * FROM users WHERE (name = '" + userName + "') and (pw = '
```

惡意填入

```
userName = "1' OR '1'='1";
```

與

```
passWord = "1' OR '1'='1";
```

時，將導致原本的SQL字串被填為

```
strSQL = "SELECT * FROM users WHERE (name = '1' OR '1'='1') and (pw = '1' O
```

也就是實際上執行的SQL命令會變成下面這樣的

```
strSQL = "SELECT * FROM users;"
```

因此達到無帳號密碼，亦可登入網站。所以SQL隱碼攻擊被俗稱為駭客的填空遊戲。

SQL Injection攻擊的方式就很像填空題，攻擊者在網頁裡任何可以輸入資料的地方試著去猜想設計者背後的語法撰寫方式，並去猜測完整的command應該會長成怎麼樣，還有推測欄位數，table的名字，SQL的版本資訊，試著去拼湊輸入一條SQL指令，輕則刪掉資料庫，重則竊取全部的個資。可以說是任何一個撰寫互動式網頁的開發者首先也必要處理的問題。

防範方式：錯誤的訊息不應該顯示給管理員以外的人，以防攻擊者得到更多有用的資訊；控管使用者的帳號權限；嚴密檢查使用者輸入的任何字串，檢查檢查再檢查，筆者認為身為一個開發者一定要假設任何使用者輸入的字串都可以作為攻

擊的用途，所以對使用者所輸入的任何字串都應該做過安全性的檢查。

這10條內容都是很嚴重，也很常見的弱點，害人之心不可有，防人之心不可無，身為網頁應用程式開發員可以說不可不注意不可不防範；就算不是開發者只是一般的使用者，筆者認為也應該要了解這其中的問題，至少你可以提早發現一些有問題的網站並做好適當的自我防範(例如盡量不使用有問題的網站所提供的服務，或避免留下過多的個人敏感資料)；個資外洩的問題目前可以說層出不窮，資訊安全議題的重要性由此可見一般。

以上是關於OWASP TOP 10 2013的簡單介紹，如果想要了解其中更詳細的內容可以直接拜訪他們的官網一探究竟。

情？為什麼iKala的app使用者數量沒有出現「有機的」成長？為什麼iKala在app ...

Sega Cheng — 你說的大陸這些低價峽體，上面跑的就是 Android 系統，這個系統是哪家公司寫出來的，不用我多做說明吧？...

吃不到的大餅？由服貿前的經濟發展開始看

3 comments • a month ago

Albert Wu —

台灣產業轉型的困境

4 comments • 5 months ago

徐仲威 — 完全認同，企業主根深蒂固的傳統觀念是無法轉型的主因。

Subscribe Add Disqus to your site DISQUS