活动目录的单点登录

王志国

摘　要　本文从业务的角度分析了单点登录的需求和应用领域，从技术本身的角度分析了单点登录技术的内部机制和实现手段，介绍了 Web－SSO 的实现，在 ASP.NET 下给出一个基于中石油邮件用户的 SSO 解决方法。

关键词　ASP.NET，SSO 单点登录，AD，活动目录

一、基本概念

单点登录（Single Sign On），简称为 SSO，是一种认证和授权机制，它允许用户只登录到系统上一次，而后授权访问其他连接的系统，无需再进行登录。SSO 的定义是在多个应用系统中，用户只需要登录一次就可以访问所有相互信任的应用系统。

随着信息技术和网络技术的发展，各种应用服务的不断普及，用户每天需要登录到许多不同的信息系统，如网络、邮件、数据库、各种应用服务器等。每个系统都要求用户遵循一定的安全策略，比如要求输入用户 ID 和口令。随着用户需要登录系统的增多，出错的可能性就会增加，受到非法截获和破坏的可能性也会增大，安全性就会相应降低。因此，在市场上提出了这样的需求：网络用户可以基于最初访问网络时的一次身份验证，对所有被授权的网络资源进行无缝的访问。从而提高网络用户的工作效率，降低网络操作的费用，并提高网络的安全性。

使用“单点登录”整合后，只需要登录一次就可以进入多个系统，而不需要重新登录，这不仅仅带来了更好的用户体验，更重要的是降低了安全的风险和管理的消耗。

二、实现机制

单点登录的机制其实是比较简单的，用一个现实中的例子做比较。在公园内部有许多独立的景点，都可以在各个景点门口单独买票。很多游客需要游玩所有的景点，这种买票方式很不方便，需要在每个景点门口排队买票，钱包拿进拿出的，容易丢失，很不安全。于是绝大多数游客选择在大门口买一张通票（也叫套票），就可以玩遍所有的景点而不需要重新再买票。他们只需要在每个景点门口出示一下刚才买的套票就能够被允许进入每个独立的景点。

单点登录的机制也一样，如图 1 所示，当用户第一次访问应用系统 1 的时候，因为还没有登录，会被引导到认证系统中进行登录；根据用户提供的登录信息，认证系统进行身份校验，如果通过校验，应该返回给用户一个认证的凭据——ticket；用户再访问别的应用的时候就会将这个 ticket 带上，作为自己认证的凭据，应用系统接受到请求之后会把 ticket 送到认证系统进行校验，检查 ticket 的合法性。如果通过校验，用户就可以在不用再次登录的情况下访问应用系统 2 和应用系统 3 了。

从上面的视图可以看出，要实现 SSO，需要以下主要的功能：

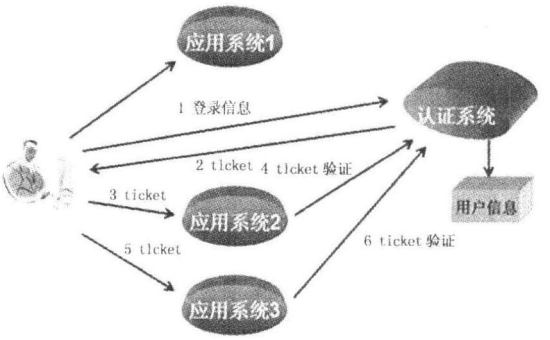

图 1　单点登录的机制

1. 所有应用系统共享一个身份认证系统

统一的认证系统是 SSO 的前提之一。认证系统的主要功能是将用户的登录信息和用户信息库相比较，对用户进行登录认证；认证成功后，认证系统应该生成统一的认证标志（ticket），返还给用户。另外，认证系统还应该对 ticket 进行校验，判断其有效性。

2. 所有应用系统能够识别和提取 ticket 信息

要实现 SSO 的功能，让用户只登录一次，就必须让应用系统能够识别已经登录过的用户，应用系统应该能对 ticket 进行识别和提取，通过与认证系统的通讯，能自动判断当前用户是

否登录过，从而完成单点登录的功能。

上面的功能只是一个非常简单的 SSO 架构，在现实情况下的 SSO 有着更加复杂的结构。

三、Web－SSO 的实现

随着互联网的高速发展，Web 应用几乎统治了绝大部分的软件应用系统，因此 Web－SSO 在 SSO 应用当中最为流行。Web－SSO 有其自身的特点和优势，实现起来比较简单易用。

