



TARUMT
TUNKU ABDUL RAHMAN UNIVERSITY OF
MANAGEMENT AND TECHNOLOGY

MDECTM
Premier Digital
Tech Institution



Membership and role management

Chapter 5

What Are You Going To Learn?

- At the end of this lesson, you will be able to:
 - Compare authorization and authentication
 - Apply forms authentication
 - Use Login Controls
 - Set authorization in `Web.config` file
 - Manage role

Introduction to Web Security

- Websites rely on identifying users to provide access permissions to data and functions.
- System information leakage Web servers, errors, staff, partner organizations, search engines and rubbish can all be the source of important information about your website - its technologies, business logic and security methods. An attacker can use such information to their advantage so it is important to avoid system information leakage as far as possible.

Basic Principles

Authentication

- The process of verifying the identity of a user or computer
- To verify: Who are you? How do you prove it?
- Credentials include: _____.

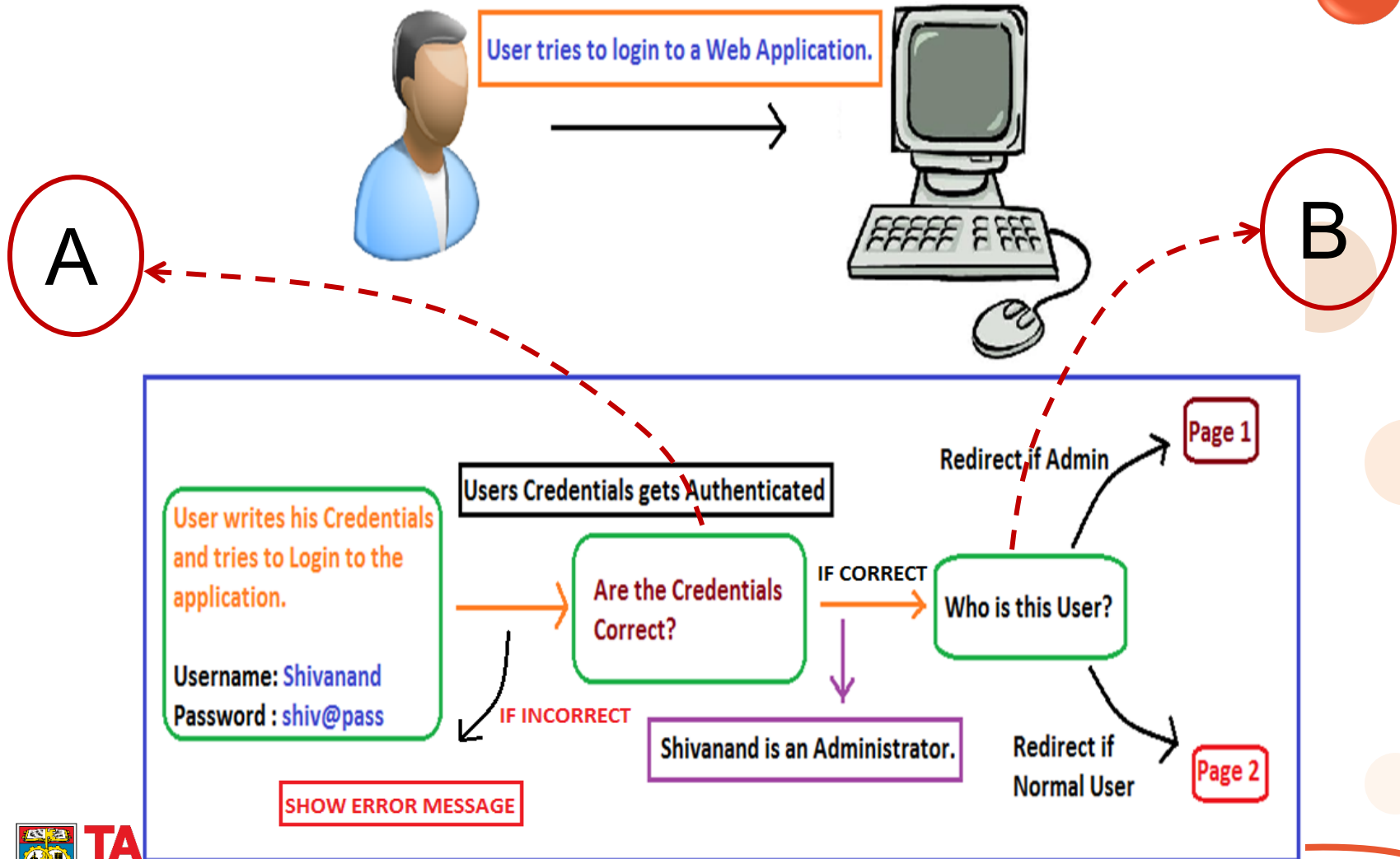
Authorization

- The process of determining what a user (or a group of users) is permitted to do on a computer or network, or to use a resource.
- To verify: What are you allowed to do?
- Example: _____.

ASP.NET Security

- ASP.NET provides the **membership** management service
 - to deal with authenticating users to access a page or an entire site.
 - provides Login control to work with the end user through the process of authentication.
- ASP.NET provides the **role** management service that covers authorization.

