

*Hashem(S)*

*ASP Security: Ways to Avoid Attack*

امنیت در ASP.NET

راههای مقابله با نفوذ در ASP.Net

نویسنده : *Hashem(S)*



**مقدمه:**

امروزه استفاده از صفحات ASP.NET یکی از وسایل تولید برنامه های کاربردی وب (Applittaction) است. ولی این صفحات دارای ضعفهای فراوان می باشد. امیدوارم این مقاله بتواند برای طراحان سایتها کمکی باشد تا صفحاتی با ضریب امنیت بالاتر طراحی شود. شاید افزایش امنیت برنامه های کاربردی وب چندان به چشم نیاید ولی شاید یکی از راههای نفوذ هکرها عزیز !!! ما می باشد. طراحی برنامه های کاربردی وب با ASP.NET آنچنان آسان نیست. این نوع طراحی امنیت و امکانات مخفی زیادی داره اما آیا ما با آنها می توانیم کار کنیم در ضمن این قابل ذکر است که هکرها همیشه یک قدم از ما جلوترند. یکی از راههای مقابله با آنها دانستن روش کار هکرهاست یعنی دانستن روشهای نفوذ حالا با بستن راه ها می شود تا حدی جلو آنها را گرفت. بررسی راهها شاخه های زیادی داره اما به طور کل می توان ۸ روش است.

**روش اول: Cross - Sit Scripting (XSS)**

این روش ایجاد اختلال در صفحات وب با ASP توسط Script های قابل اجرا است. XSS یکی از بیشترین روش حمله می باشد. یک هکر برای حمله XSS به راحتی با تزریق یک Script در میان صفحه وب از طریق قسمت ورودی های کاربران و کلیک دکمه تایید بدون چک کردن لازم می تواند صفحه شما را مورد حمله قرار دهد. موقعی که داده ها که همراه Script است در وب پویا ذخیره می شود این داده می تواند برای هر کاربر دیگر در مرورگر ویش بار شود. برای فهمیدن بهتر XSS مثالی می زنیم: در یک صفحه وب که از یک کار بر سوال میشود User Name خود را در کادری در صفحه وارد کند از طریق این کادر (شکل ۱) وقتیکه ما نام کاربر را وارد کنیم دکمه را کلیک می کنیم حال ما می توانیم به جای نام کاربر در کادر یک خط دستور Script وارد کنیم برای مثل: (شکل ۲)

```
<script> alert ("HELLO THERE"); </script>
```



شکل ۲- این یک رشته کاراکتری نیست. بلکه یک حمله با Script است.



شکل ۱- آماده وارد کردن یک رشته کاراکتری

در مورد زمانی که این Script را با زدن دکمه Click اجرا می شود و (شکل ۳) پدید می آید.



شکل ۳- بعد از کلیک Script در فرم ۲

این یک مثال بود اما Script های وجود دارد که خسارات زیادی میزند حتی بعضی از آنها می توانند مقدار Cookie های و یا اطلاعات کاربران را که از این سرور استفاده کرده اند حتی اگر از سایتها دیگر استفاده کرده باشند. خوشبختانه در ASP.NET V1.1 دارای سیستم می باشد که می تواند Script ها را شناسایی کند و پیغام (شکل ۴) را می دهد. این ساختار در سیستم درونی ASP.NET V1.1 تعبیه شده که



## Hashem(S)

بتواند جلوی نفوذ با Script را بگیرند روش آن بصورت این است که به کلمه کلیدی <script> در کلمه ورودی می گردد و در صورت یافتن آن پیغام می دهد. اما این سیستم آنچنان هم قوی نیست می گویند چرا؟ چون راههایی است که به راحتی می شود این سیستم را فریب داد مثلاً یکی از این راهها با اضافه کردن یک کاراکتر NULL (%00) قبل از Script در دستور <Script> برای مثال قبل می شود:

