

淡江大學

程式碼安全漏洞修復說明

日期：99/6/22

叡揚資訊產品顧問 林榮秋

Willy_lin@mail.gss.com.tw

常見安全的弱點修復說明

1. SQL Injection
2. Cross Site Scripting (XSS)
3. HTTP Response Splitting
4. Command Injection
5. Path Manipulation
6. Cross Site Request Forgery (CSRF)
7. Password Management
8. Race Conditions
9. Error Handling
10. Misconfiguration

程式碼修復說明

SQL Injection



SQL Injection 漏洞問題

[X] 有安全漏洞的 SQL 程式碼 (使用字串相加方式)

```
string userName = ctx.getAuthenticatedUserName();  
string query = "SELECT * FROM items WHERE owner = "  
    "' + userName + "' AND itemname = "  
    "' + ItemName.Text + "'";  
    ' or '1'='1'  
sda = new SqlDataAdapter(query, conn);  
DataTable dt = new DataTable();  
sda.Fill(dt);
```

傳回所有 items Table 內的資料

預防 SQL Injection 漏洞的安全寫法

[V] 安全的寫法 (撰寫使用參數化 SQL 語法)

```
string userName = ctx.getAuthenticatedUserName();  
conn = new SqlConnection(_ConnectionString);  
conn.Open();
```

```
SqlCommand query = new SqlCommand(  
"SELECT * FROM items WHERE itemname=@ItemName  
AND owner=@OwnerName", conn);
```

```
query.Parameters.AddWithValue("@ItemName", ItemName.Text);  
query.Parameters.AddWithValue("@OwnerName", userName);
```

```
SqlDataReader objReader = objCommand.ExecuteReader();
```

查無資料 !

SQL Injection 問題

■ 修補漏洞的安全撰寫方式範例

```
objcon.Open()
sqlstr = "Insert INTO [Announcements] (Title, Content, Creator, CreateTime) "
sqlstr = sqlstr & " Values( @parameter1 , @parameter2 , @parameter3 , @parameter4 )"

conn = New SqlCommand(sqlstr, objcon)

conn.Parameters.Add(" @parameter1", SqlDbType.NVarChar,20).Value = institleTextBox.Text
conn.Parameters.Add(" @parameter2", SqlDbType.NVarChar,100).Value = InsertHtmlEditor.HTML
conn.Parameters.Add(" @parameter3", SqlDbType.NVarChar,10).Value = uid
conn.Parameters.Add(" @parameter4", SqlDbType.DateTime).Value = GETDATE()

cnt = conn.ExecuteNonQuery()
```

修改步驟解析：

[Step1]：原本相加的字串,改為 @ parameter1 ~ @ parameterN

[Step2]：使用 conn.Parameters.Add(@ parameter1 ~ N)
= 原本要相加變數

MS Help .Net Parameter 寫法的 Sample Code

DataAdapter 參數 (ADO.NET) - Microsoft Visual Studio 2008 文件 - Microsoft Document Explorer

檔案(E) 編輯(E) 檢視(V) 工具(T) 視窗(W) 說明(H)

索引

篩選(I): .NET Framework

尋找(L): Parameter 物件

parameter arrays
Parameter 方法
Parameter 列舉型別成員
Parameter 物件
Parameter 類別
Parameter 類別, 方法
Parameter 類別, 所有成員
Parameter 類別, 建構函式
Parameter 類別, 關於 Parameter 類別
Parameter 類別, 屬性
Parameter 屬性
Parameter.Clone 方法
Parameter.ConvertEmptyStringToNull 屬性
Parameter.DefaultValue 屬性
Parameter.Direction 屬性
Parameter.Evaluate 方法
Parameter.ICloneable.Clone 明確實作的方法
Parameter.IStateManager.IsTrackingViewState 明確實作
Parameter.IStateManager.LoadViewState 明確實作
Parameter.IStateManager.SaveViewState 明確實作
Parameter.IStateManager.TrackViewState 明確實作
Parameter.IsTrackingViewState 屬性
Parameter.LoadViewState 方法
Parameter.Name 屬性
Parameter.OnParameterChanged 方法
Parameter.Parameter 建構函式
Parameter.SaveViewState 方法
Parameter.SetDirty 方法
Parameter.Size 屬性
Parameter.ToString 方法
Parameter.TrackViewState 方法
Parameter.Type 屬性
Parameter.ViewState 屬性
ParameterAttribute 類別
ParameterAttribute 類別, 所有成員
ParameterAttribute 類別, 建構函式

URL(U): ms-help://MS.VSCC.v90/MS.MSDNQTR.v90.cht/wd_adonet/html/f21e6aba-b76d-46ad-a83e-2ad8e0af1e12.htm

全部摺疊 程式碼: 全部顯示

DataAdapter 參數 (ADO.NET)

請參閱 傳送意見

使用 SqlClient 參數

下列範例將示範如何建立 `SqlDataAdapter` 並將 `MissingSchemaAction` 設定為 `AddWithKey`, 以便從資料庫中擷取其他結構描述資訊。 `SelectCommand`、`InsertCommand`、`UpdateCommand` 和 `DeleteCommand` 屬性已設定而且其對應的 `SqlParameter` 物件已加入至 `Parameters` 集合。此方法會傳回 `SqlDataAdapter` 物件。

Visual Basic

```
Public Function CreateSqlDataAdapter( _
    ByVal connection As SqlConnection) As SqlDataAdapter

    Dim adapter As SqlDataAdapter = New SqlDataAdapter
    adapter.MissingSchemaAction = MissingSchemaAction.AddWithKey

    ' Create the commands.
    adapter.SelectCommand = New SqlCommand( _
        "SELECT CustomerID, CompanyName FROM CUSTOMERS", connection)
    adapter.InsertCommand = New SqlCommand( _
        "INSERT INTO Customers (CustomerID, CompanyName) " & _
        "VALUES (@CustomerID, @CompanyName)", connection)
    adapter.UpdateCommand = New SqlCommand( _
        "UPDATE Customers SET CustomerID = @CustomerID, CompanyName = " & _
        "@CompanyName WHERE CustomerID = @oldCustomerID", connection)
    adapter.DeleteCommand = New SqlCommand( _
        "DELETE FROM Customers WHERE CustomerID = @CustomerID", connection)

    ' Create the parameters.
    adapter.InsertCommand.Parameters.Add("@CustomerID", _
        SqlDbType.Int, 5, "CustomerID")
```

索引結果 - Parameter 物件 - 找到 2 個主題

標題	位置
DataAdapter 參數 (ADO.NET)	Accessing Data with ADO.NET
設定參數 (ADO.NET)	Accessing Data with ADO.NET

MS Help : SqlDbType 列舉型別

索引

篩選 (I):

Visual Basic

尋找 (I):