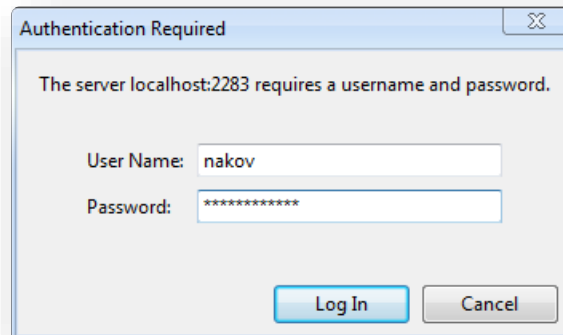
Question: Differentiate between Authentication and Authorization



Types of authentication

- Forms authentication
- Windows authentication *
- Microsoft Account(formerly Passport authentication) *
- Social Login*

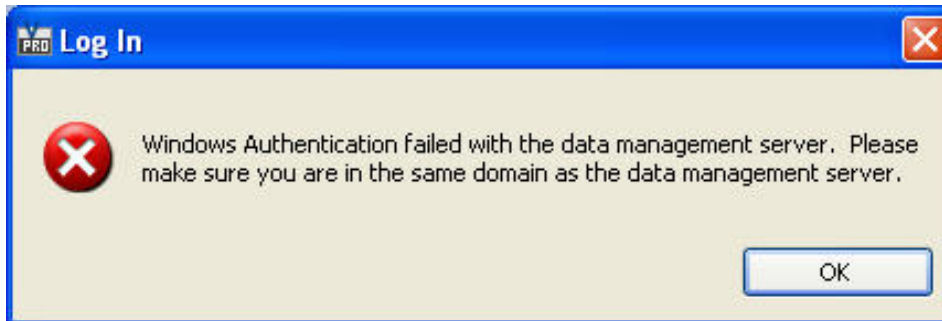
** Not covered in the syllabus*



Windows Authentication

Uses Active Directory/
Windows account

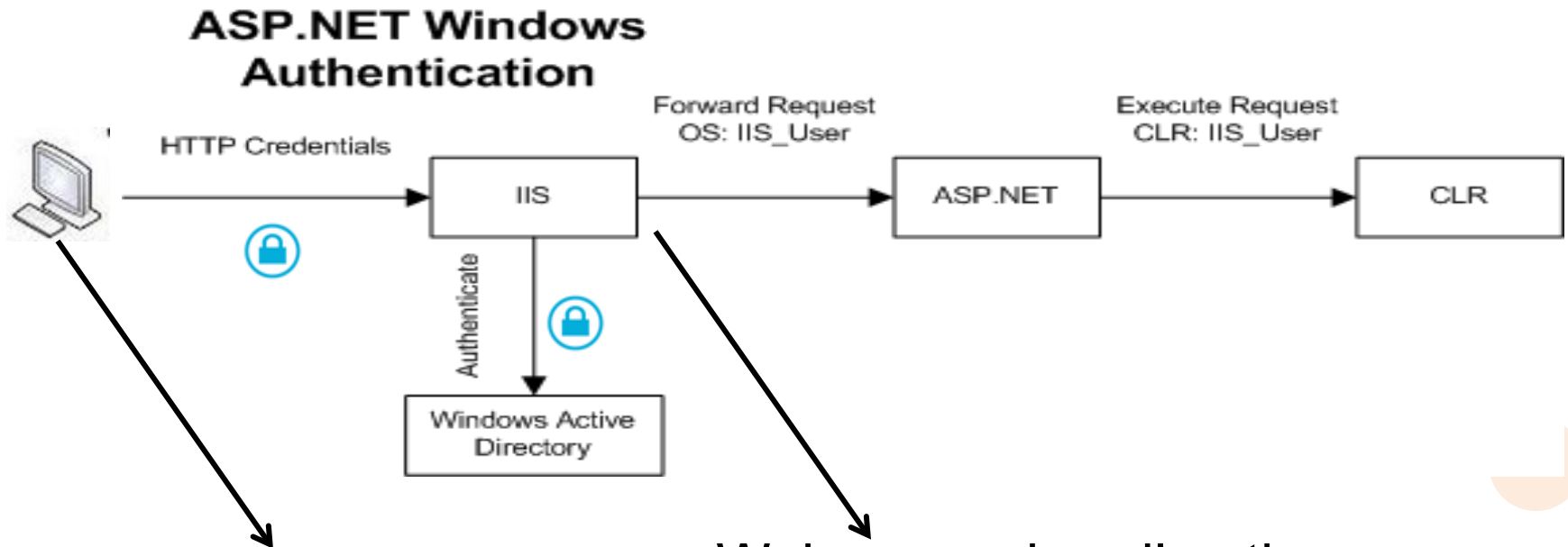
Users credentials are validated with the information stored in the Windows Users Group.



Suitable for developing intranet application

It is not available in Windows 7 Home - Premium, Basic and Starter Versions.

Windows authentication



Login pages pass user credentials to a web server.

Web server handles the authentication using whichever methods is configured on the virtual directory that the application is running within.