<%00script> alert ("HELLO THERE"); </script>

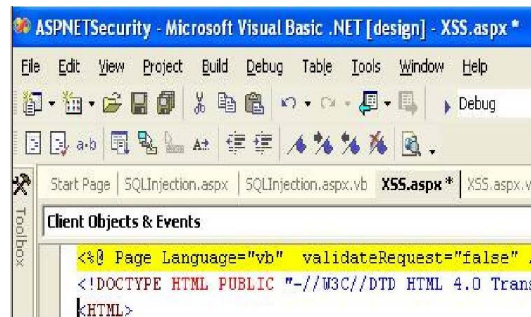
شکل ۴



پس با این سیستم هم می شود جلو نفوذ کامل را گرفت. یک روش خوب استفاده کردن سرور از روش (HTML ENCODE) است که کارش این است که ورودی ها را به رمز (کد) تبدیل می کند با رشته های HTML ENCODE می باشد. حال ورودی کاربر با استفاده از تابع زیر رمز گذاری می شود:

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) _
    Handles Button1.Click
    lblMessage.Text = Server.HtmlEncode(txtName.Text)
End Sub
```

خوب حالا اگر کاربر یک Script وارد کند HTML ENCODE ورودی را گرفته آن را به شکل رشته کارتری در آورده و در مرورگر نشان می دهد نه به صورت اجرایی شما اگر به هر دلیلی خواستید خاصیت محافظ در ASP.NET V1.1 را از کار ببندازید کافی است خاصیت ValidateRequest را به حالت False در آورید. (شکل ۵) در این حالت خاصیت محافظت را به صورت دستی غیر فعال می کنیم اما ما پیشنهاد این کار را نمی کنیم بالاخره شاید بتواند جلوی هکر های تازه کار بگیرد گفتیم شاید فقط شاید!!!!!!

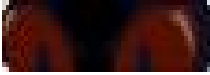


شکل ۵

## روش دوم: SQL INJECTION

دومین روش بسیار ساده که با یک اکسپلویت معروف با نام SQL می باشد. جالب این که کمتر هکر ها از آن استفاده می کنند بهتر بگویم این روش فقط در بین هکرهای با سابقه رواج دارد تا غیر حرفه ایها. جالبتر که تعداد افرادی که این نقطه ضعف را می دانند بسیار کم هستند و حتی درباره آن نشنیده اند. حالا بگذارید با یک مثال طرز کار با این روش که با وارد کردن یک SQL است کار می کند را توضیحی کوتاه بدهیم: فرض کنید یک صفحه داریم که برای Login (وصل شدن) باید Username و Password را وارد کنیم. در صفحات وب پویا یک تابع (Function) بسیار معمولی وجود دارد. بیشتر طراحان سایت (مخصوصاً تازه شروع به طراحی برنامه های بانک اطلاعاتی در ویها کرده اند) تابعی به صورت زیر می نویسند:

```
Private Sub btnLogin_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) _
    Handles btnLogin.Click
    SqlConnection1.Open()
    Dim str As String = "SELECT * FROM Users WHERE UserID=" & _
        & txtName.Text & " AND Password=" & _
        txtPassword.Text & ""
    Dim comm As New SqlCommand(str, SqlConnection1)
    Dim reader As SqlDataReader = comm.ExecuteReader()
    If Not reader.HasRows Then _
        Response.Write("Login failed. Please try again")
```



```

While reader.Read()
Response.Write("Hello " & reader("UserName"))
End While
End Sub

```

در این کد به سادگی اطلاعات وارد شده توسط کاربران در روی صفحه وب آشکار است و از فرمولهای رشته ها در SQL برای برنامه نویسی استفاده شده است برای گرفتن داده های کاربر.

باید در طراحی صفحات کمترین اطلاعات از کاربران در صفحه درج نشود مثلا در بعضی سایت ها Username کاربران مشخص است. بهترین راه کنترل طول ورودی که کاربر وارد می کند است تا مطمئن شویم ورودی طولانی تر از حد معمول نباشد. با وجود این کدها نامرتب خطر بزرگی ما رو تهدید می کند. وقتی نام کاربران مشخص باشد یعنی برای هکر ها نصف کار را انجام دادیم!! مطمئن باشید چیز هایی وجود دارد که آنها به جای پسورد وارد کنند و به بانک اطلاعات شما دست بیابند. برای مثل (شکل ۶)

در این شکل با وارد کردن چند کلمه کلیدی SQL می توان اطلاعات رو دستکاری کرد و بدون داشتن پسورد وارد شد. یکی دیگر از اکسپلویت هایی که هکرها از آن برای حمله استفاده می کنند SQL Union است. به این صورت کار می کند که به دنبال رشته کاراکتری وارد شده در Username یا پسورد و اجرا آن با زدن دکمه Click (شکل ۷) می توان اطلاعات مهم و زیادی در مورد سرور در یافت کرد.

کد SQL Union بصورت زیر می تواند باشد:

