

Re: The message must contain a wsa:To header

Re: The message must contain a wsa:To header

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.webservices.enhancements/2006-08/>

- *From:* "Pablo Cibraro" <pcibraro@xxxxxxxxxxxx>
 - *Date:* Tue, 18 Jul 2006 10:12:44 -0400
-

mmm, that is strange. If a trace file is not being generated, perhaps WSE is not running at all.

No, WSDL does not say anything about the WS-* stack, that's why the WS-Policy spec was designed. However, only WCF implements that spec for the moment.

Regards,
Pablo Cibraro.

"Chris Fink" <ChrisFink@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message news:6A45BFC1-8903-4929-B480-F333E4C45F95@xxxxxxxxxxxxxxxxxxxx

My client app is not generating a trace file. In fact, I have been able to create a service and a test client that consumes this service using WSE2 SP3 and client side certs and it works fine, but still no trace file in either app. The web.config looks fine to me, and permission wise their should be no problem. The event log is also empty. So I need to divert and fix that problem before I can see if the header is missing in the trace....

```
... <microsoft.web.services2>
<security>
<x509 storeLocation="CurrentUser" allowTestRoot="true" />
</security>
<diagnostics>
<trace enabled="true" input="InputTrace.webinfo"
output="OutputTrace.webinfo" />
<policyTrace enabled="true" input="ReceivePolicy.webinfo"
output="SendPolicy.webinfo" />
</diagnostics>
</policy>
<cache name="policyCache.config" />
</policy>
</microsoft.web.services2>
</configuration>
```

Re: The message must contain a wsa:To header

I would assume that my problem with the missing wsa header is that the client is not setup the same way as the service (response/response signing/encryption). I'll need to talk to the customer to understand what options I should select while going through the WSE 2 wizard.

Off topic, but shouldn't all this information be part of the web service contract (WSDL)? Be nice if I could just read my customer's WSDL to get all this information.

"Pablo Cibraro" wrote:

Hi Chris,

Did you check if the wsa:To header is being sent to the server ? (Using the trace file).

Maybe, the client is not applying the WSE policy at all because of an difference in the mapping.

Regards,

Pablo Cibraro

<http://weblogs.asp.net/cibrax>

[MVP – Connected Systems Developer]

"Chris Fink" <ChrisFink@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message news:3D0B2021-2FFB-4D24-B567-930580AC74C3@xxxxxxxxxxxxxxxxxxxx

I am using WSE 2 SP3 and receiving the following error from my client calling a secure service using client certificates. When I setup the policy, I required signing and encryption for the request and response.

```
System.Web.Services.Protocols.SoapException: Server was
unable to
process
request. ----> The message must contain a wsa:To header.
at
System.Web.Services.Protocols.SoapHttpClientProtocol.ReadResponse(SoapClientMessage
message, WebResponse response, Stream responseStream,
Boolean
asyncCall)
at
System.Web.Services.Protocols.SoapHttpClientProtocol.Invoke(String
methodName, Object[] parameters)
at ApplicationMessagingWS.Dispatch(String messageType,
String
```

Re: The message must contain a wsa:To header

correlationId, String messageBody, String userName, String
applicationName,
String instance, String postBackUrl)
at DellWSE2SP3.ConsumeDellMSS.Dispatch(String
messageType, String
correlationId, String messageBody, String userName, String
applicationName,
String instance, String postBackUrl)

This is the policy file

```
<?xml version="1.0" encoding="utf-8"?>
<policyDocument
xmlns="http://schemas.microsoft.com/wse/2003/06/Policy">
  <mappings
xmlns:wse="http://schemas.microsoft.com/wse/2003/06/Policy">
    <!--The following policy describes the policy requirements
for the
service:
http://xxx/mss/webservices/ApplicationMessagingWS.asmx
.-->
    <endpoint
uri="http://xxx/mss/webservices/ApplicationMessagingWS.asmx">
      <defaultOperation>
        <request policy="#Sign-X.509-Encrypt-X.509" />
        <response policy="#Sign-X.509-Encrypt-X.509-1" />
        <fault policy="" />
      </defaultOperation>
    </endpoint>
  </mappings>
  <policies
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2002/12/policy"
xmlns:wssp="http://schemas.xmlsoap.org/ws/2002/12/secext"
xmlns:wse="http://schemas.microsoft.com/wse/2003/06/Policy"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing">
    <wsp:Policy wsu:Id="Sign-X.509-Encrypt-X.509">
      <!--MessagePredicate is used to require headers. This
assertion
should
be used along with the Integrity assertion when the presence
of the
signed
element is required. NOTE: this assertion does not do
anything for
enforcement (send-side) policy.-->
      <wsp:MessagePredicate wsp:Usage="wsp:Required"
Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part">wsp:Body()
wsp:Header(wsa:To) wsp:Header(wsa:Action)
wsp:Header(wsa:MessageID)
wse:Timestamp()</wsp:MessagePredicate>
```