Microsoft Account/Passport



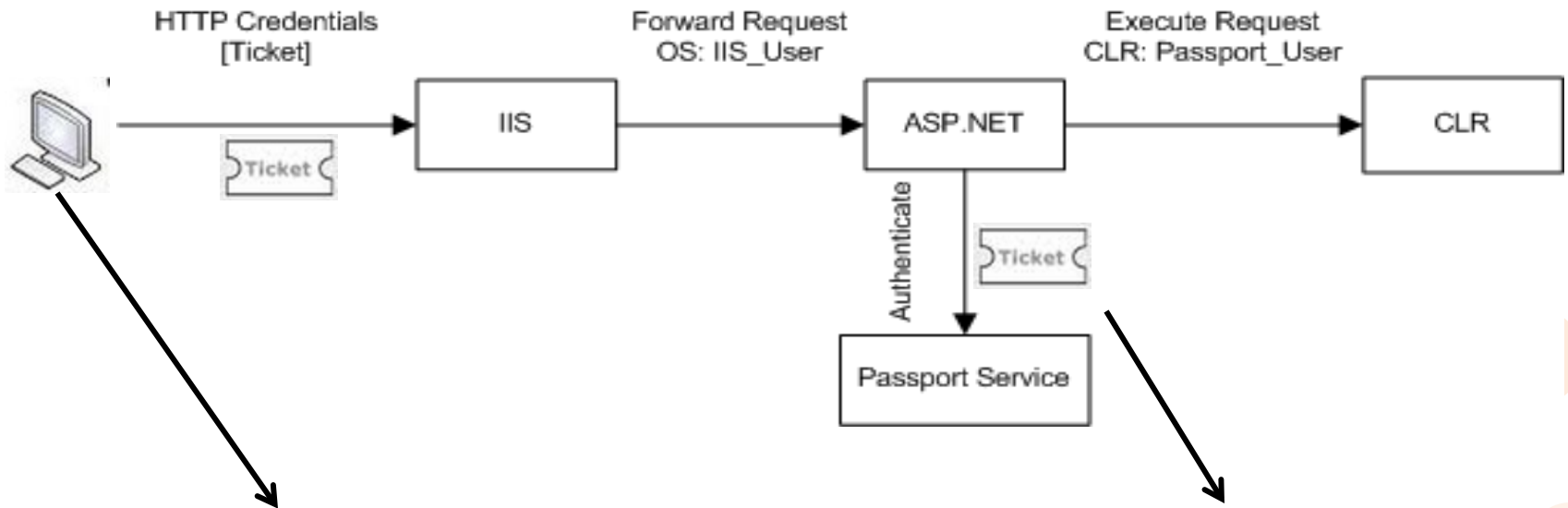
Uses Microsoft's Account service

a centralized **authentication** service provided by Microsoft that offers a single logon and core profile services for member sites

Register using your existing e-mail or @hotmail.com/
@live.com/@msn.com

Passport Authentication

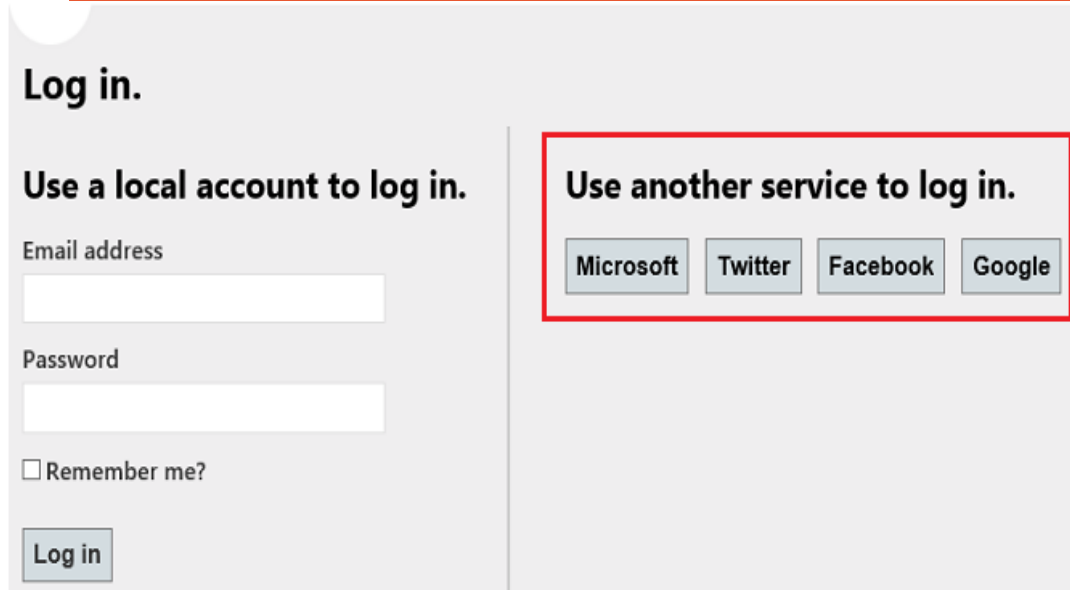
ASP.NET Passport Authentication



Login pages pass user credentials to a web server.

Login credentials are passed to a Microsoft Passport server where user profile are stored centrally. E.g. MSN account

Social Authentication/Social Login



The screenshot shows a login interface with two main sections. The left section, titled 'Log in.', is for local accounts and includes fields for 'Email address' and 'Password', a 'Remember me?' checkbox, and a 'Log in' button. The right section, titled 'Use another service to log in.', is highlighted with a red border and contains four buttons for social login: 'Microsoft', 'Twitter', 'Facebook', and 'Google'.

