[在IIS上SSL的部署和启动SSL安全](http://www.cnblogs.com/whitewolf/archive/2010/07/07/1773066.html)

2010-07-07 18:09 by 破狼, 15664 阅读, 24 评论, [收藏](http://www.cnblogs.com/whitewolf/archive/2010/07/07/1773066.html), [编辑](http://i.cnblogs.com/EditPosts.aspx?postid=1773066)

*在这次的项目中遇见了这个问题，之前我并懂了不了多少，只对了SSL和HTTPS理论了解。但并不知道在实际中如何运行。经过自己在网上查阅一番，最后靠自己解决了这个问题，现在在这里和大家分享一下。如果写的有不对或者是不恰当的，就请大家指正，多交流。*

*SSL（安全套接子层：Secure Socket Layer）：*

*SSL是Secure Socket Layer（安全套接子层）：是由网景公司（Netscape）自主研发的用以保障在Internet上敏感数据传输之安全，利用数据加密技术，可确保数据在网络上之传输过程中不会被截取及窃听。SSL协议位于TCP/IP协议与各种应用层协议之间，为数据通讯提供安全支持。SSL协议可分为两层： SSL记录协议（SSL Record Protocol）：它建立在可靠的传输协议（如TCP）之上，为高层协议提供数据封装、压缩、加密等基本功能的支持。 SSL握手协议（SSL Handshake Protocol）：它建立在SSL记录协议之上，用于在实际的数据传输开始前，通讯双方进行身份认证、协商加密算法、交换加密密钥等。SSL协议提供的服务主要有：１：认证用三户和服务器，确保数据发送到正确的客户机和服务器；２：加密数据以防止数据中途被窃取；３：维护数据的完整性，确保数据在传输过程中不被改变。*

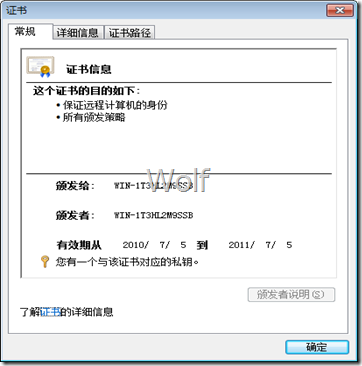
*HTTPS（安全超文本传输协议：Secure Hypertext Transfer Protocol）：*

*HTTPＳ（Secure Hypertext Transfer Protocol）安全超文本传输协议就是应用了网景公司的安全套接字层（SSL）作为HTTP应用层的子层。（HTTPS默认端口为443）SSL使用40 位关键字作为RC4流加密算法，这对于商业信息的加密是合适的。HTTPS和SSL支持使用X.509数字认证，如果需要的话用户可以确认发送者是谁．HTTPS是以安全为目标的HTTP通道，简单讲是HTTP的安全版。即HTTP下加入SSL层，https的安全基础是SSL，因此加密的详细内容请看SSL。HTTPS是一个URI scheme(抽象标识符体系)，句法类同http:体系。用于安全的HTTP数据传输。https:URL表明它使用了HTTP，但HTTPS存在不同于HTTP的默认端口及一个加密/身份验证层（在HTTP与TCP之间）。这个系统的最初研发由网景公司进行，提供了身份验证与加密通讯方法，现在它被广泛用于万维网上安全敏感的通讯，例如交易支付方面。*