SqlDbType

SqlDbType 列舉型別

SqlDbType 屬性

SQL Debugging 類別

方法

所有成員

建構函式

關於 SQLDebugging 類別

SQLDebugging.SQLDebugging 建構函式

SqlDecimal 結構

方法

所有成員

建構函式

關於 SqlDecimal 結構

屬性

欄位

SqlDecimal.Abs 方法

SqlDecimal.Add 方法

SqlDecimal.AdjustScale 方法

SqlDecimal.BinData 屬性

SqlDecimal.Ceiling 方法

SqlDecimal.CompareTo 方法

SqlDecimal.ConvertToPrecScale 方法

SqlDecimal.Data 屬性

SqlDecimal.Divide 方法

SqlDecimal.Equals 方法

SqlDecimal.Floor 方法

SqlDecimal.GetHashCode 方法

SqlDecimal.GetKsdType 方法

SqlDecimal.GreaterThan 方法

SqlDecimal.GreaterThanOrEqual 方法

SqlDecimal.IsNull 屬性

SqlDecimal.IsPositive 屬性

SqlDecimal.LessThan 方法

SqlDecimal.LessThanOrEqual 方法

SqlDecimal.MaxPrecision 欄位

SqlDecimal.MaxScale 欄位

SqlDecimal.MaxValue 欄位

SqlDecimal.MinValue 欄位

SqlDecimal.Multiply 方法

SqlDecimal.NotEquals 方法

SqlDecimal.Null 欄位

SqlDecimal.op_Addition 方法

SqlDecimal.op_Division 方法

SqlDecimal.op_Equality 方法

SqlDecimal.op_Explicit 方法

SqlDecimal.op_GreaterThan 方法

SqlDecimal.op_GreaterThanOrEqual 方法

內容 索引 變更說明

SqlDbType 列舉型別

URL: ms-help://MS.VSCC.v80/MS.MSDN.v80/MS.NETDEVFX.v20.cht/cpref4/html/T_System_Data_SqlDbType.htm

.NET Framework 類別庫

SqlDbType 列舉型別

請參閱

全部摺疊 語言篩選設定: 全部顯示

成員

成員名稱	說明
 BigInt	Int64 。64 位元帶正負號的整數 (Signed Integer)。
 Binary	型別 Byte 的 Array 。二進位資料的固定長度資料流，範圍在 1 和 8,000 位元組之間。
 Bit	Boolean 。不帶正負號的數值，這個值可以是 0、1 或 Null 參照 (即 Visual Basic 中的 Nothing)。
 Char	String 。非 Unicode 字元的固定長度資料流，範圍在 1 到 8,000 個字元之間。
 DateTime	DateTime 。日期和時間的資料，值範圍從 1753 年 1 月 1 日到 9999 年 12 月 31 日，正確率為 3.33 毫秒。
 Decimal	Decimal 。固定精確度和小數點位數數值，介於 $-10^{38} - 1$ 和 $10^{38} - 1$ 之間。
 Float	Double 。浮點數，範圍為 $-1.79E + 308$ 到 $1.79E + 308$ 。
 Image	型別 Byte 的 Array 。二進位資料的可變長度資料流，範圍從 0 到 $2^{31} - 1$ (或 2,147,483,647) 個位元組。
 Int	Int32 。32 位元帶正負號的整數 (Signed Integer)。
 Money	Decimal 。貨幣值，範圍從 -2^{63} (或 -922,337,203,685,477.5808) 到 $2^{63} - 1$ (或 +922,337,203,685,477.5807)，正確率為貨幣單位的千分之一。
 NChar	String 。Unicode 字元的固定長度資料流，範圍在 1 到 4,000 個字元之間。
 NText	String 。Unicode 資料的可變長度資料流，具有 $2^{30} - 1$ (或 1,073,741,823) 個字元的最大長度。
 NVarChar	String 。Unicode 字元的可變長度資料流，範圍在 1 到 4,000 個字元之間。如果字串大於 4,000 個字元，則隱含轉換會失敗。當使用大於 4,000 個字元的字串時，明確設定物件。
 Real	Single 。浮點數，範圍為 $-3.40E + 38$ 到 $3.40E + 38$ 。
 SmallDateTime	DateTime 。日期和時間資料，值範圍從 1900 年 1 月 1 日到 2079 年 6 月 6 日，正確率為 1 分鐘。

SQL Injection – 使用黑名單方式處理

■ 凡遇黑名單字串, 取代成空白

```
String name = Request["itemname"];  
name = name.Replace ("or", "");
```

■ 駭客更高竿的鑽洞語法

```
itemname = chr(111)+chr(114) + .....
```

```
itemname = Convert.ToChar(111).ToString()  
+ Convert.ToChar(114).ToString()+ .....
```

SQL Injection – 使用黑名單方式處理

■ 凡遇黑名單字串, 取代成空白

```
String name = Request["itemname"];  
name = name.Replace ("or", "");
```

■ 駭客更高竿的鑽洞語法

```
itemname = " oorr 1=1 " + .....
```

SQL 語法無法參數化的部分的安全寫法

```
select * from items where  
itemname = @Itemname And owner=@owner  
order by @orderbyoption
```

實務上沒有這樣的 SQL 語法

SQL 語法無法參數化的部分的安全寫法

```
int iOrderBy = Convert.ToInt32(Request["Orderby"]);  
if ( iOrderBy == 1)  
    strOrderBy = " order by vendor "; // 固定字串  
else  
    strOrderBy = " order by shipdate,vendor ";
```

```
strSQL = "select * from items where itemname =  
        @Itemname And owner=@owner " ;  
strSQL += strOrderBy ;
```

網站程式安全撰寫基本樣式：N_To_S Pattern

- 由網頁取值的方式儘可能使用數值型態

```
int iOrderBy= Convert.ToInt32(Request["Orderby"]);
```

若有攻擊字串就會發生異常例外(Exception)

- 依據由網頁取得的數值 > 在程式中對應固定常數字串

1 > strOrderBy = "order by vendor"

2 > strOrderBy = "order by shipdate, vendor"

3 > strOrderBy = "order by custom, orderdate"

- 再進行後續的程式處理

程式碼修復說明

Cross-Site Scripting (XSS)



Cross-site scripting (XSS) 常見類型

- 攻擊語法直接寫在超連結或引用的參數欄位
 - 攻擊語法寫入在資料庫的字串欄位中
 - 攻擊語法寫入在竄改的網頁
- … 因為可以撰寫程式所以 **XSS** 變化無窮
- … 不同的 **XSS** 攻擊法要用不同的防禦法

XSS 攻擊案例：總統府網站事件

字級設定：

總統府網站被駭？ 專家：嵌入連結非遭 入侵

 NOWnews
今日新聞

更新日期:2009/08/28 01:47

總統府網站27日晚間傳出疑似遭到駭客入侵，打開網路上連結的總統府網頁，竟然出現當下最流行的韓國舞曲「sorry sorry」，馬總統的頭還被移植在畫面上，看起來就像是馬總統在跳舞一樣。

這段影片是之前網友爲了諷刺政府救災不力的惡作劇，把馬總統及劉兆玄院長等官員人頭接上影片，大跳sorry舞，27日晚間在網路上許多網友奔相走告，連結到疑似總統府網站，「欣賞」這個被入侵的畫面。

不過，資訊專家看過後表示，這可能只是網友利用某種嵌入法進行的連結，不是真的入侵到總統府網站，所以一般網友從IE瀏覽器進入總統府網站，是看不到這段連結畫面。（新聞來源：東森新聞）

本則新聞由NOWnews提供 2009/08/28

XSS 攻擊案例：總統府網站事件

2009-08-27 PM 22:09 | [網站導覽](#) | [兒童版](#) | [English](#) |

 **中華民國總統府** 
Office of the President, Republic of China (Taiwan) 

[回首頁](#) | [國是論壇](#) | [影音照片](#) | [服務信箱](#) | [網站檢索](#)

