

IIS Setting prevents AD Query from working?

Source:

<http://www.tech-archive.net/Archive/Internet-Server/microsoft.public.inetserver.iis/2004-08/0262.html>

From: patrick (patl_at_reply.newsgroup.msn.com)

Date: 08/04/04

Date: Wed, 4 Aug 2004 12:27:41 +0100

After a bit of experimentation, i found out that, the AD query with the ASP.NET on WSS Web Part

- works when IIS is set to only use Basic Authentication
- does NOT work when IIS is set to only use Integrated Windows Authentication

Obviously, Basic Authentication is not good enough. I read somewhere which says that when NTLM is used for Integrated Windows authentication, then the credentials are NOT delegatable. Apparently, the credentials from IIS is only delegatable when Integrated Windows Authentication is used if Kereberos is used.

My Settings

- Sharepoint portal server /Windows Sharepoint Services running on IIS6 on a Windows 2003 Server
- Domain Controller on a seperate Windows 2000 server

I have attempted to force IIS6 to use Kereberos by

1. In IIS Manager, right-click the local computer, and then click Properties.
2. Select the Enable Direct Metabase Edit check box.
3. Edited %systemroot%\system32\inetsrv\MetaBase.xml
4. Under the website in question, changed NTAAuthenticationProviders from "NTLM" to "Negotiate,NTLM"

(but as soon as I did 4, the site won't let me logon!)

How could I force IIS6 to use Kerebors Authentication when Integrated Windows authentication is selected (assuming this is the right way forward)?

Note, I have also marked the domain user account I am trying to logon as throug IE6SP1 as delegatable.

"patrick" <patl@reply.newsgroup.msn.com> wrote in message news:eE4dPXgeEHA.2812@tk2msftngp13.phx.gbl...

> *This is really weird.....*

>

microsoft.public.inetsrvr.iis: IIS Setting prevents AD Query from working?

> I got the following code (at the end of this message) working in a Win Forms
Forms
> C# .NET application, however when done as a Web Part for Sharepoint Portal
Portal
> Server/WSS 2003, it doesn't work. The forms App and the WebParts have the
> same label name and text box names. In both Web.config for the SPS
> Application and the machine.config, <identity impersonate="true" /> is set
> without a username (so it should be passing in the identify of the person
> who logged onto SPS/WSS.
>
> When I run this via a Forms application, I get the attributes of users
> matching the firstname and surname listed in the label.
>
> However, when I run it as a Web Parts (after compiling it into a Cab File,
> installed it using stsadm, populated it to a Gallery, and added it to a
> subsite under the portal), I get the following displayed
>
> Note that when I did an LDIFDE as follows, the ADOutput.ldf does indicate
> that the givenName and sn attribute were exported, so they do exists!!
> LDIFDE -f ADOutput.ldf -r "(objectClass=user)"
>
>
> -----Start of message displayed-----
> Results Results for filter
> (&(objectClass=user)(givenName=*C*)(sn=*C*))---beforeSearch--
>
> System.Runtime.InteropServices.COMException (0x8007200A): The specified
> directory service attribute or value does not exist at
> System.DirectoryServices.DirectoryEntry.Bind(Boolean throwIfFail) at
> System.DirectoryServices.DirectoryEntry.Bind() at
> System.DirectoryServices.DirectoryEntry.get_AdsObject() at
> System.DirectoryServices.DirectorySearcher.FindAll(Boolean
findMoreThanOne)
> at System.DirectoryServices.DirectorySearcher.FindAll() at
> AcmeWebParts.ActiveDirectoryWebPart.ADHandler(Object sender, EventArgs e)
> -----End of message displayed-----
>
>
>
>
> -----Start of Code Snippet (ADHandler)-----
> //ADHandler is the event handler for a button which when clicked is meant
to
> search the AD.
> string strTemp= "";
> strTemp = "Results for filter (&(objectClass=user)(givenName=*" +
> txtFirstname.Text + "*)(sn=*" + txtSurname.Text + "*)---";
> try
> {
> DirectoryEntry objADRoot = new
> DirectoryEntry("LDAP://dc=myDomain,dc=co,dc=test");

microsoft.public.inetserver.iis: IIS Setting prevents AD Query from working?

```
> DirectorySearcher mySearcher = new DirectorySearcher(objADRoot);
> //(giveName=" + txtFirstName.Text + "*"
> mySearcher.Filter= "(&(objectClass=user)(givenName=*" +
> txtFirstname.Text + "*)(sn=*" + txtSurname.Text + "*)";
> strTemp += "beforeSearch--";
> SearchResultCollection colSearchResults = mySearcher.FindAll();
> strTemp += "afterSearch--<br>";
> foreach(SearchResult objEachResult in colSearchResults)
> {
> // Iterate through each property name in each SearchResult.
> foreach(string strPropKey in objEachResult.Properties.PropertyNames)
> {
> strTemp = strTemp + strPropKey + "=";
> // Retrieve the value assigned to that property name
> // in the ResultPropertyValueCollection.
> ResultPropertyValueCollection colVal =
> objEachResult.Properties[strPropKey];
> // Iterate through values for each property name in each
SearchResult.
> foreach(Object objProp in colVal)
> {
> // Handle results. Be aware that the following WriteLine
> // only returns readable results for properties that are strings.
> strTemp = strTemp + objProp.ToString();
> }
> strTemp= strTemp + "<br>\n";
> }
> }
> lblDisplay.Text = strTemp;
> }
> catch (Exception excp)
> {
> lblDisplay.Text= strTemp + "<br><br>" + excp.ToString();
> }
> -----End of Code Snippet (ADHandler)-----
>
>
```