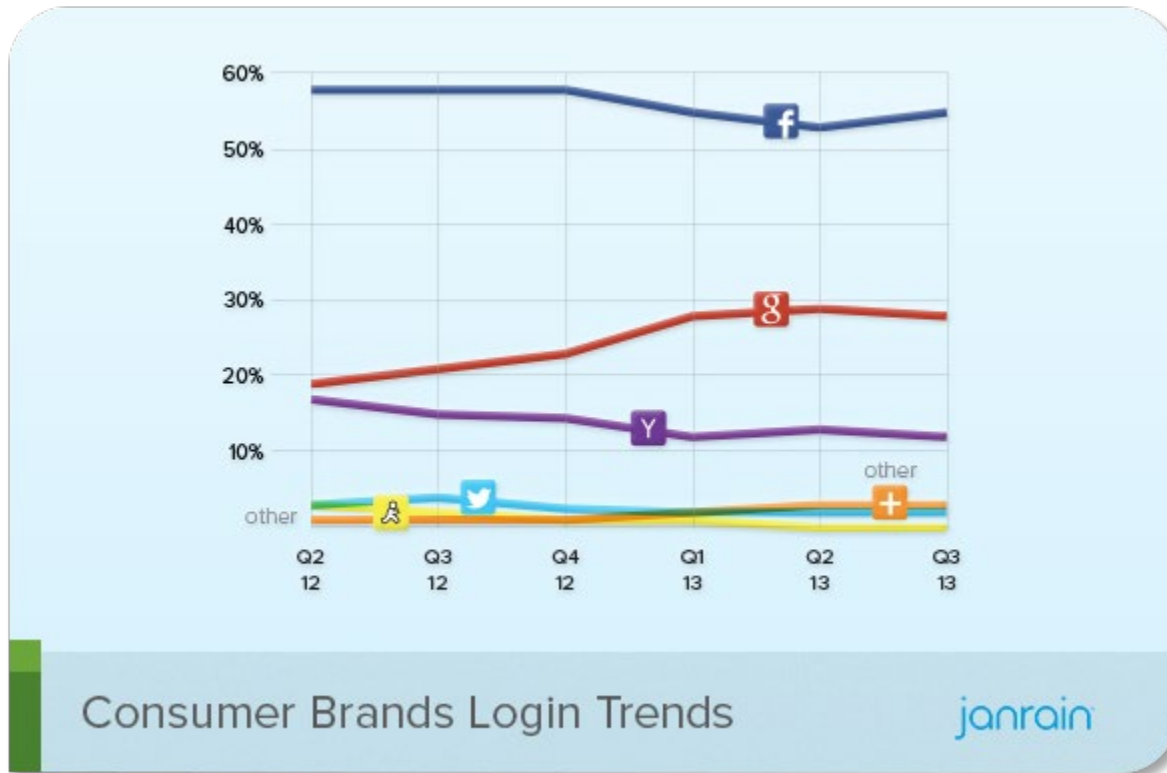
Uses OAuth and OpenID

a form of single sign-on using existing login information from a social networking service such as Facebook, Twitter or Google+ to sign into a third party website

OAuth: An **open protocol** to allow **secure authorization** in a **simple** and **standard** method from web, mobile and desktop applications

OpenID: A Simple Identity layer on top of OAuth 2.0

Social Login



For reference, visit: <http://www.asp.net/web-pages/tutorials/security/enabling-login-from-external-sites-in-an-aspnet-web-pages-site>

Forms Authentication

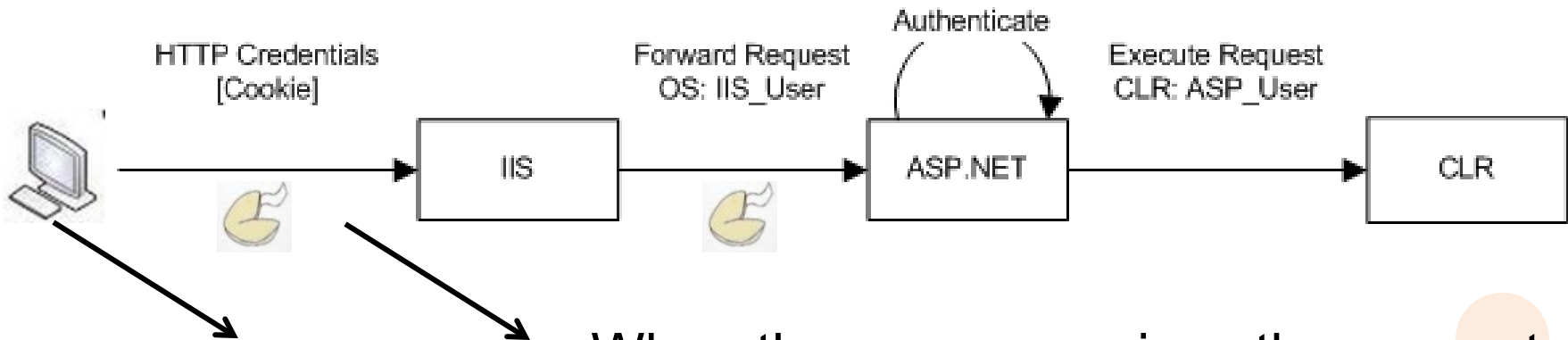


The screenshot shows a web browser window with a login form. The form has a title bar that says "Log In". Below the title bar, there are two input fields: "User Name:" with the text "test" entered, and "Password:". Below these fields is a checkbox labeled "Remember me next time." and a red error message that reads "Your login attempt was not successful. Please try again." At the bottom right of the form is a "Log In" button.