- [總統專欄](#)
- [副總統專欄](#)
- [新聞稿](#)
- [中華民國簡介](#)
- [總統府組織](#)
- [總統府公報](#)
- [法令查詢](#)
- [公布欄](#)
- [便民服務](#)
- [導覽與藝文](#)

總統府新聞稿



XSS 攻擊語法直接寫在引用的參數欄位

http://www.president.gov.tw/php-bin/dore2/list.php4?issueDate=&issueYY=&issueMM=&issueDD=&title=%3E%22+%3Ciframe+src%3Dhttp://www.youtube.com/watch_popup?v%3DTdFTeWHQ3CA%3E+%3Ci&content=&_section=3&_pieceLen=50&_orderBy=issueDate,rid&_desc=1

配合電子郵件
或部落格網頁
撰寫
超連結 XSS 攻擊

寄件者: 聯強國際
日期: 2007年12月26日 上午 03:26
收件者: free.willy@msa.hinet.net
主旨: 【聯強EMBA】觀念改變是突破現狀的關鍵

(本文取材自「聯強EMBA」，為聯強國際集團內部管理課程主題)

更多「聯強EMBA」：

- [見詞生義](#)
- [公車理論](#)
- [老鳥，也會墜機](#)

- 聯強國際保留本文所有著作相關權利
- 知識與經驗分享是最佳回饋社會方式，歡迎轉載，惟不得作為商業用途
- 如欲查詢更多相關內容，請至聯強 e 城市「[聯強EMBA](#)」主題

URL 連結 XSS 攻擊語法 + 搭配 Google 搜尋



XSS 修補漏洞的安全撰寫方式

- **XSS 攻擊語法致命傷** > 要撰寫程式所以字元數較多
- 執行前要加檢查程序：
 - (1) 合理長度值檢查 (白名單模式) On Server Side
例：**title** 字串長度是否正常長度，例如 `title.length() < 20`
 - (2) 不合理值檢查 (黑名單模式) On Server Side
例如：**iframe**、`Convert.ToChar(`

謹記在心：瀏覽器前端的驗證函式防駭功效不足

The screenshot shows the Mozilla Firefox browser interface. The 'Tools' menu is open, and the 'Tamper Data' option is highlighted with a red box. A 'Tamper PopUp' window is open, displaying the request headers and post parameters for the URL `http://localhost/CommerceAD/Login.aspx`. The 'Post Parameter Name' and 'Post Parameter Value' columns are circled in red.

Request Header Name	Request Header Value	Post Parameter Name	Post Parameter Value
Host	localhost	__VIEWSTATE	dDwzNTk1MDE1MjU7O2w8U
User-Agent	Mozilla/5.0 (Windows;	email	test%40test.com
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8	password	test
Accept-Language	zh-tw,en-us;q=0.7,en;q=0.3	LoginBtn.x	66
Accept-Encoding	gzip,deflate	LoginBtn.y	12
Accept-Charset	Big5,utf-8;q=0.7,*;q=0.3		
Keep-Alive	300		
Connection	keep-alive		
Referer	http://localhost/CommerceAD/images/		

The browser window shows the 'ASP.NET COMMERCIAL STARTER KIT' and a 'Sign Into Your Account' form with fields for 'Email' (test@test.com) and 'Password' (masked with dots).

Client Side Validation
或 Client Side 限制資料長度
只能防呆，不能防駭客竄改資料！

Use .Net Server 端的驗證控制項檢核資料

- **RequiredFieldValidator**
- **RangeValidator**
- **RegularExpressionValidator**
- **CompareValidator**
- **CustomValidator**
- **ValidationSummary**

ASP.NET 快速入門教學課程

驗證控制項

- ▣ [RequiredFieldValidator](#)
- ▣ [RangeValidator](#)
- ▣ [RegularExpressionValidator](#)
- ▣ [CompareValidator](#)
- ▣ [CustomValidator](#)
- ▣ [ValidationSummary](#)

◀ [回到 ASP.NET 首頁](#)

RangeValidator

RangeValidator 控制項會測試輸入值是否在指定的範圍之內。**RangeValidator** 使用三個主要有效範圍的最小值，而 **MaximumValue** 則是定義有效範圍的最大值。這些常數是儲存為字串。