```

xyz' union select @@servername, @@servicename,
@@version -

```

یک روش بسیار خوب بروی فرمولبندی رشته های SQL استفاده از موضوع Parameters در SQL Command است. علت برتری این معبر ایجاد شده این است که ADO.NET نمی تواند عمل جانشینی (Substitution) را انجام دهد. در این معبر پارامترهای SQL Server همان جا عمل جانشینی و خطایابی و یا تایید رخ می دهد.



شکل ۶- با داشتن نام کاربر بدون داشتن پسورد وارد شوند.



شکل ۷- کاربر بجای پسورد در فرم Login یک رشته دستورات SQL برای آشکار کردن اطلاعات لازم برای حمله استفاده کرده.

در زیر این روش را برای Login پیاده سازی کرده ایم:

```

Private Sub btnLogin_Click(ByVal sender As System.Object,ByVal e As System.EventArgs) _
Handles btnsecureLogin.Click
SqlConnection1.Open()
Dim str As String = "SELECT * FROM Users WHERE " & _
"UserID=@userID AND Password=@password"
Dim comm As New SqlCommand(str, SqlConnection1)
comm.Parameters.Add("@userID", txtName.Text)
comm.Parameters.Add("@password", txtPassword.Text)
Dim reader As SqlDataReader = comm.ExecuteReader()
If Not reader.HasRows Then Response.Write("Login failed. Please try again")
While reader.Read()
Response.Write("Hello " & reader("UserName"))
End While
End Sub

```



### روش سوم: تایید ورودی های کاربر توسط خود برنامه نویسی وب

در این روش باید اطلاعاتی از پیش معین شده را در اختیار کاربران قرار دهیم تا کاربر فقط بتواند همانها را وارد و انتخاب کند. در این روش با استفاده از اسکریپت ها و **check box**ها در ورودی ها و محدود کردن گزینه های ورودی و قرار دادن مقدار پیش فرض در ورودی ها. اما این روش برای ورودی های قابل پیش بینی و محدود کاربرد دارد مثل: تاریخ تولد و نام کشور و نام شهر و .... .

در ورودی های که نمی توان از حالت قابل پیش بینی استفاده کرد هم در کلاینت و هم در سرور برای آن ورودی یک مقدار پیش فرض در کادر ورودی کلاینت و در بانک سرور قرار دهید.

### روش چهارم: استفاده از روش مخلوط کردن (Hahsing)پسوردهای ذخیره شده

یک روش خوب برای ذخیره پسود ها در بانک اطلاعاتی سرور استفاده از روش **Hashing (درهم)** است. **Hashig** دارای دو روش است:

**یک:** روش پردازش برای نگارش داده ها (**متن های بدون رمز**) برای این کار از یک رشته بایتی منحصر به فرد با طول ثابت استفاده می شود. به این رشته بایتی با طول ثابت **Hash** گویند.

**دوم:** روش ایستا در این روش تمام اطلاعات توسط یک متد رمز گذاری و ذخیره می شود. در مورد ذخیره پسوردها در بانک اطلاعات باید گفت این اطلاعات ذخیره شده ارزش **Hash** هر یک از پسورد ها است و برتری بالاتری برای ذخیره پهنای پسورد دارد. وقتیکه یک کاربر پسورد خود را برای **Login** شدن وارد می کند ارزش **Hash** پسورد محاسبه شده و سپس با ارزش **Hash** ذخیره شده مقایسه می شود. اما باز مشکلات فراوانی است اگر هرکرا به اطلاعات دست یابند درست است به پسوردهای اصلی دسترسی ندارد ولی هنوز می تواند مقدار آنها را تغییر دهد یا با ایجاد چند کاربر برای خود (**در حالت ایستا**) به فرمول رمز گذاری دست یابد.

تابع زیر استفاده می کند از الگوریتم عددی درهم سازی (**hash**) **SHA1** استفاده شده :