Web 协议（也就是 HTTP）是一个无状态的协议。一个 Web 应用由很多个 Web 页面组成，每个页面都有唯一的 URL 来定义。用户在浏览器的地址栏输入页面的 URL，浏览器就会向 Web Server 去发送请求。所谓无状态的协议也就是表现在这里，浏览器和 Web 服务器会在第一个请求完成以后关闭连接通道，在第二个请求的时候重新建立连接。Web 服务器并不区分哪个请求来自哪个客户端，对所有的请求都一视同仁，都是单独的连接。这样的方式大大区别于传统的 C/S 结构，在那样的应用中，客户端和服务器端会建立一个长时间的专用的连接通道。正是因为有了无状态的特性，每个连接资源能够很快被其他客户端所重用，一台 Web 服务器才能够同时服务于成千上万的客户端。

通常的应用是有状态的。浏览器和服务器之间有约定：通过使用 cookie 技术来维护应用的状态。cookie 是可以被 Web 服务器设置的字符串，并且可以保存在浏览器中。当浏览器访问了页面 1 时，Web 服务器设置了一个 cookie，并将这个 cookie 和页面 1 一起返回给浏览器，浏览器接到 cookie 之后，就会保存起来，在它访问页面 2 的时候会把这个 cookie 也带上，Web 服务器接到请求时也能读出 cookie 的值，根据 cookie 值的内容就可以判断和恢复一些用户的信息状态。

Web－SSO 完全可以利用 cookie 结束来完成用户登录信息的保存，将浏览器中的 cookie 和上文中的 Ticket 结合起来，完成 SSO 的功能。

为了完成一个简单的 SSO 的功能，需要两个部分的合作：首先要统一的身份认证服务；其次需要修改 Web 应用，使得每个应用都通过这个统一的认证服务来进行身份校验。

四、活动目录

Active Directory 是面向 Windows Standard Server、Windows Enterprise Server 以及 Windows Datacenter Server 的目录服务。

Active Directory 存储了有关网络对象的信息，并且让管理员和用户能够轻松地查找和使用这些信息。Active Directory 使用了一种结构化的数据存储方式，并以此作为基础对目录信息进行合乎逻辑的分层组织。

Active Directory 支持标准的 LDAP(轻型目录访问协议)，它是一种工业标准服务协议，在大多数平台和大多数开发语言中都可以进行访问。

五、基于 AD 的 SSO

ASP. NET 提供了 forms 验证的方式，应用系统可以通过这种方式对 AD 进行用户认证。Forms 身份验证通常指这样一个系统，在该系统中使用 HTTP 客户端重定向将未经身份验证的请求重定向到 HTML 窗体。如果应用程序需要在登录时通过 HTML 窗体收集自己的用户凭据，那么选择 Forms 身份验证就很好。用户提供凭据并提交该窗体。如果应用程序对请求进行身份验证，系统会发出一个 Cookie，在其中包含用于重新获取标识的凭据或密钥。随后发出在请求头中具有该 Cookie 的请求。ASP. NET 事件处理程序使用应用程序指定的任何验证方法对这些请求进行身份验证和授权。应用系统从 form 中获得用户名和密码后，然后调用 ADSI (Active Directory Services Interface)在 AD 中进行用户验证。