- Uses traditional login/logout pages
- Code associated with a Web form handles users authentication by username / password
- Users are usually stored in a database

Forms Authentication

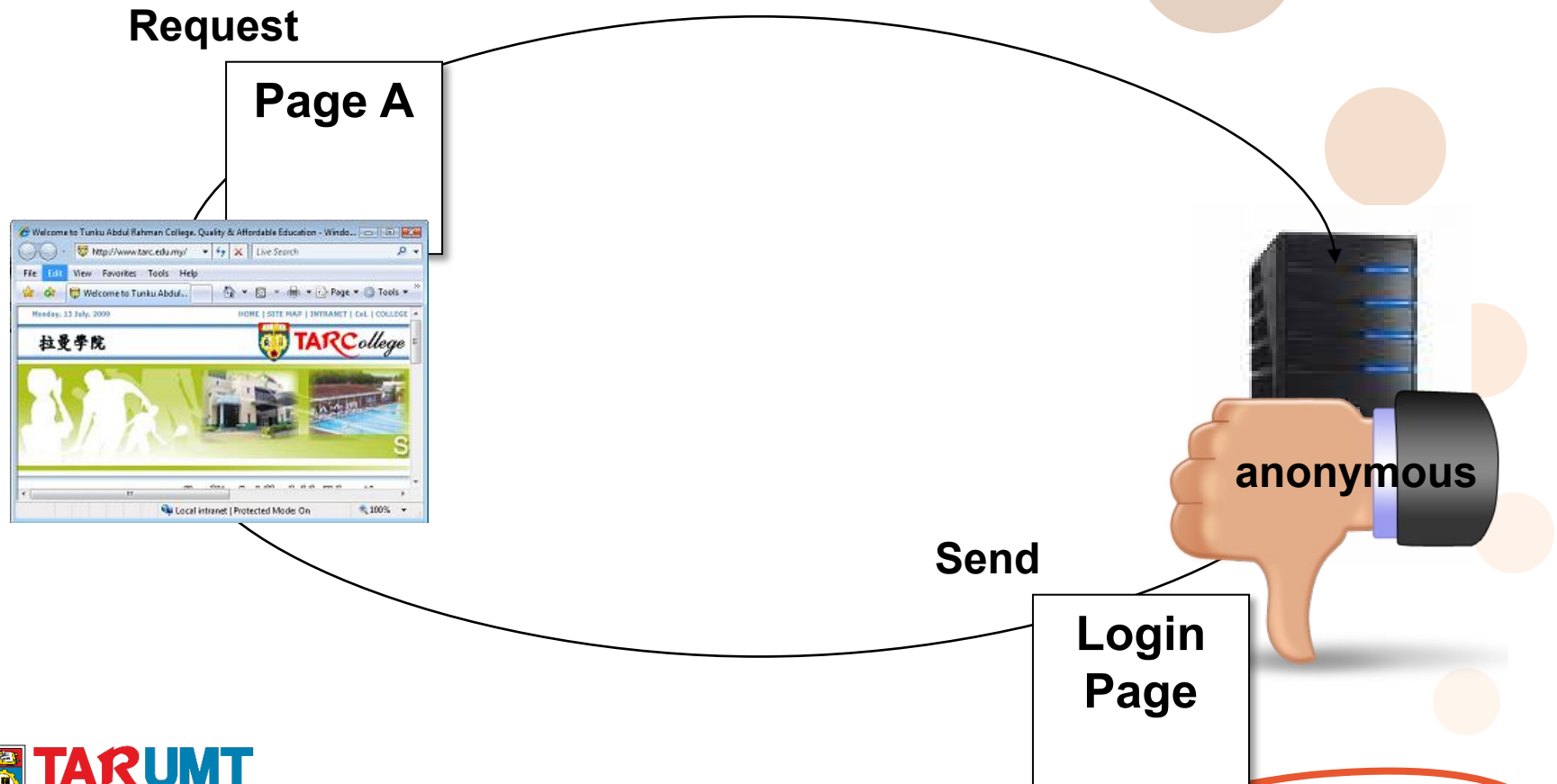
ASP.NET Forms Authentication



Login requests are made by filling in a form on a web page and submitting that form to the server.

When the server receives the request, a cookie is written to the user's local machine, and this cookie is passed back to the server & the browser along with each request that is sent so that the user remains authenticated.

