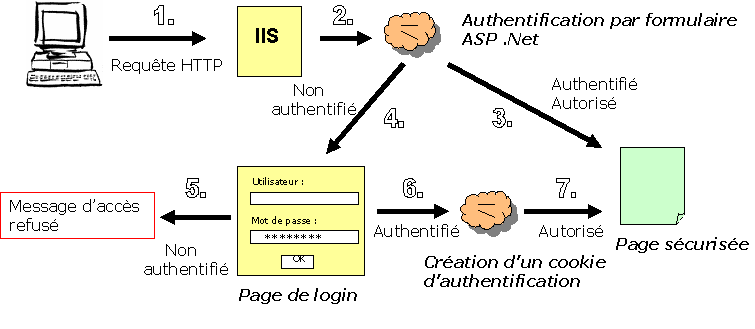
**Introduction**

Nous allons aborder l'authentification par formulaire. Ce mode d'authentification consiste à rediriger tout utilisateur vers une page de login tant que celui-ci ne s'est pas authentifié. L'authentification peut être réalisée soit à partir d'un fichier de configuration XML soit à partir d'un système externe (SGBD ou Active Directory).

Nous allons décrire la mise en œuvre du mécanisme d'authentification à base de formulaire. La liste des utilisateurs autorisés sera dans un premier temps décrit dans un fichier de configuration XML puis intégrée à une table des utilisateurs d'une base de données .

Regardons plus précisément le fonctionnement interne de l'authentification par formulaire.

# II. L'authentification par formulaire sous ASP.NetI. L'authentification par formulaire sous ASP.Net



Passons à la pratique et mettons en place ce système d'authentification en déroulant les étapes suivantes :

* Paramétrage du fichier Web.Config : Définition du mode d'authentification. Définition de la page de redirection. Définition de la liste des utilisateurs authentifiés avec cryptage de leur mot de passe ;
* Création d'un formulaire Web pour recueillir les informations d'identification du client ;
* Vérification des informations d'authentification du client à partir d'un fichier de configuration XML.

**III. Paramétrage du fichier Web.Config**

Pour positionner l'authentification par formulaire, voici les modifications à réaliser dans le fichier Web.Config.

 <authentication mode="Forms">

<forms loginUrl="login.aspx" timeout="20">

<credentials passwordFormat="MD5">

<user name="christian" password="7FF135854376850E9711BD75CE942E07" />

</credentials>

</forms>

</authentication>

<authorization>

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

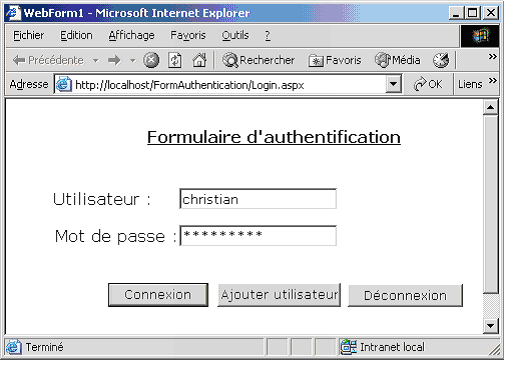
Tout d'abord le mode d'authentification a été basculé à **"Forms"**. La balise **LoginUrl** indique la page vers laquelle l'utilisateur sera systématiquement redirigé tant qu'il ne sera pas authentifié. La balise **timeout** indique la durée en minutes du cookie d'authentification. Les informations concernant les utilisateurs sont définies dans la balise **credentials**. Il faut ensuite définir pour chaque utilisateur un nom et un mot de passe au sein d'une balise user, seuls ces comptes seront habilités à s'authentifier sur notre application.Enfin, nous complétons la sécurité de notre application Web, en interdisant son accès à tous les utilisateurs non authentifiés.

Le cryptage du mot de passe :

Le cryptage du mot de passe peut être réalisé grâce à la fonction **HashPasswordForStoringInConfigFile** de la classe **FormsAuthentication**. Elle permet de générer un mot de passe crypté selon un algorithme de hachage SHA1 ou MD5. Ces deux valeurs peuvent être utilisées comme format du mot de passe ainsi qu'une valeur « Clear » à utiliser si vous ne souhaitez pas le crypter.

# IV. IV. Le formulaire d'authentification

Voici une copie de notre écran d'authentification :

[](https://dotnet.developpez.com/articles/authentification/images/img2.gif)

L'écran d'authentification est relativement simple avec une zone de saisie pour le nom de l'utilisateur, une zone de saisie pour le mot de passe et un bouton de connexion. Ne nous préoccupons pas pour l'instant des autres boutons.