```
Public Function ComputeHashValue(ByVal data() As Byte) As Byte()
Dim hashAlg As SHA1 = SHA1.Create
Dim hashvalue() As Byte = hashAlg.ComputeHash(data)
Return hashvalue
End Function
```

میشود پسورد ها را با فراخوانی تابع بالا تبدیل به ارزش **Hash** کرد:

```
Dim hashValue() As Byte
hashValue = ComputeHashValue(Encoding.ASCII.GetBytes(txtPassword.Text))
```

می شود این کار را برای همه اطلاعات و یا فقط پسورد به کار برد.

### روش پنجم: رمز گذاری داده های مهم برای جلوگیری از نفوذ پذیری

صفحه وبها و برنامه های کاربردی وب ساخته شده با **ASP.NET** می توانند در بعضی مواقع برای ذخیره اطلاعات مفید باشند. مثلا: برای ذخیره رشته ها در بانکهای اطلاعات از فایل **Web.config** سریعتر نسبت به **Hard-code** در برنامه های کاربردی. در بانکهای اطلاعاتی سرور همیشه باید تغییرات و اصلاحات و کامپایل مجدد انجام داد بنابراین ذخیره اطلاعات به صورت رشته کارا کتری ساده (**این رشته کارا کتری می تواند اطلاعاتی در مورد کاربران ویا پسورد ها باشند**) در فایل **Web.config** کاری بسیار خطرناک است و اصلا پیشنهاد نمی شود چرا که فایل **Web.config** در مستندات (**Document**) **XML** ذخیره شده که این فایل یک فایل متنی ساده است و به راحتی در دسترس دیگران (**هکرها عزیز!!!!**) قرار می گیرد.

به راه ساده و آسان رمز گذاری داده های مهم (**و غیر مهم چون گاهی همان ها هم خطرناکند**) و ذخیره آن به صورت رمز متنی در میان فایل **Web.config** است.

دو الگوریتم برای پنهان سازی وجود دارد :

۱- متقارن (**Symmetric**)      ۲- نامتقارن (**ASymmetric**)



## Hashem(S)

**۱- الگوریتم متقارن:** رمز گذاری و بازکردن رمز اطلاعات با استفاده از یک کلید مشترک باری همه داده ها و همه کاربران. این روش بسیار ساده و کم هزینه است پس دارای امنیت کمتری می باشد چرا؟؟؟ چون نفوذ گر بعد از نفوذ و دسترسی به فایل ما با درست کردن چند کاربر برای خود و کمی مهارت می تواند با مشاهده تغییرات کلید رمز ما را به راحتی کشف کند و آن وقت ....

**۲- الگوریتم نا متقارن:** رمز گذاری و رمز گشایی اطلاعات با چند کلید. این کلیدها دو دسته هستند کلید شخصی و کلید عمومی. رمز گذاری داده ها از یک کلید عمومی استفاده می کند که فقط برای رمز گشایی از یک کلید شخصی استفاده می کند و بر عکس. این الگوریتم دارای پیچیدگی های محاسباتی است و پر هزینه پس دارای امنیت بیشتری نسبت به نوع اول است.

**لیست ۱** نمایش می دهد آرایه ای به نام **Rijndael** را که در الگوریتم رمز گذاری متقارن قرار گرفته و **لیست ۲** نشان می دهد آرایه ای به نام **RSA** را که در الگوریتم رمز گذاری نا متقارن قرار گرفته است. توابعی که در زیر می بینید در لیستهای **۱** و **۲** باعث رمز گذاری داده ها در برنامه های کاربردی وب به شکلی مخصوص و فایل **XML** می شود. **لیست ۲** که مشاهده می کنید یک تابع پشتیبان است که از توابع استفاده شده در لیست **۱** بهره می برد. این تابع پشتیبان باعث تبدیل رشته کاراکتری به آرایه بایتی می شود. تبدیل رشته کاراکتری "1 2 3 4 5 6" به آرایه بایتی {1,2,3,4,5,6}.

برای رمز گذاری نامتقارن نیازمند سازنده کلیدهای عمومی (**Public Key**) و کلید شخصی (**Private Key**) هستیم:

=====For Asymmetric use=====

**استفاده از نامتقارن**