Forms Authentication Model



Forms Authentication Model

**Login request:
Username +
Password**

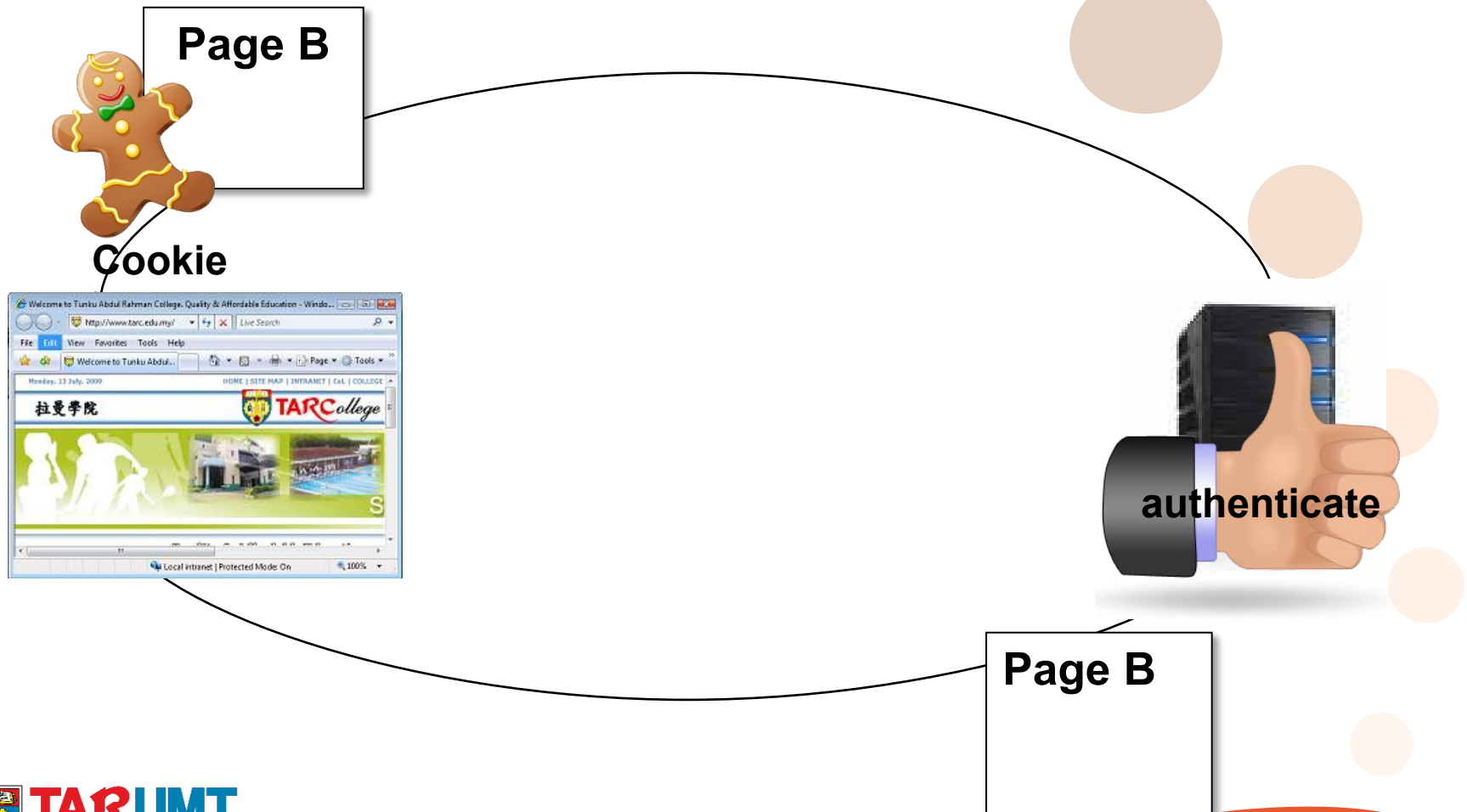


Page A



Cookie


Forms Authentication Model



Advantages and Disadvantages



Method	Advantages	Disadvantages
Windows Authentication	<ul style="list-style-type: none">▪ Suitable to be used in an existing Windows intranet infrastructure	<ul style="list-style-type: none">▪ Unsuitable if web application resides outside of organization intranet
Forms Authentication	<ul style="list-style-type: none">▪ Easy to implement	<ul style="list-style-type: none">▪ Need to maintain your own database
Microsoft Account/ Passport Authentication	<ul style="list-style-type: none">▪ Single sign-on▪ Integrated with other services such as Windows Azure, XBox Live, and Skype.▪ No need to maintain a database to store user information	<ul style="list-style-type: none">▪ Fees involved▪ Windows proprietary protocol as opposed to open source such as OpenID



Web Security Issues

Broken Authentication and Session Management Attack

- If authentication, authorization and session management can be circumvented or altered, a user could access resources they are not allowed to.
- Authentication systems involve passwords, key management, session IDs, and cookies that can allow a hacker to access your account from any computer (as long as they are valid).

Security Measurement

- Beware of how password reminders, remember-me, change password, log out and updating account details are handled, how session tokens are used
- always have login forms on dedicated and encrypted (SSL) pages, i.e. use HTTPS (rather than HTTP)

For more reference, visit: <http://www.asp.net/web-api/overview/security/working-with-ssl-in-web-api>

Security Measurement

- Ask yourself these questions to find out if your website is vulnerable to a broken authentication and session management attack:
 - ☐ Are user credentials weak (e.g. stored using hashing or encryption)?
 - ☐ Can credentials be guessed or overwritten through weak account management functions (e.g. account creation, change password, recover password, weak session IDs)?
 - ☐ Are session IDs exposed in the URL (e.g. URL rewriting)?
 - ☐ Are session IDs vulnerable to session fixation attacks?
 - ☐ Do session IDs timeout and can users log out?

Question

- What are the three authentication provided by ASP.NET?
- How Form Authentication works?

Forms authentication setup

1. Prepare your database.

- Use the `aspnet_regsql` tool to auto-generate and add the Membership table into your database.

