

Can it work...? – ASP.NET Impersonation with Remoting

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.remoting/2007-12/msg00001.html>

- *From:* Dave <davidhomer@xxxxxxxxxx>
 - *Date:* Tue, 4 Dec 2007 09:06:54 –0800 (PST)
-

Hello,

OK I think this might be a challenge for you...

I have an ASP.NET Web Page / Web Service that runs on IIS and uses impersonation. The user accesses the site and code runs as that account no problems. I find however is ASP.NET tries to access Active Directory or Exchange via CDOEXM that it does not work as the AD Server expects a Principal token not a cheapo Impersonation token

OK no problems..... I'll create a Service running as a User Account with sufficient Privilege to do the work on Exchange and Active Directory and the ASP.NET Web Page / Web Service will Remote to it and request it do the work (The Service will check the credentials and only perform the action based on it's own ACLs....) The Service runs on the same machine as IIS so I will use IPC remoting.

Great. Works fine I can see that the WindowsIdentity is the SuperAccount we created which will do the work, and the My.UserName correctly reports the user account that made the call (so I can validated they have the appropriate rights)

HOWEVER again you run the IE browser from another machine and then the IIS requests the IPC it fails with the error below because the Token is just an impersonation token goddam there must be a way around this....? I need to pass the credentials to the Service but it must be impersonated credentials that get passed.

Error

```
System.Runtime.Remoting.RemotingException: Failed to connect to an IPC
Port: Access is denied. Server stack trace: at
System.Runtime.Remoting.Channels.Ipc.IpcPort.Connect(String portName,
Boolean secure, TokenImpersonationLevel impersonationLevel, Int32
timeout) at
System.Runtime.Remoting.Channels.Ipc.ConnectionCache.GetConnection(String
portName, Boolean secure, TokenImpersonationLevel level, Int32
timeout) at
System.Runtime.Remoting.Channels.Ipc.IpcClientTransportSink.ProcessMessage(IMessage
msg, ITransportHeaders requestHeaders, Stream requestStream,
ITransportHeaders& responseHeaders, Stream& responseStream) at
System.Runtime.Remoting.Channels.BinaryClientFormatterSink.SyncProcessMessage(IMessage
msg) Exception rethrown at [0]: at
System.Runtime.Remoting.Proxies.RealProxy.HandleReturnMessage(IMessage
reqMsg, IMessage retMsg) at
System.Runtime.Remoting.Proxies.RealProxy.PrivateInvoke(MessageData&
msgData, Int32 type) at
ProvisionController.ProvisionController.HelloWorld() at
_Default.Button1_Click(Object sender, EventArgs e)
```

' CODE FOR SERVICE

```
Dim oChannelProps As New Dictionary(Of String, String)

oChannelProps.Add("authorizedGroup", "Everyone")

oChannelProps.Add("portName", "hello")

Dim serverChannel As New IpcServerChannel(oChannelProps, Nothing)

ChannelServices.RegisterChannel(serverChannel, True)

RemotingConfiguration.RegisterWellKnownServiceType(GetType(ProvisionController.ProvisionController),
"HelloWorld", WellKnownObjectMode.Singleton)

System.Diagnostics.EventLog.WriteEntry("VBService", "Server Listening
on: " & serverChannel.GetChannelUri & " as: " &
System.Security.Principal.WindowsIdentity.GetCurrent.Name)
```

Can it work...? – ASP.NET Impersonation with Remoting

' CODE FOR CLIENT (ASPX Page)

```
Dim ClientChannel As New IpcClientChannel("MyChannel", Nothing)
```

```
ChannelServices.RegisterChannel(ClientChannel, True)
```

```
RemotingConfiguration.RegisterWellKnownClientType(GetType(ProvisionController.ProvisionController),  
"ipc://hello/helloworld")
```

```
oProvisionControl = New ProvisionController.ProvisionController
```

```
Me.Label1.Text = oProvisionControl.HelloWorld
```

I am using the new Flow impersonation policy for .NET 2 in the ASPNET.CONFIG file

```
<legacyImpersonationPolicy enabled="false"/>
```

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

Thanks,

Dave

.