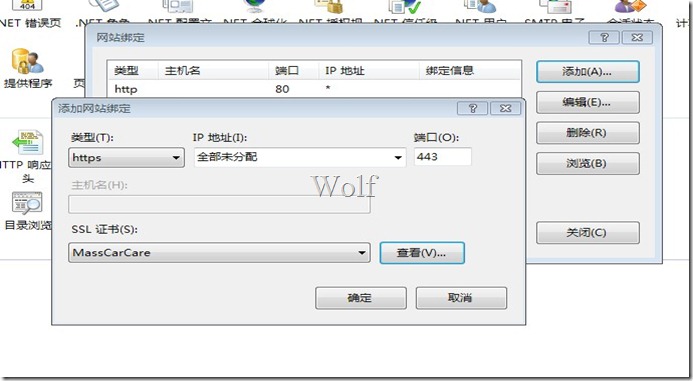
*在IIS上的HTTPS配置：*

*１：打开控制面板－》管理工具－》IIS；*

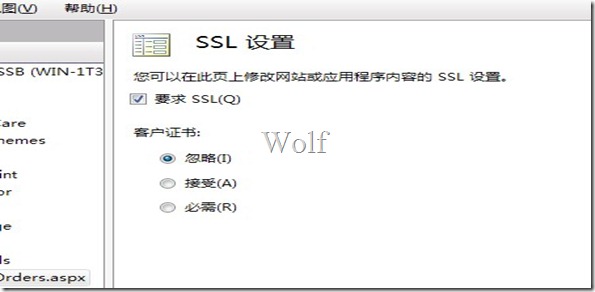
*２：选中IIS上的顶级节点应用，在右边的功能试图里选择服务器证书，双击进入，可以看见配置在本机的证书，我们为了试验，需要在右边的点击创建自签名证书，按照操作进行完成自签名证书。在实际应用中这个证书是需要在第三方申请的。以下是我的自定义证书：*

*[](http://images.cnblogs.com/cnblogs_com/whitewolf/WindowsLiveWriter/IISSSLSSL_C0FF/image_2.png)*

*3：选择需要启动SSL的网站，并点击网站，在最右边的工具选择绑定,就可以添加HTTPS端口和服务证书，如下图：*

*[](http://images.cnblogs.com/cnblogs_com/whitewolf/WindowsLiveWriter/IISSSLSSL_C0FF/NTLA%5b@SI@J1H5G2CNEEBJMR_2.jpg)*

                4：在我们需要启动SSL的网站是点击中间的SSL设置按钮，选择启动SSL，以及确定客户端是否必须持有证书。如入：

[](http://images.cnblogs.com/cnblogs_com/whitewolf/WindowsLiveWriter/IISSSLSSL_C0FF/B7L%7dL%5b00P9B3U3W%5d51NPH%7bX_2.jpg)

这是在IIS中就配置完成了，我们就可以利用HTTPS或者HTTP两种方式去访问我们所启动SSL的网页了，他们的不同在于不同的端口，和安全性。

在IIS中以HTTPS访问页面页面会出现阻止界面：因为我们自定义的证书在Ie中认为是可疑证书如下图，就表明配置成功了。

点击[继续浏览此网站(不推荐)。](http://www.cnblogs.com/whitewolf/archive/2010/07/07/1773066.html" \t "_blank)就可以访问页面了。

[](http://images.cnblogs.com/cnblogs_com/whitewolf/WindowsLiveWriter/IISSSLSSL_C0FF/%E6%9C%AA%E5%91%BD%E5%90%8D_2.jpg)

   接下来我们主要是在程序中访问页面，和HTTP没有什么区别，很简单，你只需要将其当做HTTP一样处理就是了。唯一需要注意的是他们的端口不同，所以在Response.Redirect（“url”）的时候我们必须转化为对应的决定URL，而不是相对了。这里就需要用Request.Host和Port来拼接了。就这么多了，我也挤不出来了把我的拼接类贴出来共享下：

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Collections.Specialized;

using System.Diagnostics;

namespace MassCarCare.Web.Utils

{

public class URLHelper

{

public HttpContext HttpContext

{

get;

set;

}

public string GetAbsoluteURL(string page, string pre, NameValueCollection paramCollection)

{

if (HttpContext == null)

throw new ArgumentNullException("HttpContext");

string port = ""; ;

if (System.Configuration.ConfigurationManager.AppSettings[pre.ToUpper() + "Port"] != null)

{

port = System.Configuration.ConfigurationManager.AppSettings[pre.ToUpper() + "Port"].ToString();

}

if (string.IsNullOrEmpty(port))

{

if (pre.ToLower().Equals("http"))

{

port = "80";

}

else if (pre.ToLower().Equals("https"))

{

port = "443";

}

else

port = HttpContext.Request.Url.Port + ""; ;

}

string pageUrl = pre + "://" + HttpContext.Request.Url.Host + ":" + port + "/" + page;

string param = "";

if (paramCollection != null)

{

foreach (var item in paramCollection.Keys)

{

param += "&" + item + "=" + paramCollection[item.ToString()];

}

}

if (!string.IsNullOrEmpty(param))

{

param = param.TrimStart('&');

pageUrl += "?" + param;

}

Debug.Write(pageUrl);

return pageUrl;// HttpUtility.UrlEncode(pageUrl);

}

public string GetAbsoluteURL(string page, NameValueCollection paramCollection)

{

return GetAbsoluteURL(page, "http", paramCollection);

}

public string GetAbsoluteURL(string page, string pre)

{

return GetAbsoluteURL(page, pre, null);

}

public string GetAbsoluteURL(string page)

{

return GetAbsoluteURL(page, "http", null);

}

}

}