下列範例說明使用 **RangeValidator** 控制項。

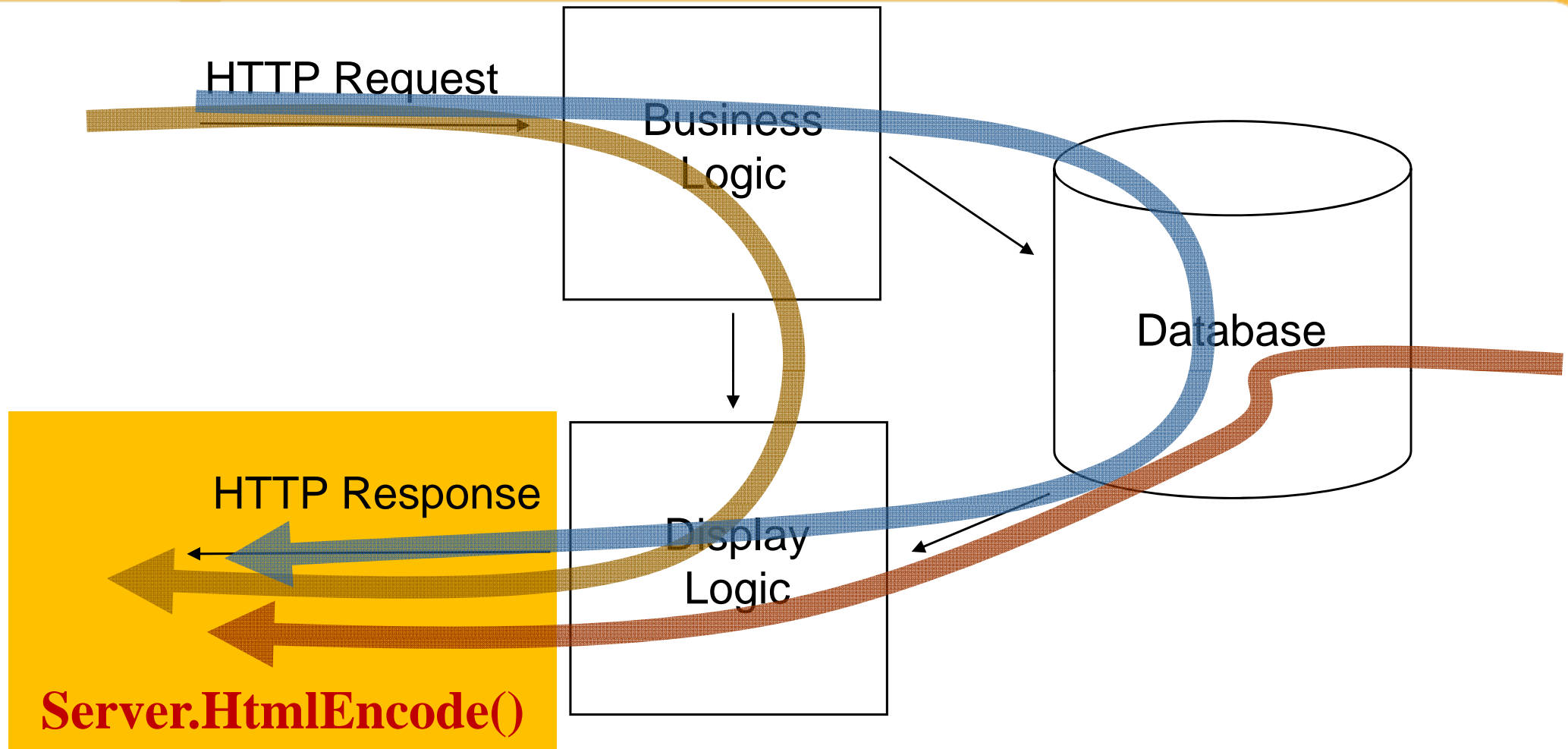
C# RangeValidator1.aspx

執行

檢視原始檔

如需 Web Form 驗證的詳細說明，請參閱這個快速入門教學課程的[驗證表單輸入控制項](#)章節。

顯示資料到網頁的 XSS 防禦方式



駭客輸入：

```
<script> alert("xss"); </script>
```

Encoding Mapping

<	<
>	>
"	"
'	'
&	&

Server.HtmlEncode() 之後

```
&lt;script&gt;  
alert(&quot;xss&quot;);  
&lt;/script&gt;
```

瀏覽器僅會顯示文字在網頁上

```
<script> alert("xss"); </script>
```


PHP 使用 htmlspecialchars()

Encoding Mapping

<	<
>	>
"	"
'	'
&	&

駭客輸入:

```
<script> alert("xss"); </script>
```

htmlspecialchars()之後

```
&lt;script&gt;  
alert(&quot;xss&quot;);  
&lt;/script&gt;
```

瀏覽器僅會顯示文字在網頁上

```
<script> alert("xss"); </script>
```

.Net 對於 XSS 攻擊不安全的元件

The screenshot shows the Visual Studio IDE with a web form titled "Default.aspx". The toolbox on the left lists various controls. The form contains the following elements:

- Two text boxes labeled "b1" and "bNoXSS".
- A large cyan rectangular area.
- A "HyperLink" control.
- A "Button" control with a red 'x' icon.
- A "RadioButton" control labeled "[RB1]".
- A "CheckBox" control labeled "[CheckBox1]".
- A "LinkButton" control.
- A "Button" control.
- A "資料繫結" (Data Binding) label.
- A "資料繫結" (Data Binding) label.
- A table with 3 columns: "CustomerID", "CompanyName", and "ContactName". Each cell contains the text "資料繫結".

The screenshot shows a web browser displaying the Carrefour website. The page contains the Carrefour logo and the Chinese characters "家樂福" (Carrefour). The URL is "www.carrefour.com.tw". A JavaScript payload is injected into the page, highlighted in pink and cyan:

```
<script>document.write('<img src =  
"http://www.iibrand.com/customer/news-  
images/20080417/1534170420083410_1.gif">')  
</script>
```

```
<script>document.write('<img src  
= "http://www.iibrand.com/customer/news-  
images/20080417/1534170420083410_1.gif">')
```

The page also shows a "我的最愛" (My Favorites) button and a search bar. The Carrefour logo and Chinese characters "家樂福" are repeated on the page. The URL "www.carrefour.com.tw" is also visible. A "完成" (Done) button is visible at the bottom right.

XSS 問題程式：Response.write ()

■ 修補漏洞的方式分析

輸出到網頁的 XSS 問題，加 **Server.HtmlEncode()** 安全防護

```
88  
89         }  
90  
91     Response.Write(strReturn);  
92     Response.End();  
93     }  
94 }  
95 }
```

■ 修補漏洞的安全撰寫方式

```
Response.Write( Server.HtmlEncode(strReturn) );
```

XSS 問題類型：

System.Web.UI.WebControls.BaseDataList.set_DataSource()

■ 修補漏洞的方式分析

- **DataList** 預設套用的樣版物件是 **Label** 有 **XSS** 問題
樣板中無法撰寫 **Server.HtmlEncode()**
- 所以修改方式之一是手動改樣版預設元件,套用安全的 **TextBox** 元件

為了產生與 **Label** 相同呈現效果,設定屬性
BorderStyle="None" ReadOnly="True"

Sample Code

```
<asp:DataList ID="DataList1" runat="server" DataKeyField="CustomerID" DataSourceID="AccessDataSource1" BorderWidth="10px" CellPadding="1" GridLines="Both">
  <ItemTemplate>
    CustomerID:
    <asp:Label ID="CustomerIDLabel" runat="server" Text='<%# Eval("CustomerID") %>'></asp:Label>
    <br />

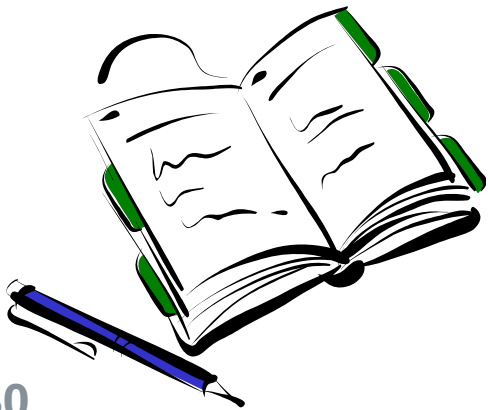
    CompanyName:
    <asp:TextBox ID="CompanyNameLabel" BorderStyle="None" ReadOnly="True" runat="server"
      Text='<%# Eval("CompanyName") %>'></asp:TextBox>

    <br />
    .....

  </ItemTemplate>
</asp:DataList>
```

程式碼修復說明

HTTP Response Splitting



Header Manipulation 問題

- Addition of unvalidated data to the HTTP header
 - Could result in XSS vulnerability
 - Browser cache poisoning
 - Server cache poisoning

- Consider :

```
<%  
response.sendRedirect("/region.jsp?  
    regionCode="+  
request.getParameter("regionCode"));  
%>
```

Header Manipulation 問題

- An HTTP response would look like :

```
HTTP/1.1 302 Moved Temporarily
Date: Wed, 24 Dec 2003 12:53:28 GMT
Location: http://120.14.10.16/region.jsp?regionCode=us
Server: Apache/2.0.49 Fri Jan 2 13:15:34 PDT
Content-Type: text/html
Set-Cookie:
JSESSIONID=alkjwerf345sdf0sd9f8; path=/
Connection: Close

<html><head><title>302 Moved Temporarily</title></head>
<body bgcolor="#FFFFFF">
<p>This document you requested has moved temporarily.</p>
</body></html>
```


Header Manipulation 問題

- Since input for region is not validated
 - Attacker could supply

```
/region.jsp?regionCode=us%0d%0aContent-  
Length:%20%0d%0a%0d%0aHTTP/1.1%20200%20OK%0d%0aContent-  
Type:%20text/html%0d%0aContent-  
Length:%2019%0d%0a%0d%0a<html>Got you hacked  
mate !</html>
```

Header Manipulation 問題

- Since input for region is not validated
 - Attacker could supply

```
HTTP/1.1 302 Moved Temporarily
Date: Wed, 20 Jan 2003 15:26:41 GMT
Location: 120.141.100.100.jsp?regionCode=us
Content-Length: 0
```

1st Response

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 19
```

1 Request, 2 Responses
(Response Splitting)

Hacker provided data

