避免網站遭受 SQL Injection 攻擊的 10 個步驟

資料來源 :WhiteHat Security | 翻譯整理 : 叡揚資訊 資訊安全事業處

資料竊取越來越普遍，黑市裡信用卡資料的價錢從 2006 年一筆 10 元變成 2009 年幾毛錢。客戶對於電子商務、網路銀行與其他電子商業活動失去信心。同時駭客竊取資料的方法也越來越聰明，造成越來越多企業成為他們的犧牲品。至 2009 年時法令與法規以更嚴格地保護消費者，但仍不斷發生新的事件。在 Verizon Business Data Breach Investigations Report 報告中，研究過去五年內的 600 個事件，其中 SQL Injection 是單一攻擊中獲取大量資料的手法。

這個結果並不令人驚訝!若設計網站應用程式的成員不了解 SQL Injection 攻擊時，是很容易發生的。通常事件發生時，信用卡公司(如 VISA)，會通知受害者駭客的活動，不過往往為時已晚。

SQL Injection 攻擊可能會對企業造成重大與昂貴的損失。攻擊的目標是資料庫，其中儲存了員工、客戶的資料。這類型的攻擊利用應用程式中的漏洞，操作從瀏覽器輸入的 SQL 查詢。

在 SQL Injection 攻擊中，惡意使用者可發送任意輸入至伺服器且假冒 web 應用程式產生與原先不同的 SQL 描述。因此資料庫伺服器會執行並取得與原先不同的結果。SQL Injection 攻擊是最常對資料庫獲得未經授權的存取與資料操作。

許多文章已討論如何進行 SQL Injection 攻擊。這篇著重於如何防禦。以下 10 個步驟分別考量開發人員與資料庫管理人員可以預防攻擊的方法。

對於資料庫管理者

1. **將資料庫獨立安裝，確保所有系統都在最新的版本：**

   在現今各種服務都有可能被攻擊的情況下，將資料庫與 web 應用伺服器安裝於不同的主機是比較好的做法。因此若資料存在於同一台主機內，當資料庫被攻擊時，攻擊者可輕易的攻擊 web 應用伺服器。另外，若不幸發生被主機關閉或毀損時，位於不同主機的資料可快速的回復既有作業。

2. **將資料庫中所有的預設帳號關閉，包括超級使用者：**

   資料庫的預設帳號，包含超級使用者、手冊中註明或是可以在網路上找到的。在 2002~2003 年有一個 SQL

GSS 叡揚資訊 Galaxy Software Services
台北總公司：10461 台北市中山區德惠街 9 號 5 樓 TEL：(02) 2586-7890 FAX：(02) 2586-8787
高雄辦公室：80453 高雄市明華路 315 號 13 樓之 3 TEL：(07) 586-6195
海外據點：上海、北京、東京 www.gss.com.tw

蠕蟲利用這些管理者未修改的預設帳號密碼攻擊資料庫，人們才開始注意到這類的問題。即使是現在，還是會發現有些管理者忘記移除預設帳號。攻擊者可以輕易的利用預設的帳號/密碼來測試是否可連線成功。若有需要使用超級使用者的帳號，建議新建一個不同名字的帳號而不要使用預設的。

3. **在資料庫中新建應用程式的使用者帳號，這個帳號是限制最小權限，只給允許存取必要的資料。移除所有範例資料表。禁止應用程式的使用者任意存取不必要的預存程序：**

針對不同的應用程式給予不同的資料庫帳號是比較好的做法。這個帳號應有限制資料的存取-只針對該應用程式的需求設定。若應用程式不需要使用任何的預存程序，則不允許使用。這想法是為了避免任何不必要的存取導致產生 SQL Injection 的弱點而設限的。

4. **SQL 查詢必需是透過應用程式且只能執行允許的查詢，如 Select, Insert, Update 等：**

大部份的應用程式都應對 SQL 查詢設限，偶爾透過應用程式刪除資料。若應用程式不需要執行 Drop 指令，那就不要允許使用。若有使用應用程式執行預存程序才提供權限，但需要在資料庫設定各應用程式可執行的部份為何。這麼做可以預防攻擊造成嚴重的損害以及保持資料庫的完整性。

5. **當情境是不需要 Insert 與 Updates 時，只能予唯讀(Read-Only)權限，如資料檢索或登入功能：**

網站中有許多功能其實只需要從資料庫取得資訊，如登入或資料檢索功能。完全不需要去 insert 或 update 資訊回資料庫；唯讀權限便可達到此需求，僅提供資訊也可預防資料庫被任意變更。透過這項設置，即使應用程式中有 SQL Injection 弱點，攻擊者也無法修改資料庫內容，保有資料的完整性。

對於開發人員

6. **在程式中對於輸入資訊進行消毒驗證：**

在 Web 應用程式中輸入驗證是最重要的。經過適當的輸入驗證，可避免大多數的攻擊包含 SQL Injection。輸入驗證可分為：

a. 資料型態：確認資料是正確的型態，如資料為數值，則進行數值轉換，若資料內容不是數值會出現錯誤訊息並回應給使用者。

b. 資料長度：檢查輸入資料長度，包含最小值與最大值。如使用者名稱最小 8 位最大 50 位，則輸入的內容必需位於此範圍內否則應回應錯誤給使用者。

c. 資料格式：另一個要注意的是輸入的格式，若是電話號碼則需確認格式為：xx-xxxx-xxxx，若輸入其他格式則拒絕該輸入。

其實上述的例子只是輸入驗證中的幾個例子。以下提供一個正規表示法(Regular Expression)供參考：

在理想的情況時，設計階段會定義可輸入資料的類型(字母、數字或字母數字都有)。以下為 Java 使用 Regex API(java.util.regex)的輸入驗證範例。