```
Dim publicKey, privateKey As String
Dim RSA As New RSACryptoServiceProvider()
publicKey = RSA.ToXmlString(False) ' get public key
privateKey = RSA.ToXmlString(True) ' get private key
```

=====Asymmetric Encryption=====

**رمز گذاری نامتقارن**

```
Dim cipherText as String = AsymmetricEncryption _
(txtAsyPlainText.Text, publicKey)
```

=====Asymmetric Decryption=====

**رمز گشایی نامتقارن**

```
Dim plainText as String = AsymmetricDecryption _
(txtAsyCipherText.Text, privateKey)
```

برای رمز گذاری متقارن نیاز داریم به کلیدی با ۱۶ بیت و بردار مقدار دهی (IV):

=====Symmetric=====

```
Dim Key, IV as String
Key="1234567890123456"
IV="1234567890123456"
Dim cipherText As String = SymmetricEncryption _
(txtSyPlainText.Text, Key, IV)
```

=====Symmetric=====

```
Dim plainText as String = SymmetricDecryption _
(txtSyCipherText.Text, Key, IV)
```

زیرا پیغامهای **SOAP** (Simple Access Protocol) پروتکل مبتنی بر **XML** برای تبادل اطلاعات ساختار یافته و نوع دادهها در وب) فرستاده می شود در متن بدون رمز سرویسهای وب می توانند همچنین استفاده کنند از فرم رمز گذاری شده در عوض استفاده از **SSL** (**Secure Sockets Layer**) پروتکلی که توسط شرکت **Netscap** طراحی شده و برای رمز گذاری با کلید عمومی استفاده می شود و برای کار های مالی طراحی شده ولی برای امکانات و



## Hashem(S)

سرویسهای دیگر هم کار می کند) از ورود مسیرهای ارتباطی نگهداری می کند از رمز گذاری می توان برای نگهداری از اطلاعات مهم مثل شماره credit cardها از دید دیگران می شود استفاده کرد.

### روش هشتم: ذخیره اطلاعات مهم در رجیستری

برای رمز گذاری دادهای که به صورت دستی وارد می شوند از رجیستری در ذخیره اطلاعات مهم استفاده کرد. برای مثال ممکن برای پیکربندی وب سرور شما به بانک اطلاعات تان با قبول شناسه ها در ویندوز از راه دور Login می شوید و همچنین ممکن در پیکربندی شما استفاده بشه از برنامه های کاربردی وب با استفاده از Impersonate, تخصیص یک User و پسورد خاص این ارتباط را برقرار کنیم.

```
<identity impersonate="true"
username="someuser"
password="tosecret"
/>
```

بنابراین ذخیره Username و پسورد در Web.config به شکل متن بدون رمز کاری بسیار خطر دارد یک کار نسبتاً مناسب ذخیره کردن از Username و پسورد در رجیستری است. در پیروی سری کارهای انجام شده برای ذخیره رشته ها در بانک اطلاعات در فایل Web.config و استفاده از نرم افزار ASPNET\_SETREG.exe که محصول شرکت ماکرو سافت است می توان Username و پسورد را در رجیستری ذخیره نمود.

1- برای دانلود aspnet\_setreg.exe می توانید به آدرس زیر مراجعه کرد:

<http://support.microsoft.com/default.aspx?scid=kb;en-us;Q329290>

2- ساختن یک کاربر جدید برای ویندوز کامپیوترتان. من می توانم با صدا کردن  
"secret" Username: "ASPNETUSER" با پسورد

3- اضافه کردن عبارت <appSetting> در میان فایل Web.config این تغییر را ذخیره می کنیم .

```
<configuration>
<appSettings>
<add key="Distributor" value="workstation id=F16;packet size=4096;integrated
security=SSPI;data
source=F16;persist security info=True;initial catalog=Distributor" />
</appSettings>
```

4- در این کد ما , ما می توانیم رشته کاراکتری معین را که در فایل Web.config استفاده شده را باز یافت کنیم .