Re: The message must contain a wsa:To header

Re: The message must contain a wsa:To header

<!--The Integrity assertion is used to ensure that the message is signed with X.509. Many Web services will also use the token for authorization, such as by using the <wse:Role> claim or specific X.509 claims.-->

<wssp:Integrity wssp:Usage="wssp:Required">

<wssp:TokenInfo>

<!--The SecurityToken element within the TokenInfo element

describes which token type must be used for Signing.-->

<wssp:SecurityToken>

<wssp:TokenType>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-

<wssp:TokenIssuer>CN=Root Agency</wssp:TokenIssuer>

<wssp:Claims>

<!--By specifying the SubjectName claim, the policy system

can

look for a certificate with this subject name in the certificate store

indicated in the application's configuration, such as

LocalMachine or

CurrentUser. The WSE X.509 Certificate Tool is useful for finding the

correct

values for this field.-->

<wssp:SubjectName MatchType="wssp:Exact">CN=

HBTClient</wssp:SubjectName>

<wssp:X509Extension OID="2.5.29.14"

MatchType="wssp:Exact">u1IEv47jqXyrb0gujx/GRPFUrw=</wssp:X509Extension>

</wssp:Claims>

</wssp:SecurityToken>

</wssp:TokenInfo>

<wssp:MessageParts

Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part>wssp:Body()

wssp:Header(wsa:Action) wssp:Header(wsa:FaultTo)

wssp:Header(wsa:From)

wssp:Header(wsa:MessageID) wssp:Header(wsa:RelatesTo)

wssp:Header(wsa:ReplyTo)

wssp:Header(wsa:To)

wse:Timestamp()</wssp:MessageParts>

</wssp:Integrity>

<!--The Confidentiality assertion is used to ensure that the SOAP

Body

is encrypted.-->

<wssp:Confidentiality wssp:Usage="wssp:Required">

<wssp:KeyInfo>

<!--The SecurityToken element within the KeyInfo element

Re: The message must contain a wsa:To header

Re: The message must contain a wsa:To header

describes

which token type must be used for Encryption.-->

<wssp:SecurityToken>

<wssp:TokenType>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-

<wssp:TokenIssuer>O= Inc., CN= Inc. Enterprise Utility

CA1</wssp:TokenIssuer>

<wssp:Claims>

<!--By specifying the SubjectName claim, the policy

system

can

look for a certificate with this subject name in the certificate

store

indicated in the application's configuration, such as

LocalMachine or

CurrentUser. The WSE X.509 Certificate Tool is useful for

finding the

correct

values for this field.-->

<wssp:SubjectName MatchType="wssp:Exact">C=US,

S=TX,

L=Austin,

O= Inc., OU=Information Technology, CN=MSS Spore,

E=webfarm@

.com</wssp:SubjectName>

<wssp:X509Extension OID="2.5.29.14"

MatchType="wssp:Exact">rrRD87efOO5bpHFLxT+psuYqMKM=</wssp:X509Extension>

</wssp:Claims>

</wssp:SecurityToken>

</wssp:KeyInfo>

<wssp:MessageParts

Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part>wsp:Body()</wssp:MessageParts>

</wssp:Confidentiality>

</wsp:Policy>

<wsp:Policy wsu:Id="Sign-X.509-Encrypt-X.509-1">

<!--MessagePredicate is used to require headers. This

assertion

should

be used along with the Integrity assertion when the presence

of the

signed

element is required. NOTE: this assertion does not do

anything for

enforcement (send-side) policy.-->

<wsp:MessagePredicate wsp:Usage="wsp:Required"

Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part>wsp:Body()

wsp:Header(wsa:To) wsp:Header(wsa:Action)

wsp:Header(wsa:MessageID)

wse:Timestamp()</wsp:MessagePredicate>

<!--The Integrity assertion is used to ensure that the

message is

Re: The message must contain a wsa:To header

Re: The message must contain a wsa:To header

signed with X.509. Many Web services will also use the token for authorization, such as by using the <wse:Role> claim or specific X.509 claims.-->

<wssp:Integrity wssp:Usage="wssp:Required">

<wssp:TokenInfo>

<!--The SecurityToken element within the TokenInfo element

describes which token type must be used for Signing.-->

<wssp:SecurityToken>

<wssp:TokenType>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-

<wssp:TokenIssuer>O= Inc., CN= Inc. Enterprise Utility

CA1</wssp:TokenIssuer>

<wssp:Claims>

<!--By specifying the SubjectName claim, the policy system

can

look for a certificate with this subject name in the certificate store

indicated in the application's configuration, such as

LocalMachine or

CurrentUser. The WSE X.509 Certificate Tool is useful for

finding the

correct

values for this field.-->

<wssp:SubjectName MatchType="wssp:Exact">C=US,

S=TX,

L=Austin,

O= Inc., OU=Information Technology, CN=MSS Spore,

E=webfarm@

.com</wssp:SubjectName>

<wssp:X509Extension OID="2.5.29.14"

