

# Re: Problem with SslStream when using Windows Vista

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework/2007-07/msg00199.html>

---

- *From:* "Dave" <[KingOfTheBeach@xxxxxxxxxxxxxxxxxxxxx](mailto:KingOfTheBeach@xxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Tue, 10 Jul 2007 08:17:31 -0500
- 

Here is a sample that someone else posted and I had a link to in my first post:

—

Exception: A call to SSPI failed, see inner exception.  
Inner exception: The Local Security Authority cannot be contacted  
Authentication failed – closing the connection.

I've reviewed tracing information and it doesn't really help. What seems to occur is an internal error and then the client side connection is shutdown.

System.Net Information: 0 : [3804] AcceptSecurityContext(In-Buffer length=1333, Out-Buffer length=0, returned code=InternalError).

Anyone with any hints it would be much appreciated.

The code for the server & client follow.

```
--- server.cs ---
using System;
using System.Collections;
using System.Net;
using System.Net.Sockets;
using System.Net.Security;
using System.Security.Authentication;
using System.Text;
using System.Security.Cryptography.X509Certificates;
using System.IO;
using System.Threading;

namespace Examples.System.Net
{
    public class SslTcpServer
    {
        private X509Certificate _serverCertificate = null;
    }
}
```

## Re: Problem with SslStream when using Windows Vista

```
private Thread _thread;
private int _port = 0;

// The certificate parameter specifies the name of the file
// containing the machine certificate.
void RunServer()
{
    // Create a TCP/IP (IPv4) socket and listen for incoming
    // connections.
    TcpListener listener = new TcpListener(IPAddress.Any, 0);
    listener.Start();
    lock(this)
    {
        _port = ((IPEndPoint)listener.LocalEndpoint).Port;
        Monitor.Pulse(this);
    }
    Console.WriteLine("Waiting for a client to connect on port " +
        _port);
    TcpClient client = listener.AcceptTcpClient();
    ProcessClient(client);
    listener.Stop();
}

void ProcessClient(TcpClient client)
{
    // A client has connected. Create the
    // SslStream using the client's network stream.
    SslStream sslStream = new SslStream(client.GetStream(), false);
    try
    {
        sslStream.AuthenticateAsServer(
            _serverCertificate, false, SslProtocols.Tls, true);

        // Set timeouts for the read and write to 5 seconds.
        sslStream.ReadTimeout = 5000;
        sslStream.WriteTimeout = 5000;
        // Read a message from the client.
        Console.WriteLine("Waiting for client message...");
        string messageData = ReadMessage(sslStream);
        Console.WriteLine("Received: {0}", messageData);

        // Write a message to the client.
        byte[] message = Encoding.UTF8.GetBytes("Hello from the
        server.<EOF>");
        Console.WriteLine("Sending hello message.");
        sslStream.Write(message);
    }
    catch (AuthenticationException e)
    {
        Console.WriteLine("Exception: {0}", e.Message);
        if (e.InnerException != null)

```

## Re: Problem with SslStream when using Windows Vista

```
{
Console.WriteLine("Inner exception: {0}",
e.InnerException.Message);
}
Console.WriteLine ("Authentication failed – closing the
connection.");
sslStream.Close();
client.Close();
}
finally
{
// The client stream will be closed with the sslStream
// because we specified this behavior when creating
// the sslStream.
sslStream.Close();
client.Close();
}
}

string ReadMessage(SslStream sslStream)
{
// Read the message sent by the client.
// The client signals the end of the message using the
// "<EOF>" marker.
byte [] buffer = new byte[2048];
StringBuilder messageData = new StringBuilder();
int bytes = -1;
do
{
// Read the client's test message.
bytes = sslStream.Read(buffer, 0, buffer.Length);

// Use Decoder class to convert from bytes to UTF8
// in case a character spans two buffers.
Decoder decoder = Encoding.UTF8.GetDecoder();
char[] chars = new
char[decoder.GetCharCount(buffer,0,bytes)];
decoder.GetChars(buffer, 0, bytes, chars,0);
messageData.Append (chars);
// Check for EOF or an empty message.
if (messageData.ToString().IndexOf("<EOF>") != -1)
{
break;
}
} while (bytes !=0);

return messageData.ToString();
}

public SslTcpServer()
{
```

## Re: Problem with SslStream when using Windows Vista

```
_serverCertificate = new
X509Certificate2("s_rsa_nopass_ca1.pfx", "password");
_thread = new Thread(new ThreadStart(RunServer));
_thread.Name = "SslTcpServer";
_thread.Start();
}

public int GetPort()
{
lock(this)
{
while(_port == 0)
{
Monitor.Wait(this);
}
}
return _port;
}
}

public class FactoryServer
{
void ProcessClient(TcpClient client)
{
try
{
SslTcpServer server = new SslTcpServer();
int port = server.GetPort();
NetworkStream stream = client.GetStream();
byte[] msg = Encoding.UTF8.GetBytes(port.ToString());
stream.Write(msg, 0, msg.Length);
byte[] term = new byte[1];
term[0] = 0;
stream.Write(term, 0, 1);
stream.Flush();
int bytes = stream.Read(term, 0, 1);
}
finally
{
client.Close();
}
}

public void Run()
{
TcpListener listener = new TcpListener(IPAddress.Any, 8080);
listener.Start();
while (true)
{
Console.WriteLine("Waiting for a client to connect...");
// Application blocks while waiting for an incoming
```