2. Configure web.config file.

- Set up connection string to your database. Example:

```
<connectionStrings>  
    <add name="MyAppDB" connectionString="Data Source=(LocalDB)\  
    .....;AttachDbFilename=|DataDirectory|\somedb.mdf;Integrated  
    Security=SSPI;" />  
</connectionStrings>
```


Forms authentication setup

- Add a Membership provider. Example:

```
<membership>
  <providers>
  <clear/>
  <add name="AspNetSqlMembershipProvider"
    type="System.Web.Security.SqlMembershipProvider"
    connectionStringName="....." ← MUST follow your db's connection name
    enablePasswordRetrieval="false"
    enablePasswordReset="true"
    requiresQuestionAndAnswer="false"
    requiresUniqueEmail="false"
    minRequiredPasswordLength="5"
    minRequiredNonalphanumericCharacters="0"
    passwordFormat="Hashed" />
  </providers>
</membership>
```

Forms authentication setup

- Set the authentication mode to Forms. Example:

```
<authentication mode="Forms" >  
    <forms name="anyname" loginUrl="LoginPage.aspx" />  
</authentication>
```

- Set authorization to Allow/Deny access to anonymous users in one or more files/directories in the application.

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

3. Create a login page

Storage Mechanisms

- Forms Authentication classes enable you to store usernames and passwords in different storage mechanisms:
 - Web.config File
 - Database Table
 - XML File (Not covered)

Authenticating Users with Web.Config File

- If you need to store a small number of usernames and password, you can store them directly in the Web.config file.



Hardcoded Credentials should be encrypted

Authenticating Users with Web.Config File

- Storing User Credentials in Web.config

```
<system.web>
  <authentication mode="Forms">
    <forms name="Wrox" path="/" loginUrl="Login.aspx">
      <credentials passwordFormat="Clear">
        <user name="BillEvjen" password="Bubbles"/>
      </credentials>
    </forms>
  </authentication>
</system.web>
```

clear / **MD5** / SHA1



Storing username and password in “Clear” format is a big NO NO

Authenticating Users with a Database Table

- The most common methods of creating a custom user registration system is to use a database table to hold usernames and password. For large Web sites, this solution is the best because it is the most scalable.

Using ASP.NET Login Control

- Login
 - Provides text boxes, buttons and built-in validation to add login functionality
- CreateUserWizard
 - Used to create user accounts
- ChangePassword
 - Allows user to change password
- PasswordRecovery
 - Allows user to retrieve password

Using ASP.NET Login Control

- LoginStatus
 - Give feedback to users so that they know whether they have remembered to log in to the site
 - Control the login and logout processes
- LoginName
 - Display a user's login name if the user has logged in using ASP.NET membership

Using ASP.NET Login Control

- LoginView
 - Display login information
 - Altering the appearance of the page dependent on whether user is logged in or not
 - Show different content to different groups of users



TARUMT
TUNKU ABDUL RAHMAN UNIVERSITY OF
MANAGEMENT AND TECHNOLOGY

MDECTM
Premier Digital
Tech Institution



DEMO

Using Login Controls

Types of users

- Many sites traditionally feature three levels of user security:
 - Anonymous Users
 - Registered Users
 - Administrators

Setting in web.config file

- Under the authorization element, you can set to **deny** or **allow** access to **specified users**.
- To deny access, code a **deny** element and set the users and roles attributes.
- To allow access, code an allow element and set the **users** and **roles** attributes.

Enabling User Role

- To allow role management, add the following under the <system.web> element:

```
<roleManager enabled = "true"/>
```

- But you will have to code manually to add roles. Use the `Role` class from `System.Web.Security`. Functions available:

```
Roles.CreateRole(...)  
Roles.GetUsersInRole(...)  
Roles.GetRolesForUser(...)  
Roles.GetAllRoles(...)  
Roles.AddUserToRole(...)  
Roles.DeleteRole(...)
```

Setting Authorization

```
<system.web>
  <authorization>
    <allow attribute="value"/>
    <deny attribute="value" />
  </authorization>
</system.web>
```

attribute is either 'users' or 'roles'

action is either 'allow' or 'deny'

- ** *attribute* = "value":
 - if all users, write *users*="*"
 - if all unauthenticated users (anonymous) , write *users*="?"
 - if a specific user, write the name, e.g. *users*="Eric"
 - if a specific role, write the role, e.g. *roles* = "Admin"

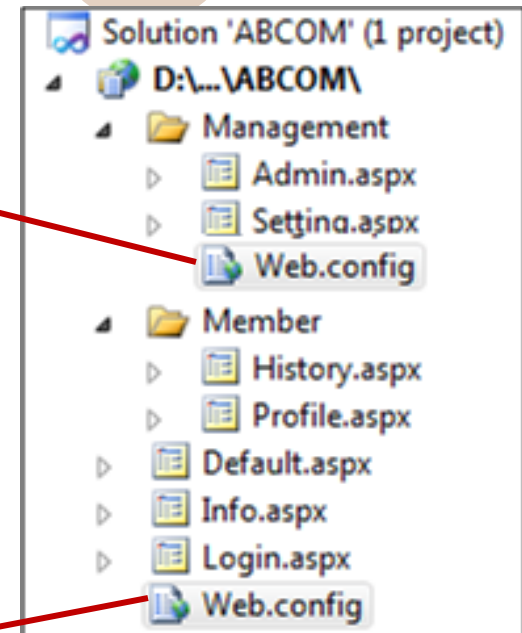
Authorization setting

- Web.config setting in Management directory/folder

```
<authorization>
  <allow roles = "admin" />
  <deny users="*" />
</authorization>
```