```
<html>Got you hacked mate !</html>
Server: Apache/2.0.49
2003 271009 with
Content-Type: text/html
Set-Cookie:
JSESSIONID=123wertyu567345; path=/
Connection: Close
```

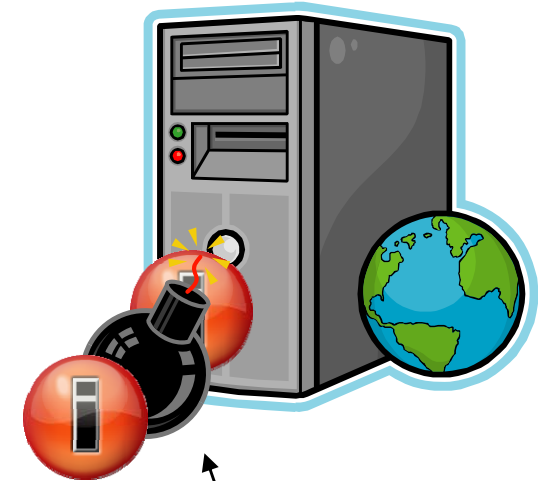
2nd Response
(Controlled by Hacker)

.....

Header Manipulation 問題



Normal User



Proxy Server



Proxy server see 2 requests and 3 responses, dropped the last response and cached the hacker controlled response as the valid response for 2nd request

The 2nd request is due to HTTP Response Splitting and is controlled by hacker

Hacker



Hacker send 2 requests to HTTP Server

Good news for HTTP Response Splitting

■ ASP.NET 2.0

- By default, .NET 2.0 will return 500 and throw exception when there is "\r\n" in methods that involve HTTP response headers
- You can set "enableHeaderChecking" to false in web.config in order to disable this protection

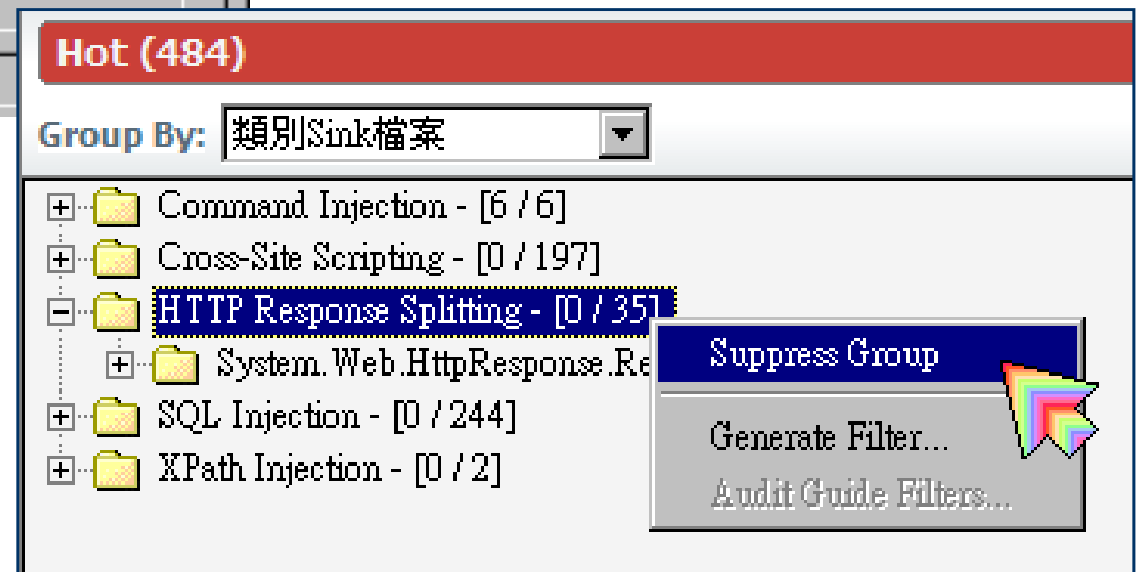
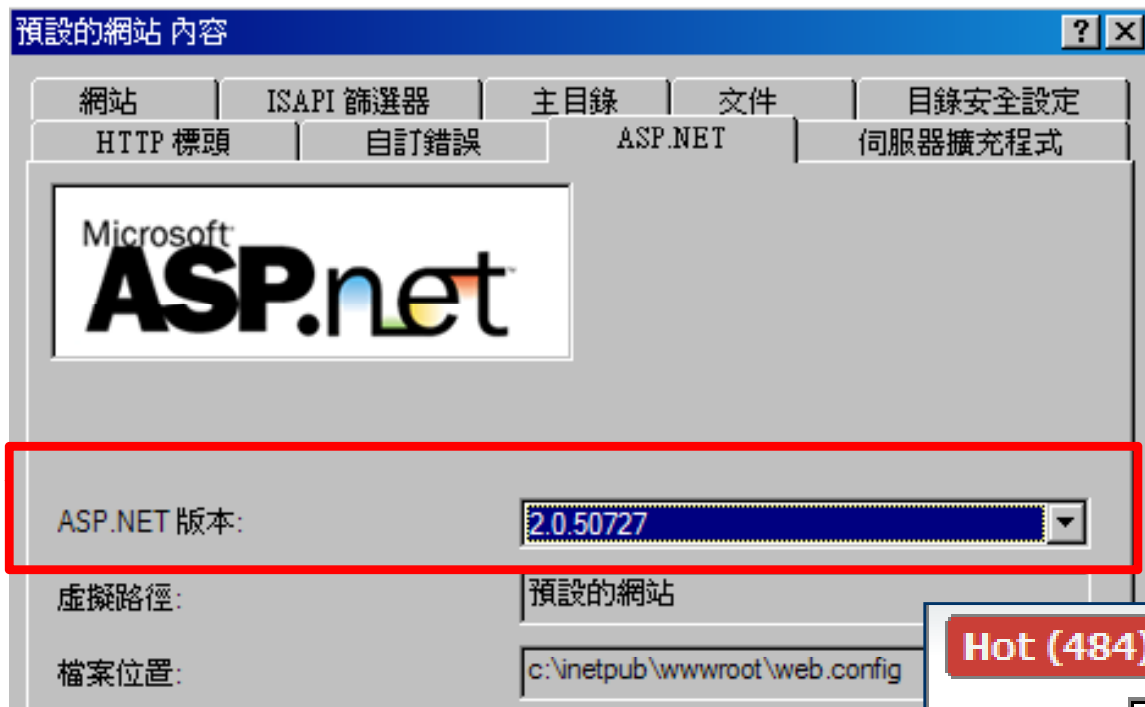
■ ASP.NET 1.1

- Please apply ASP.NET SP1
- To disable, `<httpWebRequest useUnsafeHeaderParsing="true" />`

■ Tomcat 5.0

- Tomcat will escape "\r\n" you try to add extra HTTP header
- Tomcat 4.x is vulnerable

HTTP Response Splitting



程式碼修復說明

Command Injection



Command Injection 安全漏洞問題

```
string fileLocation = Request["filename"];  
  
Process test = new Process();  
test.StartInfo.FileName = fileLocation;  
test.Start();  
  
Request["filename"] = del *.* /q
```

Command Injection 安全漏洞問題

[V] 安全的寫法 (使用白名單語法)