## Re: Problem with SslStream when using Windows Vista

```
connection.
// Type CNTL-C to terminate the server.
TcpClient client = listener.AcceptTcpClient();
ProcessClient(client);
}
}

}

public class Server
{
public static int Main(string[] args)
{
FactoryServer server = new FactoryServer();
server.Run();
return 0;
}
}
}

-- client.cs --
using System;
using System.Collections;
using System.Net;
using System.Net.Security;
using System.Net.Sockets;
using System.Security.Authentication;
using System.Text;
using System.Security.Cryptography.X509Certificates;
using System.IO;
using System.Threading;

namespace Examples.System.Net
{
public class SslTcpClient
{
private static Hashtable certificateErrors = new Hashtable();

public static bool ValidateServerCertificate(
object sender,
X509Certificate certificate,
X509Chain chain,
SslPolicyErrors sslPolicyErrors)
{
return true;
}

public static void RunClient(int port)
{
TcpClient client = new TcpClient("127.0.0.1", port);
NetworkStream stream = client.GetStream();
Console.WriteLine("Client connected.");
```

## Re: Problem with SslStream when using Windows Vista

```
// Create an SSL stream that will close the client's stream.
SslStream sslStream = new SslStream(
stream, false, new RemoteCertificateValidationCallback(
ValidateServerCertificate), null);
try
{
//sslStream.AuthenticateAsClient("localhost", null,
SslProtocols.Default, false);
sslStream.AuthenticateAsClient("localhost");
}
catch (AuthenticationException e)
{
Console.WriteLine("Exception: {0}", e.Message);
if (e.InnerException != null)
{
Console.WriteLine("Inner exception: {0}",
e.InnerException.Message);
}
Console.WriteLine ("Authentication failed – closing the
connection.");
client.Close();
return;
}
// Encode a test message into a byte array.
// Signal the end of the message using the "<EOF>".
byte[] message = Encoding.UTF8.GetBytes("Hello from the
client.<EOF>");
// Send hello message to the server.
sslStream.Write(message);
sslStream.Flush();
// Read message from the server.
string serverMessage = ReadMessage(sslStream);
Console.WriteLine("Server says: {0}", serverMessage);
// Close the client connection.
client.Close();
Console.WriteLine("Client closed.");
}
static string ReadMessage(SslStream sslStream)
{
// Read the message sent by the server.
// The end of the message is signaled using the
// "<EOF>" marker.
byte [] buffer = new byte[2048];
StringBuilder messageData = new StringBuilder();
int bytes = -1;
do
{
bytes = sslStream.Read(buffer, 0, buffer.Length);

// Use Decoder class to convert from bytes to UTF8
// in case a character spans two buffers.
```

## Re: Problem with SslStream when using Windows Vista

```
Decoder decoder = Encoding.UTF8.GetDecoder();
char[] chars = new
char[decoder.GetCharCount(buffer,0,bytes)];
decoder.GetChars(buffer, 0, bytes, chars,0);
messageData.Append (chars);
// Check for EOF.
if (messageData.ToString().IndexOf("<EOF>") != -1)
{
break;
}
} while (bytes != 0);

return messageData.ToString();
}

private static int ReadPort(NetworkStream stream)
{
byte[] msg = new byte[1024];
int curr = 0;
do
{
int bytes = stream.Read(msg, curr, 1);
if(bytes == 0)
{
Console.WriteLine("Error reading port");
return 1;
}
if(msg[curr] == 0)
{
break;
}
++curr;
}
while(curr < msg.Length);
if(curr == msg.Length)
{
Console.WriteLine("Error reading port");
return 1;
}

Decoder decoder = Encoding.UTF8.GetDecoder();
char[] chars = new
char[decoder.GetCharCount(msg,0,msg.Length-1)];
decoder.GetChars(msg, 0, msg.Length-1, chars,0);
StringBuilder messageData = new StringBuilder();
messageData.Append(chars);
return Int32.Parse(messageData.ToString());
}

public static int Main(string[] args)
{
```

Re: Problem with SslStream when using Windows Vista

```
TcpClient client = new TcpClient("127.0.0.1", 8080);
NetworkStream stream = client.GetStream();
Console.WriteLine("Client connected.");
int port = ReadPort(stream);
Console.WriteLine("Connecting to " + port);

// Write termination byte drop the client.
byte[] term = new byte[1];
term[0] = 0;
stream.Write(term, 0, 1);
client.Close();

SslTcpClient.RunClient(port);
return 0;
}
}
}
```

""Jeffrey Tan[MSFT]"" <jetan@xxxxxxxxxxxxxxxxxxxxxx> wrote in message  
[news:TamELSrwwHA.4484@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:TamELSrwwHA.4484@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

Hi Dave,

The session resumption can be turned off by setting the maximum cache size  
\ time to a very small value. But this is a horrible workaround (not  
advised)  
<http://msdn2.microsoft.com/en-us/library/aa922895.aspx>

To help to find the root cause, is it possible for you to provide a little  
sample project to demonstrate this problem? Once I can give it a local  
reproduce, it would be easier to troubleshoot it and find a solution.

Thanks.

Best regards,  
Jeffrey Tan  
Microsoft Online Community Support

=====  
Get notification to my posts through email? Please refer to  
<http://msdn.microsoft.com/subscriptions/managednewsgroups/default.aspx#notif>  
ications.

Note: The MSDN Managed Newsgroup support offering is for non-urgent issues  
where an initial response from the community or a Microsoft Support  
Engineer within 1 business day is acceptable. Please note that each follow  
up response may take approximately 2 business days as the support  
professional working with you may need further investigation to reach the  
most efficient resolution. The offering is not appropriate for situations  
that require urgent, real-time or phone-based interactions or complex

Re: Problem with SslStream when using Windows Vista

project analysis and dump analysis issues. Issues of this nature are best handled working with a dedicated Microsoft Support Engineer by contacting Microsoft Customer Support Services (CSS) at <http://msdn.microsoft.com/subscriptions/support/default.aspx>.

=====  
This posting is provided "AS IS" with no warranties, and confers no rights.