```
Dim connStr As String = ConfigurationSettings.AppSettings("Distributor")
Dim Conn As New SqlConnection(connStr)
```

5- استفاده دیگری که می شود از aspnet\_setreg.exe کرد اضافه کردن Username و پسورد به عنوان کاربر که در ASP.NI برای معرفی کاربر در رجیستری :  
C:\>aspnet\_setreg -k:Software\ASPNetApp\Identity -u:ASPNETUSER -p:secret

6- همچنین می شود در ویندوز از پیغامهای طولانی در صفحه پرینت گرفت. در پایین پیغامی را می بینید که دو خط آن پررنگتر از بقیه است ما نیاز داریم که فقط این دو خط را در فایل متنی ذخیره کنیم.

Please edit your configuration to contain the following:

```
userName="registry:HKLM\Software\ASPNetApp\Identity\ASPNET_SETREG,userName"
password="registry:HKLM\Software\ASPNetApp\Identity\ASPNET_SETREG,password"
```



## Hashem(S)

The DACL on the registry key grants Full Control to System, Administrators, and Creator Owner. If you have encrypted credentials for the <identity/> configuration section, or a connection string fo

<sessionState/> configuration section, ensure that the process identity has Read access to the registry key. Furthermore, if you have configured IIS to access content on a UNC share, the account u

access the share will need Read access to the registry key.

Regedt32.exe may be used to view/modify registry key permissions.

You may rename the registry subkey and registry value in order to prevent discovery.

-7 محل فایل Machine.config در:

C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322\CONFIG

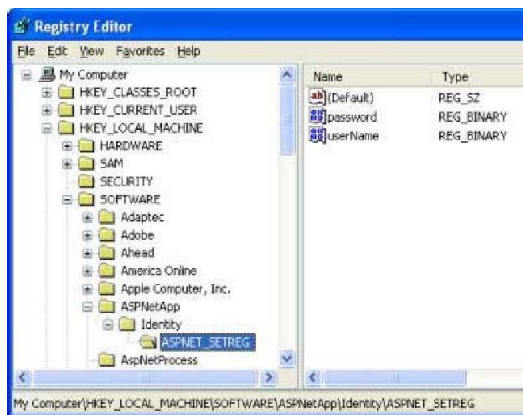
است و اصلاح کردن <identity> در فایل Machine.config به صورت زیر است:

```
<identity impersonate="true"
userName="registry:HKLM\Software\ASPNetApp\Identity\ASPNET_SETREG,userName"
password="registry:HKLM\Software\ASPNetApp\Identity\ASPNET_SETREG,password"
/>
```

ما می توانیم با جلوگیری از تغییر در فایل Machine.config به وسیله اصلاحات در فایل Web.config در برنامه های شما و معین کردن هویت دیگر کاربران و شناسه های آنها.

-8 شروع تغییر در رجستری و خاصیت آن در کامپیوتر به آدرس زیر در Registry Editor می رویم:

HKEY\_LOCAL\_MACHINE\SOFTWARE\ASPNetApp\Identity\ASPNET\_SETREG



(با استفاده از Regedt32) که در (شکل ۸) میبینید.

-9 حالا راست کلیک روی کلید رجیستری

ASPNET\_SETREG کرده و Permission را انتخاب

می کنیم. اضافه کردن یک کاربر در ASPNET و

گذاشتن آن در قسمت Read permission (در شکل ۹

می بینید)

-10 نسبت دادن کاربر ASPNETUSER به Full

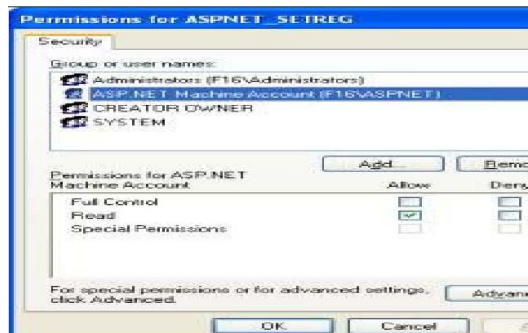
Control در قسمت Machine Account برای

دسترسی مستقیم و درست به فایل

"Temporary ASP.Net" در:

شکل ۸- استفاده از Registry Editor

تغییرات ایجاد شده را در رجیستری می بینید



C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322.

-11 با استفاده از این برنامه توانستیم یک

تولید کاربر با نام ASPNETUSER در رجیستری

تولید کرده با امنیت بالا.

شکل ۹- Read permission تغییر و قرار دادن  
ASPNET در Permission برای خواندن کلیدها





### روش هفتم: انجام چند سیستم داری قبل از Deploy برنامه های کاربردی وب

توضیح:

سیستم داری **Housekeeping** : روتین های از قبیل به روزسازی ساعت یا جمع آوری حافظه های آزاد شده که برای مرتب نگهداشتن سیستم محیط اجرا یک برنامه یا ساختار داده ای یک برنامه طراحی می شود. پیگیری مرحله به مرحله (tracing) در برنامه های کاربردی وب در ASP.Net بسیار ساده است با به کار بردن عبارت <trace> در فایل Web.config به اضافه در دستورات صفحه است. بنابراین موقعی تومی توانی با خواندن Deploy در برنامه ها مطمئن شوی که tracing غیر فعال شدن در هر دو سطح صفحه و سطح برنامه که از کد زیر استفاده کنیم :