```
int icmdfile = Convert.ToInt32(Request["filename"]);  
string fileLocation = "List.exe"  
  
if (icmdfile == 1) {fileLocation = "Query.exe";}  
if (icmdfile == 2) {fileLocation = "Report.exe";}  
...  
Process test = new Process();  
test.StartInfo.FileName = fileLocation;  
test.Start();
```


程式碼修復說明

Path Manipulation

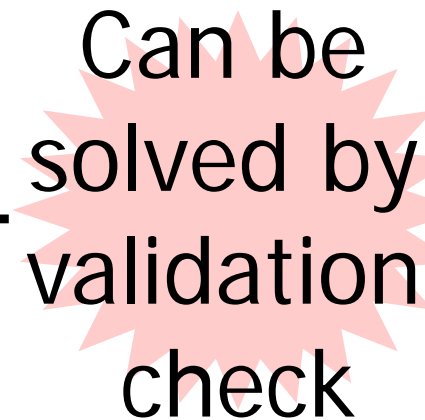


- **Hacker can control which file to be opened**

- **Filename:**

`"c:\data\" + filename`
`filename` → `../boot.ini`

Can be
solved by
validation
check



- **Filepath:**

`path` + `"myprog.dll"`

`Path` can be `"c:\tmp\hacker_upload\"`

Path Manipulation

Module1.vb

Module1

(宣告)

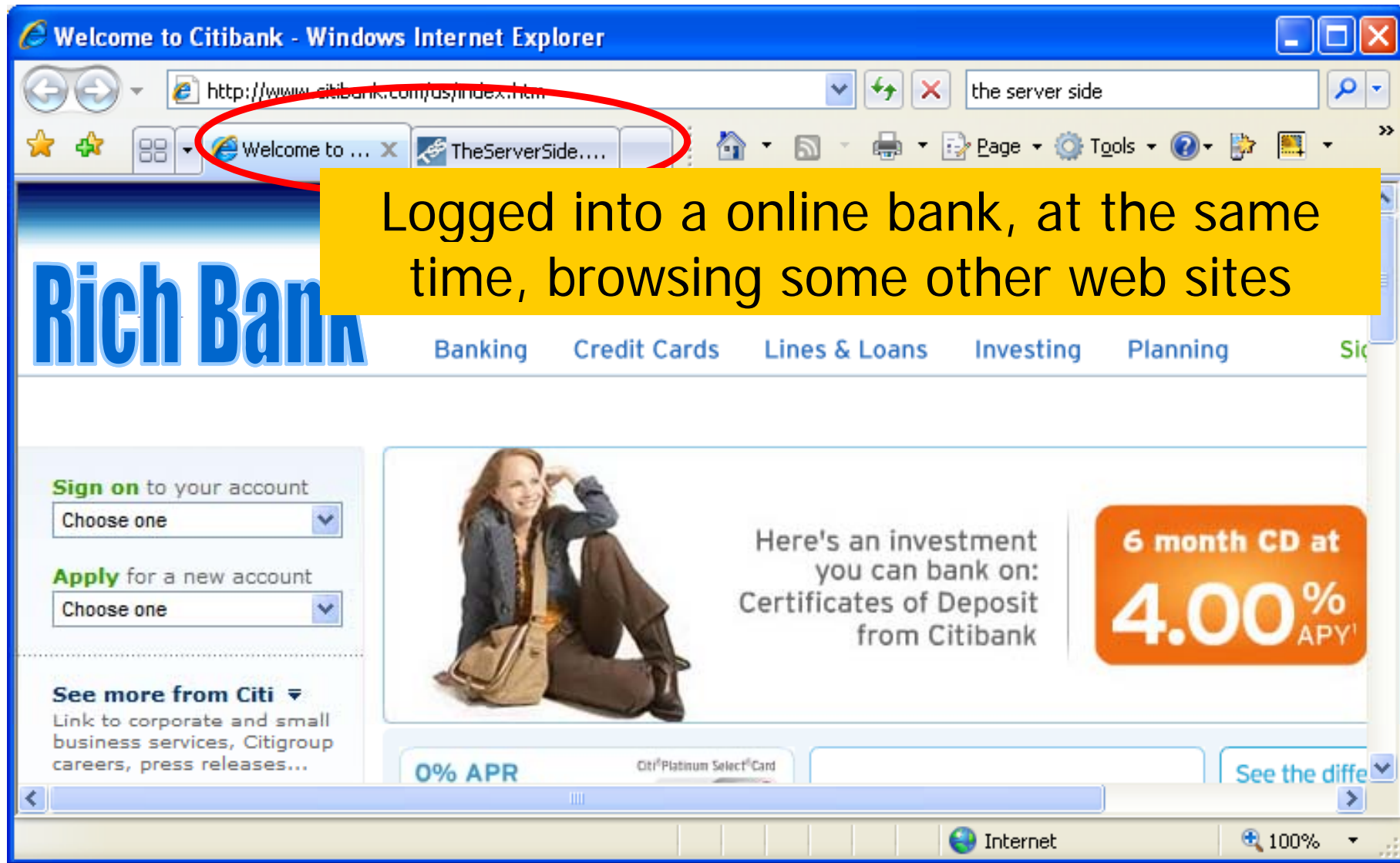
```
19 Function secure_PathManipulation(ByVal strPathFile As String) As String
20
21 '請使用 VB.Net System.Text.RegularExpressions.Regex 物件
22 '或其他安全函式()
23 '在此處撰寫您的程式路徑安全的白名單字元檢查邏輯程式。
24
25 Dim whitelist As String = "[^0-9a-zA-Z\_\.\-]"
26 Dim pattern As Regex = New Regex(whitelist)
27
28 If pattern.IsMatch(strPathFile) Then
29     If strPathFile.IndexOf("\192.168.100.28") > -1 Or strPathFile.IndexOf("\tempPDF\") > -1 Or strPathFile.IndexOf("\bPDF\") > -1 Then
30         Return strPathFile
31     Else
32         Return ""
33     End If
34 Else
35     Return ""
36 End If
37
38 Return strPathFile
39 End Function
40
```

程式碼修復說明

Cross Site Request Forgery (CSRF)



Cross-site request forgery (CSRF)



Cross-site request forgery

The online bank



The bank sends the money since this is a valid request

Hacker Website



Inside in the page, there is a image

```

```

The browser will send the request to the bank with a valid cookie.



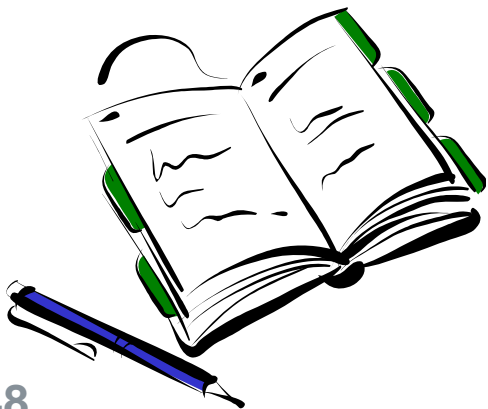
How to fix : 使用需要人工識別的機制及驗證碼

歡迎您給我寶貴意見，謝謝！(有 * 為必填欄位) [本署電子信箱信件處理流程圖](#) | [說明](#)

*您的真實姓名：	<input type="text" value="請填真實姓名"/>
*您的聯絡電話：	<input type="text" value="請填聯絡電話"/> 如：0912-345-678
*您的電子信箱：	<input type="text" value="請填電子信箱"/> 如：abc@yahoo.com.tw
您的聯絡地址：	<input type="text" value="請填聯絡地址"/>
*您的意見內容：	請勿輸入[<]或[>]符號 <input type="text" value="請填您的意見內容"/>
*驗證碼：	 <input type="text" value="請填驗證碼"/> 驗證碼有大小寫之分，輸入時請注意。若圖片不清楚，請點擊圖片。
來源IP：	00.250.147.33
<input type="button" value="寄出"/> <input type="button" value="重填"/>	

程式碼修復說明

Password Management



Password Management

- **Don't hard code password in source code**
 - You can't change password in the future
 - Pretty easy to decompile and get the password in binary code
- **Don't stored plain text password in config file or registry key**
- **DO: store obfuscated password in config file**

Password Management: Sample Code