1. 创建一个 ASP. NET 的 Web 应用

首先，创建 logon. aspx；添加 stem. DirectoryServices. dll 引用，logon. aspx 的页面如图 2 所示：

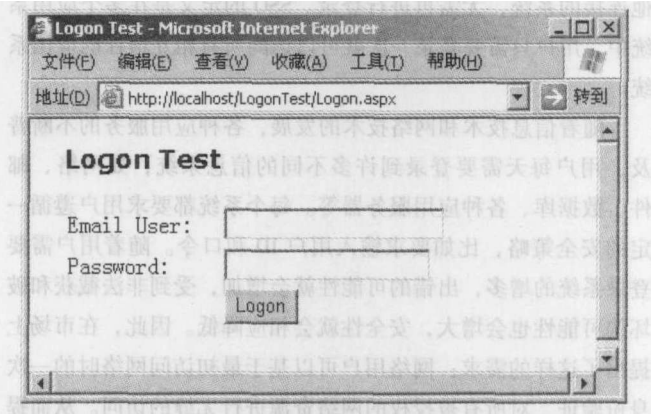

图 2 Logon. aspx 的页面

logon. aspx. cs 实现代码如下：

```
private void Submit1_ServerClick(object sender, System.EventArgs e)
{
if (this.CheckDomainUser(this.UserEmail.Value, this.UserPass.Value))
    {
this.Msg.Text = "登录成功,欢迎你" + this.UserEmail.Value;
    }
    else
    {
    Msg.Text ="登录失败，请检查用户名和密码．";
    }
}
```

```
        public bool CheckDomainUser(string uid, string pwd)
        {
            bool flag1;
            try
            {
    if (pwd == null || pwd.Trim().Length == 0)
                {
                    return false;
                }
                string sPath = "LDAP://ptr.petrochina/DC=ptr,DC=petrochina";  //登录 ptr 域的配制
                DirectoryEntry entry1 = new DirectoryEntry(sPath, uid, pwd, AuthenticationTypes.ServerBind);
        object obj1 = System.Runtime.CompilerServices.RuntimeHelpers.GetObjectValue(entry1.NativeObject);
                flag1 = true;
            }
            catch (Exception ex)
            {
                this.Msg.Text = ex.ToString();
            flag1 = false;
            }
            return flag1;
        }
```

2. 配置 Web 项目的 Forms 认证

修改需要进行单点登录认证的程序项目中的 web.config 文件，这样就把认证都指向上面完成的 AD 用户认证页面上了。比如，在 Adtest 项目中，修改 Web.config 文件，那么在访问 Adtest 项目文件的时候，系统将自动转到 logontest\Logon.aspx 页面进行认证

```
<authentication mode="Forms">
  <forms name=".ASPXFORMSAUTH" loginUrl="\logontest\Logon.aspx" />
</authentication>
<authorization>
  <deny users="?" />
  <allow users="*" />
</authorization>
<identity impersonate="true" />
```

3. 邮件测试

在未进行身份认证的时候，访问任意页面，都将自动转到系统唯一的认证页面进行认证，从而实现了单点登录。当访问 ADtest 项目的 default.aspx 页面的时候，系统自动转到 Logontest 中的 login.aspx，即系统单一的登录页面，如图 3 所示

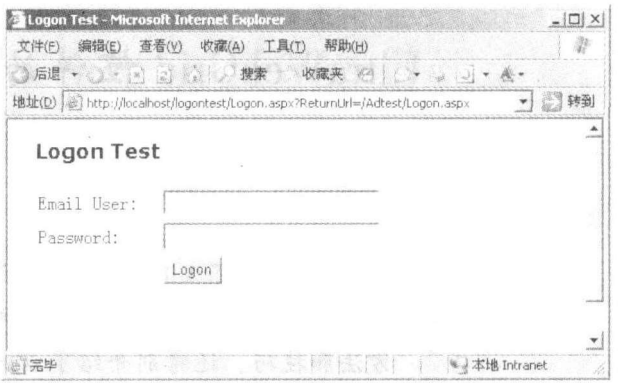

图 3 系统单一的登录页面

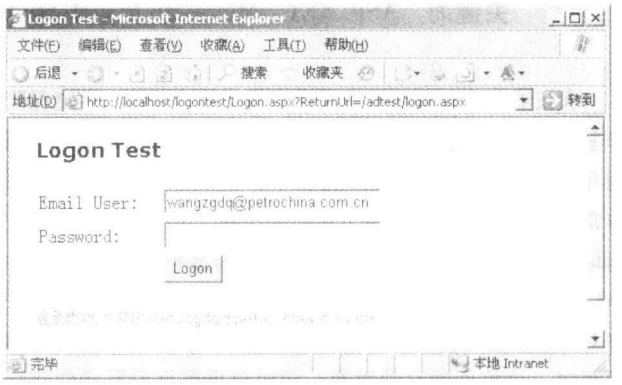

图 4 邮件用户登录成功

当用户输入正确的账号和口令后，系统就会显示登录成功，如图 4 所示，并有可能进行相应的操作，如返回原始的页面，或进行下一步的操作等

六、结语

单点登录为企业的应用开发提供通用的、单一的终端用户登录接口，给用户带来了便捷，更重要的是降低了系统的安全风险和管理的消耗。单点登录为资源访问控制、统一身份认证系统等问题提供了一种可行的方案

参考文献：

[1] Chris Goode, John Kauffman 等著，杨浩译. ASP.NET1.0 入门经典. 清华大学出版社.

[2] Alex Ferrara Matthew MacDonald 著，天宏工作室译. .NET WEB 服务编程. 清华大学出版社.

（收稿日期：2007 年 7 月 16 日）