```
<trace enabled="false" requestLimit="10" pageOutput="false" traceMode="SortByTime" localOnly="true" />
```

و همچنین خاموش کردن debug در فایل Web.config با کد:

```
<compilation defaultLanguage="vb" debug="true" />
```

در عبارت <customErrors> در Web.config یادتان باشد که مقدار مد را در حالت "RemoteOnly" قرار دهید به صورت زیر:

```
<customErrors mode="RemoteOnly" />
```

مقدار مد می تواند سه مقدار باشد :

- ۱- "on" = همیشه پیغامها را به صورت عادی نشان می دهد (توافقی)
- ۲- "off" = همیشه جزئیات اطلاعات خطا های ASP.Net را نشان میدهد
- ۳- "RemoteOnly" = نمایش عادی پیغامها فقط کاربران نمی توانند اجرا کنند در محدوده وب سرور مان.

این تنظیم را پیشنهاد می کنم چون باعث می شود که جزئیات اطلاعات کاربران نمایش داده نشود. آخرین و نه کمترین کار حذف کردن Solution و فایل های Project در فرم deployment sever است. اگر ما فیلتر ISAPI (این فیلتر یک فایل DLL که IIS ماکروسافت آن را برای اعتبار سنجی و بررسی درستی درخواست های SAPI دریافتی توسط IIS استفاده می کند) را قرار ندهیم بنابراین هکرها خیلی ساده حدس میزنند نام کاربر و دسترسی مستقیم داشتن به فرمهای ما در Web Browser ها.

### روش هشتم: استفاده از Session ها اما نه Cooki-less Session ها

اگه ما احتیاج داشته باشیم به نگهداشتن Session یک کاربر استفاده از موضوع Session برای ذخیره آن. موضوع Session در ASP.Net استفاده میکند از کوکی ها ذخیره شده در Session ID , کوکی بدست می آورد چیزی که مابین کلاینت و سرور رد و بدل می شود. موضوع Session دارای اطلاعات بسیار حساس و حیاتی که در طرف سرور ذخیره می شود. از این رو فقط اطلاعات Session ID که بدون حفاظ گذاشته شده اند و نه اطلاعات حساس و مهم.

ASP.Net پشتیبانی می کند از Cookie-less Session که ممکن است به نظراضافه (temping) بیاید بنابراین خیلی از کاربرها غیر فعال می کنند سیستم کوکی ها رو در مرورگرهاشون. استفاده می شود موضوع Cookie-less Session توسط هکرها در این حال هکر ها می توانند با استفاده از URL که تو در دسترس داری و کارگیری مرورگرها کاری که نباید بکنند را انجام می دهند. در آخر کار باید همیشه جلوگیری کرد از Cookie-less Session همین.

برداشت آخر:

در آخر تذکر می دهم تا می توانید مطالعه کنید کتابهایی در مورد شبکه و .NET و XML.

با امید بهروزی و موفقیت شما  
هاشم