程式設計師的資安第一課：密碼驗證

典型的應用程式，啓動後，第一件事是要求輸入使用者代號與密碼，如果你將使用者代號與密碼資料存放在資料庫裡，這時你要注意的第'零'件事是輸入檢核，以避免 SQL Injection 發生. 接者你要驗證密碼對不對，你最好不要真的去比對密碼值……

不要真的去撈出密碼值來與使用者輸入的密碼比對，因爲這樣你必須將密碼值以明碼方式存放在資料庫(或某檔案或 Registry)裡，你會冒了一個所有使用者代號與密碼洩漏的風險. 當然你可以將密碼欄位加密起來，但有更簡單的方式….

你可以將密碼以 Hash 的方式存起來(資料庫或檔案或 Registry 裡)，然後將使用者輸入的密碼值用同樣的 Hash Function 運算得到一個 Hash 值，再去比較你存下來的 Hash 值，如果相同就表示密碼值一樣，如果不同就是密碼不對. Hash 是一個單向的演算法，無法反運算回來，換言之，無法以 Hash Value 反推回原來的值. 這避免了你洩露密碼值的風險.

所以，你不要這樣做:

```
Username Password
-------- --------
james    god123
lucas    sex999
anna     love4ever
```

你應該這樣存:

```
Username Password
---------- ----------------------------------------------------------------
james    21298DF8A3277357EE55B01DF9530B535CF08EC1
lucas    68BB04BD54B8F6C530695E0B77DE298276A0511D
anna     9F2FEB0F1EF425B292F2F94BC8482494DF430413
```

沒有人可以由 21298DF8A32…..算出 god123
但一定要是 god123 才能得到 21298DF8A32…..

程式如何做 ? 你需要類似 MakeHash(value)這樣的 SQL Procedure 加到

你的 SQL 程式碼. Hash 的演算法有很多種, 例如 SHA-1, SHA-2, MD5…安全強度不一, 你可以選擇合適的來用. 關於 Hash 的更多資訊與用途, 可搜尋 Google 或 Wikipedia 網站.

用 Hash 來保護使用者的密碼只是您要學的第一個課題. 不要讓別人說程式設計師只會注重程式的寫作技巧,結果寫出漏洞百出, 風險極高的應用系統來

XP_Crypt 是一套寫好的 Stored Procedures,內含 Hash Functions,及其它各種國際標準的加解密演算功能, 可以很容易的崁入到您的 SQL 程式裡.

詳細資料請參考: http://www.asiapeak.com/XPCrypt.php