ASP.NET

Application Security

Authentication

- First issue is Authentication
- “Who are you and how do I know you are that person?”
- Examine a credential presented by client (agreed on by 2 parties, e.g., password)
- If not authenticated, considered to be an anonymous user.

Authentication

- ASP.NET offers three different mechanisms for authentication (really 4 if you count “none”).
- None – no authentication used
- Windows – authentication via IIS
- Forms – requests must have a cookie issued from server
- Windows Live ID – cookie issued from Microsoft

Windows Authentication

- Configure IIS to prevent anonymous access to a web site.
- Credentials checked against local user account.
- Fine for Intranet.

Live ID Authentication

- Windows Live ID (formerly Passport) is an attempt to offer a universal one step login procedure.
- Works like forms authentication, except user gets directed to an MS site where they log in and get cookie.
- Can then go to any other Live ID site and you have already authenticated.
- Can also save personal information.

Live ID Authentication

- Most of the sites that use it are MS sites: Hotmail, MSN, Zune, etc.
- A few companies pay for the service.
- 2007 – Windows Live ID Authentication SDK for .NET

Forms Authentication

- User must authenticate with password.
- Subsequent requests for pages accompanied by a cookie.
- Put into web.config:

```
<authentication mode="Forms" >
    <forms name="ASPXAUTH"
        loginUrl="login.aspx" />
</authentication>
```

Forms Authentication

- Does not make your application ask users to login .
- Add an authorization section to block all anonymous users.

```
<authorization>
    <deny users="?" />
</authorization>
```

Forms Authentication Control Flow

1. Request comes in for protected page. An http module examines request for cookie.
2. User sent to login page. Original page request put into query string parameter with key of ReturnUrl=
3. Check user's credentials on login page.
4. If OK, send user to original page request. Now a cookie accompanies all requests.

Forms Authentication

- .NET 2.0 introduced "cookieless" authentication.
- Authorization ticket is packed into the url.

/sampleapp/(XYYZ345)/default.aspx

- ISAPI dll filter intercepts the request, extracts the ticket and rewrites the current path to the app.

<authentication> section

```
<authentication mode="Forms">
<forms loginUrl="Login.aspx"
```
url for logging in if user has no authentication ticket

```
protection="All | None | Encryption | Validation"
```
How to protect the authentication ticket

<authentication> section

```
timeout="30"
```
Specifies a limited lifetime for the authentication ticket

```
name="ASPXAUTH"
```
Name of cookie

```
path="/"
```
Path for cookie

<authentication> section

requireSSL="false"

Cookie sent over non-ssl channel

slidingExpiration="true"

session timeout is periodically reset as long as a user stays active on the site.

defaultUrl="default.aspx"

Page to go to after authenticated. Only used if no ReturnUrl= found in query string

<authentication> section

cookieless="UseDeviceProfile"

- autodetect – uses cookies if browser supports it. Else uses cookieless
- UseCookie – always use cookies regardless of browser capabilities
- UseDeviceProfile – uses cookies if browser supports else cookieless.
- UseUri – never use cookies

enableCrossAppRedirects="false" />

No support for automatic processing of tickets passed between applications.

</authentication>

Login.aspx

```
void Login_Click(object sender, EventArgs e)
{
    bool bauthenticated = false;
    string user = tbUser.Text;  string pwd = tbPass.Text;
    bauthenticated = ValidateUser(user, pwd);
    if (bauthenticated)
       FormsAuthentication.RedirectFromLoginPage(user,
    false);
    else
       errmsg.Text = “Sorry. Not a valid user”;
}
```

Login.aspx

```
bool ValidateUser(string name, string password)
{
    custom logic to validate user
}
```

- When you get to the page you wanted, you can check who the user is with

string name = HttpContext.Current.User.Identity.Name;

Membership & Login Controls

- ASP.NET 2.0 introduces a membership feature and set of login Web server controls that simplify the implementation of applications that use forms authentication.
- Membership provides credential storage and management for application users.
- The membership feature is built on top of a provider model.

Membership

- Active Directory membership provider
- SQL Server membership provider
- Login controls automatically use membership and forms authentication and encapsulate the logic required to:

Login Controls

- Prompt users for credentials
- Validate users
- Recover or replace passwords
- Replace most of what we formerly wrote custom code to do.
- For more information:

http://msdn.microsoft.com/en-us/library/ms998347.aspx