- Web.config setting in root

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



Set all requirements on a **single** Web.config

<!-- files in "Management" folder -->

```
<location path="Management">
```

```
<system.web>
```

```
<authorization>
```

```
<allow roles = "admin" />
```

```
<deny users="*" />
```

```
</authorization>
```

```
</system.web>
```

```
</location>
```

<!-- files in root directory-->

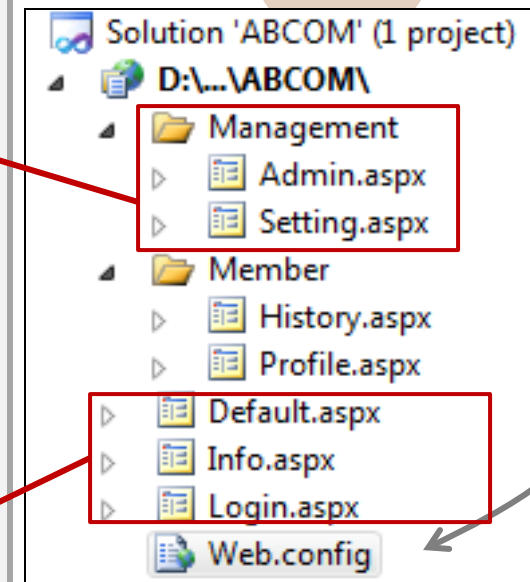
```
<system.web>
```

```
<authorization>
```

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

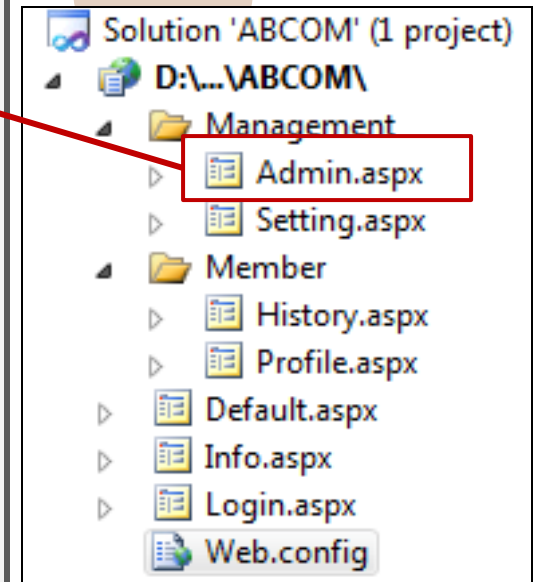
```
</authorization>
```

```
</system.web>
```



Apply authorization requirements to a **single** file

```
<location path="Management/Admin.aspx">
  <system.web>
    <authorization>
      <allow roles = "admin" />
      <deny users="*" />
    </authorization>
  </system.web>
</location>
```



Question

- Consider there are 3 roles: admin, staff and member. Examine the code segments below, what is wrong?

(a)

```
<location path="admin">  
  <system.web>  
    <authorization>  
      <allow roles = "admin" />  
    </authorization>  
  </system.web>  
</location>
```

remark: only admin can
access to all the files in the
current directory

(b)

```
<authorization>  
  <deny users="*" />  
  <allow roles = "admin" />  
</authorization>
```

remark: only admin can
access to all the files in the
current directory

Question

- Assume that there are 2 different roles in a Web application: admin and staff. “staff” can only access the files in the root and “staff_folder”. “admin” can access to all files in the root, “admin_folder” and “staff_folder”. All users must login in order to view all the pages (except Login.aspx that is placed in the root)

Use Two (2) different approaches to demonstrate how the above can be done by inserting the necessary code in the Web.config(s).



TARUMT
TUNKU ABDUL RAHMAN UNIVERSITY OF
MANAGEMENT AND TECHNOLOGY

MDECTM
Premier Digital
Tech Institution



DEMO

MembershipNRoleManagement
(Show the web.config file settings)

ASP Identity

- ASP Membership was designed to solve common requirements in 2005 such as username, password and profile data.
 - Does not support cloud infrastructure (eg. Azure)
 - Can't use Open Web Interface for .NET (OWIN) which supports external identity providers (Facebook, Google, Twitter,...)
- ASP Identity replaces ASP Membership.

ASP Identity

- Features:
 - Role provider: Same as Membership.
 - Claims based: Include rich information about the user in a form similar to key-value pairs. (Beyond scope)
 - Social login
 - OWIN integration: A middleware that decouples the login with the authentication mechanism behind it.
 - E-mail/SMS confirmation.
 - Two-factor authentication.

Using ASP Identity

- A code-first model to specify user data.
 - Located at `~\Models\IdentityModels.cs`
- **Be warned: The login controls in the Toolbox do not applies to ASP Identity.**
- Advisable to just use the generated login templates and change the code from there if needed.
 - Such as adding roles.



TARUMT
TUNKU ABDUL RAHMAN
UNIVERSITY OF
MANAGEMENT AND TECHNOLOGY

MDECTM
Premier Digital
Tech Institution



State management

Next