

Re: HttpRequest.GetRequestStream times out

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.performance/2006-06/msg00019.htm>

- *From:* "Christer Brinchmsnn" <christerbrinchmann@xxxxxxxxxxx>
 - *Date:* Tue, 13 Jun 2006 18:34:34 +0200
-

"Joerg Jooss" <news-reply@xxxxxxxxxxx> wrote in message
news:94fc5071328f68c858a01d1affb8@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

Thus wrote Christer,

Hello.

Hope this is the right group to post to. If not, please let me know where to post.

I'm having a problem doing an HTTP POST to my payment provider during load. During days with low load, no errors occur.

The code being executed is in a class library loaded by an ASP.NET application.

The server machine is a 2003 Server Std Ed. w/SP1, Intel Xeon 2.8 GHz with 2.5 GB RAM.

.NET version is 1.1. The mscorlib.dll has version 1.1.4322.2300, I take it that it means .NET SP1...

When I encounter this exception, this is printed in the log:

Payment Authorization failed (WebException): Timeout. Message: The operation has timed-out.. Response: . Stack Trace: at System.Net.HttpWebRequest.GetRequestStream()

at Payment.DoPaymentAuth()

I have tried to configure the machine.config file according to the recommendations at MSDN:

- Set maxConnection to 48 (12 * 4)
 - Set maxWorkerThreads to 100
 - Set minFreeThreads to 352 (88* 4)
- How can I investigate this further?

I hope someone can help me with this.

Re: HttpWebRequest.GetRequestStream times out

Here is the code:

```
<code>

try
{
//The postString has value "cardno=XXX&expmon=06&expyear=06" etc.
byte[] sendData = Encoding.ASCII.GetBytes(postString);
int sendLength = sendData.Length;
HttpWebRequest httpReq = (HttpWebRequest)
WebRequest.Create(PAYMENT_AUTH_URI);

httpReq.Method = "POST";
httpReq.ContentType="application/x-www-form-urlencoded";
httpReq.ContentLength = sendLength;
httpReq.ConnectionGroupName = "PaymentGroup";

//Error comes here...
Stream sendStream = httpReq.GetRequestStream();
sendStream.Write(sendData,0,sendLength);
sendStream.Flush();
sendStream.Close();
sendStream = null;
WebResponse resp = httpReq.GetResponse();
Stream respstrm = resp.GetResponseStream();
int length = 1024;
byte[] Buffer = new byte[length];
int bytesRead = 0;
bytesRead = respstrm.Read(Buffer,0,length);
if(bytesRead > 0 )
{
postReturn = Encoding.ASCII.GetString(Buffer,0, bytesRead);
}
try
{
respstrm.Flush();
respstrm.Close();
respstrm = null;
}
catch(Exception e)
{
respstrm = null;
}
try
{
resp.Close();
resp = null;
}
catch(Exception e)
{

```

Re: HttpWebRequest.GetRequestStream times out

```
resp = null;
}
}
catch(System.Net.WebException we)
{
LogError("Order Id " + this.OrderId + ": Payment Authorization
failed
(WebException)(" +
DateTime.Now.ToString(dateformat) + "): " + we.Status.ToString()
+ ".
Message: " +
we.Message + ". Response: " + we.Response + ". Stack Trace: " +
we.StackTrace);
if(we.Status == WebExceptionStatus.Timeout)
{
postReturn = "Timeout";
}
else
{
postReturn = "Error";
}
}
catch(Exception ex)
{
postReturn = "Error";
}
return postReturn;

</code>
```

Thanks!

Save for the somewhat scary exception handling (sorry!) and the fact that you never read more than 1 kB of the response, the code should work. The question is: What times out? Writing to the request stream? Obtaining the response? Reading from the response stream? Can you recreate the failure during a load test?

Cheers,

--

Joerg Jooss
news-reply@xxxxxxxxxxxxxx

Hi Joerg, thanks for your reply.

I know... the code could be way better.

The reason I haven't stressed a rewrite of the code is that this error never occurs in our test environment. I use MS ACT to test, and 200 simultaneous

Re: HttpWebRequest.GetRequestStream times out

Re: HttpWebRequest.GetRequestStream times out

users never produce this exception.

I am able to reproduce the exception in a load test in the production environment. It happens with only 50 simultaneous users. (Also, the response is never larger than about 500 bytes, so there is no need to use a larger buffer.) I even tried testing code that does everything inside try/finally blocks, closing and disposing anything that has a Close() or Dispose() method. Same result.

It is the call to GetRequestStream() that times out. All I get in the stack trace is what I posted earlier.

The request timeout is set to 100 seconds, and the requests that time out throw this exception exactly 100 seconds after the call to GetRequestStream() was issued. Requests just before and after usually complete within a second.

The hardware running the test solution is way smaller than the prod solution, and the IIS even hosts 10–15 webs. It cannot be a hardware issue.

What is actually happening when you call GetRequestStream()? Is a DNS lookup performed, or is that done earlier? I am starting to suspect that the network is a bottle neck. But, the NIC on the machine is a 1 gbps card.

I was thinking I should try editing the hosts file to maybe avoid doing a DNS resolve, but I haven't gotten around to it yet.

Any suggestions?

I hope you can help me some more with this.

Thanks!

C.

.