

# Why repeating login

---

*Source:*

<http://www.derkeiler.com/Newsgroups/microsoft.public.dotnet.framework.aspnet.security/2006-08/msg00293.html>

---

- *From:* "ad" <flying@xxxxxxxxxxxxxxxxxx>
  - *Date:* Wed, 30 Aug 2006 17:25:14 +0800
- 

I use login controls in my web application, and have web.config as below:  
<forms name="HealthCookie" loginUrl="Login.aspx" defaultUrl="Home.aspx" protection="Validation" >  
If the user enter wrong password in the login control, the failureText of login control will appear.

When I deploy my web application to another machine, and the use enter the correct password(the failureText does not appear), it login successful(the failureText did not appear), but it did not deirect to Home.aspx, the Login.aspx appear again.

What is wrong ? why repeating login?  
below is my web.config and custum MembershipProvider  
Please help me.

```
<?xml version="1.0"?>
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0>
  <appSettings>
  </appSettings>
  <connectionStrings>
  .....
  </connectionStrings>
  <system.web>
  <httpRuntime maxRequestLength="2097151"/>
  <membership defaultProvider="MyMembershipProvider">
  <providers>
  <add name="MyMembershipProvider" type="MyMembershipProvider"
  minRequiredPasswordLength="7"/>
  </providers>
  </membership>
  <httpHandlers>
  ....
```

## Why repeating login

```
</httpHandlers>
<siteMap>
<providers>
.....
</providers>
</siteMap>
<authentication mode="Forms">
<forms name="HealthCookie" loginUrl="Login.aspx" defaultUrl="Home.aspx"
protection="Validation">
</forms>
</authentication>
<authorization>
<deny users="?"/>
</authorization>
<customErrors mode="RemoteOnly"
defaultRedirect="~/ErrorPage/GenericErrorPage.aspx">
.....
</customErrors>
<pages maintainScrollPositionOnPostBack="true"
masterPageFile="~/MasterPage.master"/>
<sessionState mode="InProc"/>
<compilation debug="true">
<buildProviders>
.....
</buildProviders>
<assemblies>
.....
</system.web>
</configuration>
```

```
public class MyMembershipProvider : MembershipProvider
{
private FormsAuthenticationUserCollection _users = null;
private FormsAuthPasswordFormat _passwordFormat;
//private int _MinRequiredNonAlphanumericCharacters = 0;
private int _MinRequiredPasswordLength = 4;
//private int _MaxInvalidPasswordAttempts = 5;
//private int _PasswordAttemptWindow = 5;
```

#region Not Implemented Members

```
public override string ApplicationName
{
.....

```

```
public override MembershipUser GetUser(string username, bool
userIsOnline)
{
DateTime myDate = DateTime.Today;
MembershipUser user = new MembershipUser(
```

## Why repeating login

```
Name, // Provider name  
username, // Username  
null, // providerUserKey  
"aa@xxxxxxxxxxxxxxxx", // Email  
String.Empty, // passwordQuestion  
"Comment", // Comment  
true, // isApproved  
false, // isLockedOut  
DateTime.Now, // creationDate  
DateTime.Now, // lastLoginDate  
DateTime.Now, // lastActivityDate  
DateTime.Now, // lastPasswordChangedDate  
new DateTime(1980, 1, 1) // lastLockoutDate  
};  
return user;  
  
}

public override bool ChangePassword(string username, string  
oldPassword, string newPassword)  
{  
.....  
}

public override void Initialize(string name,  
System.Collections.Specialized.NameValueCollection config)  
{  
base.Initialize(name, config);  
_passwordFormat = getPasswordFormat();  
string sMin=config["minRequiredPasswordLength"].ToString();  
sMin = WillNs.Util.GetDefault(sMin, "4");  
_MinRequiredPasswordLength = int.Parse(sMin);  
}

public override bool ValidateUser(string username, string password)  
{  
bool Authenticated = false;  
Authenticated = DMHealth.CheckPW(username, password);  
if (Authenticated)  
{  
//HttpContext.Current.Session.Abandon();  
new AuthenticationSuccessEvent(username, this).Raise();  
return true;  
}  
else  
{  
new AuthenticationFailureEvent(username, this).Raise();  
return false;  
}
```

## Why repeating login

```
↓  
  
protected FormsAuthenticationUserCollection getUsers()  
↓  
if ( _users == null)  
↓  
AuthenticationSection section = getAuthenticationSection();  
FormsAuthenticationCredentials creds =  
section.Forms.Credentials;  
_users = section.Forms.Credentials.Users;  
↓  
  
return _users;  
↓  
  
protected AuthenticationSection getAuthenticationSection()  
↓  
Configuration config =  
WebConfigurationManager.OpenWebConfiguration("~/");  
return  
(AuthenticationSection)config.GetSection("system.web/authentication");  
↓  
  
protected FormsAuthPasswordFormat getPasswordFormat()  
↓  
return  
getAuthenticationSection().Forms.Credentials.PasswordFormat;  
↓  
protected MembershipSection getMembershipSection()  
↓  
Configuration config =  
WebConfigurationManager.OpenWebConfiguration("~/");  
return  
(MembershipSection)config.GetSection("system.web/Membership");  
↓  
↓  
  
.
```