MatchType="wssp:Exact">rrRD87efOO5bpHFLxT+psuYqMKM=</wssp:X509Extension>

</wssp:Claims>

</wssp:SecurityToken>

</wssp:TokenInfo>

<wssp:MessageParts

Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part>wssp:Body()

wssp:Header(wsa:Action) wssp:Header(wsa:FaultTo)

wssp:Header(wsa:From)

wssp:Header(wsa:MessageID) wssp:Header(wsa:RelatesTo)

wssp:Header(wsa:ReplyTo)

wssp:Header(wsa:To)

wse:Timestamp()</wssp:MessageParts>

</wssp:Integrity>

<!--The Confidentiality assertion is used to ensure that the SOAP

Body

is encrypted.-->

Re: The message must contain a wsa:To header

Re: The message must contain a wsa:To header

<wssp:Confidentiality wssp:Usage="wssp:Required">
<wssp:KeyInfo>
<!--The SecurityToken element within the KeyInfo element
describes
which token type must be used for Encryption.-->
<wssp:SecurityToken>

<wssp:TokenType>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-
<wssp:TokenIssuer>CN=Root Agency</wssp:TokenIssuer>
<wssp:Claims>
<!--By specifying the SubjectName claim, the policy
system
can
look for a certificate with this subject name in the certificate
store
indicated in the application's configuration, such as
LocalMachine or
CurrentUser. The WSE X.509 Certificate Tool is useful for
finding the
correct
values for this field.-->
<wssp:SubjectName MatchType="wssp:Exact">CN=
HBTClient</wssp:SubjectName>
<wssp:X509Extension OID="2.5.29.14"
MatchType="wssp:Exact">u1IEv47jqXyrb0gujx/GRPFUjw=</wssp:X509Extension>
</wssp:Claims>
</wssp:SecurityToken>
</wssp:KeyInfo>
<wssp:MessageParts
Dialect="http://schemas.xmlsoap.org/2002/12/wsse#part>wssp:Body()</wssp:MessageParts>
</wssp:Confidentiality>
</wssp:Policy>
</policies>
</policyDocument>

and the web.config
<?xml version="1.0" encoding="utf-8"?>
<configuration>
<configSections>
<section name="microsoft.web.services2"
type="Microsoft.Web.Services2.Configuration.WebServicesConfiguration,
Microsoft.Web.Services2, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" />
</configSections>
<system.web>
<!-- DYNAMIC DEBUG COMPILATION
Set compilation debug="true" to enable ASPX debugging.
Otherwise,
setting this value to
false will improve runtime performance of this application.
Set compilation debug="true" to insert debugging symbols

Re: The message must contain a wsa:To header

Re: The message must contain a wsa:To header

(.pdb information) into the compiled page. Because this creates a larger file that executes more slowly, you should set this value to true only when debugging and to false at all other times. For more information, refer to the documentation about debugging ASP.NET files.

-->
<compilation defaultLanguage="c#" debug="true" />
<!-- CUSTOM ERROR MESSAGES
Set customErrors mode="On" or "RemoteOnly" to enable custom error messages, "Off" to disable.
Add <error> tags for each of the errors you want to handle.

"On" Always display custom (friendly) messages.
"Off" Always display detailed ASP.NET error information.
"RemoteOnly" Display custom (friendly) messages only to users not running on the local Web server. This setting is recommended for security purposes, so that you do not display application detail information to remote clients.

-->
<customErrors mode="RemoteOnly" />
<!-- AUTHENTICATION
This section sets the authentication policies of the application.
Possible modes are "Windows", "Forms", "Passport" and "None"

"None" No authentication is performed.
"Windows" IIS performs authentication (Basic, Digest, or Integrated Windows) according to its settings for the application. Anonymous access must be disabled in IIS.
"Forms" You provide a custom form (Web page) for users to enter their credentials, and then you authenticate them in your application. A user credential token is stored in a cookie.
"Passport" Authentication is performed via a centralized

Re: The message must contain a wsa:To header

authentication service provided
by Microsoft that offers a single logon and core profile
services
for member sites.

-->

<authentication mode="Windows" />

<!-- AUTHORIZATION

This section sets the authorization policies of the
application.

You can allow or deny access
to application resources by user or role. Wildcards: "*" mean
everyone. "?" means anonymous
(unauthenticated) users.

-->

<authorization>

<allow users="*" />

<!-- Allow all users -->

<!-- <allow users="[comma separated list of users]"

roles="[comma separated list of roles]"/>

<deny users="[comma separated list of users]"

roles="[comma separated list of roles]"/>

-->

</authorization>

<!-- APPLICATION-LEVEL TRACE LOGGING

Application-level tracing enables trace log output for every
page

within an application.

Set trace enabled="true" to enable application trace logging.

If