Regardons maintenant le code exécuté lorsque l'utilisateur clique sur le bouton Connexion :

 using System.Web.Security;

....

private void BtConnexion\_Click(object sender, System.EventArgs e)

{

if (FormsAuthentication.Authenticate(txtUtilisateur.Text,txtMotDePasse.Text))

{

FormsAuthentication.RedirectFromLoginPage(txtUtilisateur.Text,false );

}

else

{

lbMessage.Text = "Login incorrect !";

}

}

La méthode Authenticate vérifie le nom d'utilisateur et son mot de passe à partir des éléments contenus dans la balise Credentials du fichier Web.Config.. Si l'utilisateur est habilité alors un cookie d'authentification est généré et l'utilisateur est redirigé vers la page demandée initialement ; ces opérations sont réalisées par la méthode RedirectFromLoginPage. Cette méthode prend également un second paramètre qui indique si le cookie doit persister lors d'un changement de session.

Précisons que dans le cas où la méthode RedirectFromLoginPage ne peut pas identifier de page de retour alors la page default.aspx est affichée.

**V. L'authentification des utilisateurs via une base de données**

Nous venons de mettre en place une authentification par formulaires en validant l'identité de l'utilisateur à partir d'informations contenues dans un fichier de configuration XML. Cette solution fonctionne mais peut devenir rapidement fastidieuse pour un administrateur chargé de saisir un grand nombre de comptes utilisateurs.

Essayons d'améliorer notre solution en validant désormais les informations à partir d'une table « utilisateurs » d'une base de données SQL. Pour ce faire, nous devons :

* Modifier le fichier Web.Config en supprimant la balise **credentials ;**
* Créer une table SQL qui contiendra les informations d'authentification ;
* Développer notre propre fonction d'authentification « Authentifier » chargée de vérifier les informations saisies par l'utilisateur avec celles contenues en base ;
* Modifier le code associé au bouton Connexion afin d'utiliser notre nouvelle méthode.

Description de la table « utilisateurs » :

|  |  |  |
| --- | --- | --- |
| **Champ** | **Type de données** | **Description** |
| Nom | Varchar(20) | Nom de l'utilisateur |
| Mot de passe | Varchar(50) | Mot de passe |

Pour des raisons de sécurité, nous crypterons le mot de passe au format MD5.

Description de la fonction « Authentifier » :

private bool Authentifier(string strUtilisateur, string strMotDePasse)

{

bool bOk=false ;

// Cryptage du mot de passe

strMotDePasse = FormsAuthentication.HashPasswordForStoringInConfigFile(strMotDePasse,"MD5");

// Création d'une connexion SGBD

SqlConnection oConnexion = new SqlConnection("user id=sa;password=;initial catalog=pubs;data source=pttravail");

// Définition de la requête à exécuter

SqlCommand oCommand = new SqlCommand("SELECT \* FROM Utilisateurs WHERE nom='" + strUtilisateur+ "'",oConnexion);

try

{

// Ouverture de la connexion et exécution de la requête

oConnexion.Open();

SqlDataReader drUtilisateur = oCommand.ExecuteReader();

// Parcours de la liste des utilisateurs

while (drUtilisateur.Read())

{

if (drUtilisateur["motdepasse"].ToString() == strMotDePasse)

{

bOk = true ; break ;

}

}

}

catch

{

bOk = false ;

}

oConnexion.Close();

return bOk;

}

Nous cryptons d'abord le mot de passe saisi par l'utilisateur, puis nous sélectionnons la liste des mots de passe de l'utilisateur qui cherche à s'authentifier, enfin nous la parcourons en comparant chaque mot de passe avec celui crypté précédemment. Dès qu'un mot de passe a été trouvé nous retournons Vrai à la fonction appelante.

Il ne nous reste plus qu'à modifier le code associé au bouton Connexion et à invoquer notre méthode Authentifier.

Sélectionnez

private void BtConnexion\_Click(object sender, System.EventArgs e)

{

if (Authentifier(txtUtilisateur.Text,txtMotDePasse.Text))

{

FormsAuthentication.RedirectFromLoginPage(txtUtilisateur.Text,false );

}

else

{

lbMessage.Text = "Erreur d'authentification, l'utilisateur ou le mot de passe n'existent pas!";

}

}

**VI. La déconnexion de l'utilisateur**

La plupart des sites, qui proposent un moyen de s'authentifier par formulaires, fournissent également un bouton de déconnexion. La méthode SignOut() de la classe FormsAuthentication réalise cette opération de déconnexion en détruisant le cookie d'authentification, ce qui obligera l'utilisateur à s'authentifier de nouveau pour accéder à l'une des pages de votre application.

Voici le code exécuté lorsque l'utilisateur clique sur le bouton Déconnexion :

Sélectionnez

private void BtDeconnexion\_Click(object sender, System.EventArgs e)

{

FormsAuthentication.SignOut(); Response.Redirect("Login.aspx");

}

Nous détruisons le cookie d'authentification et redirigeons l'utilisateur vers la page de login.

**VII. Exercice**

Réaliser l'authentification par formulaire avec ASP.NET en utilisant le contrôle Login.