```
<parameter>
  <name>url</name>
  <value>jdbc:.....</value>
</parameter>
<parameter>
  <name>driverClassName</name>
  <value>com.oracle.jdbcDriver</value>
</parameter>
<parameter>
  <name>username</name>
  <value> MGLQAbY6ADV49yWAQnaTztr742gGO1x= </value>
</parameter>
<parameter>
  <name>password</name>
  <value> DV49MGLQyWAGAbY6O1qQnaTztr742g= </value>
</parameter>
```

程式碼修復說明

Race Conditions



Race Condition : Multi-Thread

Question: What's wrong with this code?

```
public class GuestBook extends HttpServlet
{
    public static String name;
    protected void doPost (HttpServletRequest req,
        HttpServletResponse res)
    {
        name = req.getParameter("name");
        ...
        out.println(name + ", thanks for visiting!");
    }
}
```

Race Condition

Answer: Hackers will hack...

Retrieve others customer information because value is explicit to hacker

Thread 1	Thread 2	Value in "name"
name = "Dick"		"Dick"
	name = "Jane"	"Dick" -> "Jane"
	"Jane, Thanks for visiting"	"Jane"
"Jane, Thanks for visiting"		

Race Condition

Answer: don't use class variable , using local variable

```
public class GuestBook extends HttpServlet {  
    String name;  
    protected void doPost (HttpServletRequest req,  
        HttpServletResponse res){  
        String name = req.getParameter("name");  
        ...  
        out.println(name + ", thanks for visiting!");  
    }  
}
```

Race Condition (File)

Question: What's wrong with this code?

```
public class Toctou extends HttpServlet
{
    public void doGet(HttpServletRequest req, HttpServletResponse
        res) throws ServletException, IOException
    {
        . . .
        File f = new File("/tmp/file.txt");
        FileWriter fw = new FileWriter(f);
        fw.write(msg, 0, msg.length());
        fw.close();
        f.setReadOnly();
        . . .
    }
}
```

Two users access the same page at the same time. We will still use the same file name

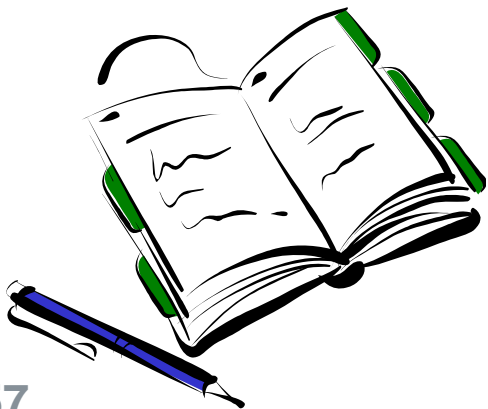
Race Condition

Solution

```
public class ToCTou extends HttpServlet
{
    public void doGet(HttpServletRequest req, HttpServletResponse
        res) throws ServletException, IOException
    {
        . . .
        File f = File.createTempFile("aaa", ".tmp");
        FileWriter fw = new FileWriter(f);
        fw.write(msg, 0, msg.length());
        fw.close();
        f.setReadOnly();
        . . .
    }
}
```


程式碼修復說明

Error Handling



Poor Error Handling

Question: What's wrong with this code?

```
DataSet dataSet = null;
```

```
try {
```

```
    dataSet = doExchange();
```

```
} catch ( Exception e) {
```

```
    e.printStackTrace();
```

```
}
```

Overly broad Catch

System Information Leak
(or Empty Catch Block if empty)

Poor Error Handling

Dump Exception is not exception handling

```
DataSet dataSet = null;
Boolean done = false;
for(int i=0; i<3; i++) {
    try {
        dataSet = doExchange();
        done = true;
        break;
    } catch ( Exception e) {
        log.warn("Do Exchange Failed: ...");
    }
}


if ( !done ) return false;
for(int i=0; i<dataSet.size(); i++) {
    DataRow row = dataSet.getRow(i);
}
```

Retry doExchange() 3 times

Systematic logging framework

Poor Error Handling

- Don't send any part of the Exception to HTML

網址(D)  http://localhost/commerceAD/OrderList.aspx

"/CommerceAD' 應用程式中發生伺服器錯誤。

遺漏字元字串' GROUP BY x.CustomerID, x.OrderID, x.ShipDate, y.FullName, z.ModelName' 後面的引號。

錯誤: 在執行日期 Web 頁上的某個字元字串中發現引號。請仔細檢查 Web 頁中各種標記的引號。以下是在字串中使用的引號。

內含錯誤字串: System.Data.SqlClient.SqlException: 遺漏字元字串' GROUP BY x.CustomerID, x.OrderID, x.OrderDate, x.ShipDate, y.FullName, z.ModelName' 後面的引號。

詳細錯誤資訊:

只有在偵錯模式編譯時，才可以顯示產生此未處理例外狀況的原始程式碼。若要啓動，請依照

1. 將 "Debug=true" 指示詞加入產生錯誤的程式碼頂端。例如:

```
<%@ Page Language="C#" Debug="true" %>
```

或:

2. 將下列區段加入您應用程式的組態檔:

```
<configuration>  
  <system.web>  
    <compilation debug="true"/>  
  </system.web>  
</configuration>
```

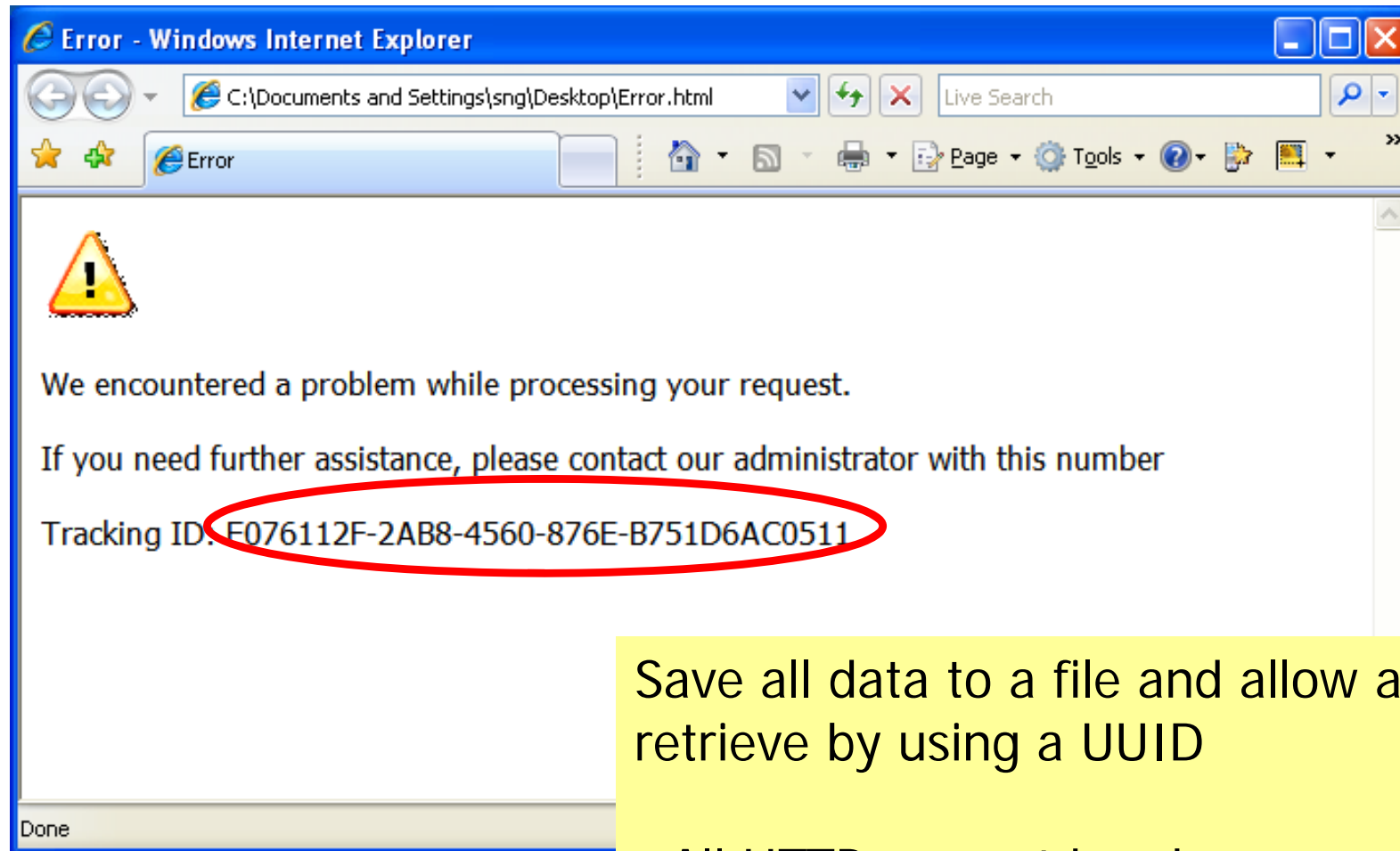
請注意: 第二種技巧會導致在偵錯模式下編譯指定應用程式中的所有檔案。第一種技巧只會造

重要資訊: 在偵錯模式下執行應用程式會過度耗用記憶體/效能。在實際執行部署之前, 應該

檢查 Web 頁:

```
[SqlException (0x80131904): 遺漏字元字串' GROUP BY x.CustomerID, x.OrderID, x.OrderDate, x.ShipDate, y.FullName, z.ModelName'  
System.Data.SqlClient.SqlConnection.OnError(SqlException exception, Boolean breakConnection) +857434  
System.Data.SqlClient.SqlInternalConnection.OnError(SqlException exception, Boolean breakConnection) +735046  
System.Data.SqlClient.TdsParser.ThrowExceptionAndWarning(TdsParserStateObject stateObj) +324  
System.Data.SqlClient.TdsParser.Run(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySt  
System.Data.SqlClient.SqlDataReader.ConsumeMetaData() +31  
System.Data.SqlClient.SqlDataReader.get_MetaData() +22  
System.Data.SqlClient.SqlCommand.FinishExecuteReader(SqlDataReader ds, RunBehavior runBehavior, String resetOptionsStrin
```

A better Error Page



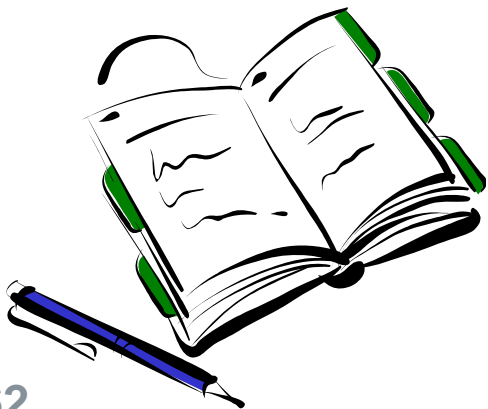
Save all data to a file and allow admin to retrieve by using a UUID

- All HTTP request headers
- Exception details

But beware of your disk space...

程式碼修復說明

Misconfiguration



Misconfiguration: Environment

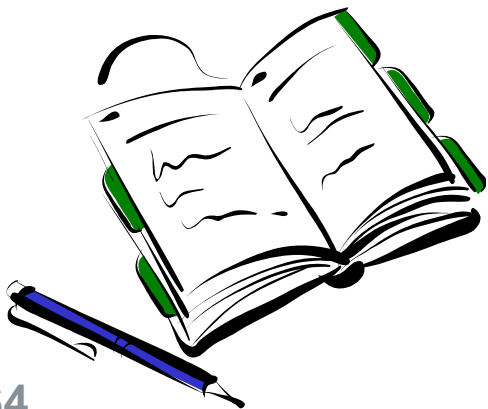
```
<%@ Page Language="C#" AutoEventWireup="true"  
    CodeFile="Default.aspx.cs" Inherits="_Default"  
    ValidateRequest="false" %>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0  
    Transitional//EN"  
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-  
    transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml" >  
<head runat="server">  
    <title>Untitled Page</title>  
</head>  
<body>  
    <form id="form1" runat="server">  
    <div>
```

程式碼修復說明

Value Shadowing



Value Shadowing 問題

此程式以不明確的方式存取伺服器變數，這可會使程式易受到攻擊。

HttpRequest類別提供透過程式從陣列存取表單中

QueryString、Form、Cookies 或ServerVariables 集合存取變數的能力

如 **Request[“myParam”]**) 。

當有一個以上名稱相同的變數時，.NET framework會傳回在集合以下列順序搜尋時，**第一個出現的變數值：QueryString、Form、Cookies**，然後**ServerVariables**。

因為QueryString依搜尋順序第一個出現，因此QueryString參數可以取代Form、cookie及伺服器變數的值。

同樣地，Form 值可以取代Cookies和ServerVariables集合中的變數，而Cookies集合的變數可取代ServerVariables的變數。

Value Shadowing 修改方式

■ 有安全漏洞問題的程式碼

```
37         if (!IsPostBack)
38         {
39             string strFunc=(Request["Func"]==null?"":Request["Func"]);
40             string strReturn = ""; //2009/04/21 將回傳格式統一
41
42             if (strFunc != "")
43             {
44                 //參數宣告宣告 //2009/04/23 收斂宣告參數，節省memory
45                 string strPara1;
```

■ 修補漏洞的安全撰寫方式

>> 使用明確的集合名稱存取

```
strFunc = (Request.Form["Func"] ==null? "" : Request.Form["Func"]);
```

```
strFunc = Request.Cookies ["Func"];
```

```
strFunc = Request.ServerVariables ["Func"];
```

```
strFunc = Request.QueryString ["Func"];
```