GSS 叡揚資訊 Galaxy Software Services
台北總公司：10461 台北市中山區德惠街 9 號 5 樓 TEL：(02) 2586-7890 FAX：(02) 2586-8787
高雄辦公室：80453 高雄市明華路 315 號 13 樓之 3 TEL：(07) 586-6195
海外據點：上海、北京、東京 www.gss.com.tw

使用者帳號驗證：

a. 可使用 a-z, A-Z, 0-9, @

b. 不可有空白

c. 最長 50 個字

d. 最短 8 個字

```
Pattern pattern = Pattern.compile("[A-Za-z0-9@]{8,50}+");
Matcher matcher = pattern.matcher(formname);
if(!matcher.matches())
errorMessage("Name contains illegal characters");
```

若系統允許數值輸入，在要與 SQL 查詢組合前第一件事應該將其轉換成數值。若發生 NumberFormatException 異常，便可得知不知正常的輸入。

```
try {
int onum = Integer.parseInt(order_num);
}catch(NumberFormatException nfe)
{
errorMessage("Only numeric characters allowed");
}
```

7. **使用參數化查詢而非動態組合查詢。以 JAVA 來說通常會使用 Prepared Statement 取代 Statement ：**

SQL Injection 會這麼成功的原因之一是開發人員使用動態組合查詢。攻擊者在動態組合查詢中插入跳脫字元(如單引號或雙引號)，截斷原先的查詢取得資料庫的錯誤訊息。當攻擊者取得錯誤訊息時，便可得知該系統具有 SQL Injection 的弱點且可以進一步的取得更多資料庫裡的資訊。

例如以下為一段有 SQL Injection 弱點的程式碼：

```
String username = request.getParameter("username");
String password = request.getParameter("password");
String sql = "select * from user_table where username = '" + username + "' and password = '" + password
+ "'";
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery(sql);
```


GSS 叡揚資訊 Galaxy Software Services
台北總公司：10461 台北市中山區德惠街 9 號 5 樓 TEL：(02) 2586-7890 FAX：(02) 2586-8787
高雄辦公室：80453 高雄市明華路 315 號 13 樓之 3 TEL：(07) 586-6195
海外據點：上海、北京、東京 www.gss.com.tw

以下則是安全的寫法：

```
String username = request.getParameter("username");
String password = request.getParameter("password");
PreparedStatement pstmt = con.prepareStatement("Select * from user_table where username = ? and password = ?");
pstmt.setString(1, username); //Validated input
pstmt.setString(2, password); //Hashed Encryption
ResultSet rs = pstmt.execute();
```

8. **採用適合的錯誤處理、系統紀錄(log)方式，以避面將資料庫錯誤、或任何技術資訊透漏給使用者**

攻擊者非常喜愛錯誤訊息。每當系統拋出錯誤訊息時多半提供大量可利用的資訊。攻擊者會嘗試透過不同的輸入取得不同的錯誤訊息。如此攻擊者可收集許多關於資料庫的訊息，如資料表、欄位等等。每個系統都應隱藏原始的錯誤訊息，只顯示一般的錯誤訊息像是「請重新再試一次」之類的。系統也不顯示過多的內容細節給使用者，以避免被濫用，而這些資訊應被記錄下來對於分析攻擊者行為非常有幫助。

9. **選擇不容被猜到的資料表與欄位名稱：**

當系統有 SQL Injection 弱點時，即使系統有其他上述的保護措施，攻擊者仍會嘗試猜測資料表與欄位名稱。以登入功能來說，通常會將資料表命名為「login」或「User」。因此建議以不易猜到的命名方式，如存放 email 的欄位可命名為：「x_eml」、「olb_mail_addr」而表格名稱可命名為「app_usr_det」。

10. **盡可能使用預存程序取代原始 SQL**

雖然預存程序也有 SQL Injection 風險，但較難從系統中操控 SQL 查詢。增加使用腳本或自動攻擊的困難。是縱深防禦的概念。

結論

其實防禦 SQL Injection 並不困難。只要幾個簡單的步驟，可大幅度的降低被攻擊的風險。若開發人員注意到這點，其實只需調整部份的撰寫風格，即能預防 SQL Injection。從過往的經驗，持繼對開發人員進行再教育對於降低系統弱點有非常大的幫助。設計更安全的網站讓資訊安全團隊與開發人員間的關係更加和協。

GSS 叡揚資訊 Galaxy Software Services
台北總公司：10461 台北市中山區德惠街 9 號 5 樓 TEL：(02) 2586-7890 FAX：(02) 2586-8787
高雄辦公室：80453 高雄市明華路 315 號 13 樓之 3 TEL：(07) 586-6195
海外據點：上海、北京、東京 www.gss.com.tw