WebDAV − Permissions on message with attachments

Source:
http://www.tech−archive.net/Archive/Exchange/microsoft.public.exchange.applications/2006−01/msg00015.html

- *From*: scanlonmeister@xxxxxxxxx
- *Date*: 16 Jan 2006 06:13:24 −0800

What permissions do you need to set on an exchange message item to be able to use the
X−MS−ENUMATTS WebDAV method.

I am using code supplied directly from MSDN web site and I always get a 403 Error returned in the response object. I have assumed that this should be to do with not having access to the attachment of the message. I can run all other queries ok. If I query the attachment−filename property I get a similar error in the HttpResponse. I am wondering what it is that is stopping me obtaining attachments from Outlook Web Access.

If you have any pointers − or if any of you have seen this situation before then I would be very grateful. The code I am using is below.. as I say, I always get a 403 from the response object and I am unable to find out why.

Can anybody can tell me what permissions I need to set on the message and how I set them with WebDAV − possibly a sample?

```
[STAThread]
static void Main(string[] args)
{
// Variables.
System.Net.HttpWebRequest Request;
System.Net.WebResponse Response;
System.Net.CredentialCache MyCredentialCache;
string strMessageURI =
"http://server/exchange/username/inbox/TestMessage.eml/;
string strUserName = "username";
string strPassword = "!Password";
string strDomain = "Domain";
System.IO.Stream ResponseStream = null;
```

```
System.Xml.XmlDocument ResponseXmlDoc = null;
System.Xml.XmlNode root = null;
System.Xml.XmlNamespaceManager nsmgr = null;
System.Xml.XmlNodeList PropstatNodes = null;
System.Xml.XmlNodeList HrefNodes = null;
System.Xml.XmlNode StatusNode = null;
System.Xml.XmlNode PropNode = null;

try
{
// Create a new CredentialCache object and fill it with the
network
// credentials required to access the server.
MyCredentialCache = new System.Net.CredentialCache();
MyCredentialCache.Add( new System.Uri(strMessageURI),
"NTLM",
new System.Net.NetworkCredential(strUserName,
strPassword, strDomain)
);

// Create the HttpWebRequest object.
Request =
(System.Net.HttpWebRequest)HttpWebRequest.Create(strMessageURI);

// Add the network credentials to the request.
Request.Credentials = MyCredentialCache;

// Specify the method.
Request.Method = "X−MS−ENUMATTS";

// Send the X−MS−ENUMATTS method request and get the
// response from the server.
Response = (HttpWebResponse)Request.GetResponse();

// Get the XML response stream.
ResponseStream = Response.GetResponseStream();

// Create the XmlDocument object from the XML response
stream.
ResponseXmlDoc = new System.Xml.XmlDocument();

// Load the XML response stream.
ResponseXmlDoc.Load(ResponseStream);

// Get the root node.
root = ResponseXmlDoc.DocumentElement;

// Create a new XmlNamespaceManager.
nsmgr = new
System.Xml.XmlNamespaceManager(ResponseXmlDoc.NameTable);
```

```
// Add the DAV: namespace, which is typically assigned the
a: prefix
// in the XML response body. The namespaceses and their
associated
// prefixes are listed in the attributes of the
DAV:multistatus node
// of the XML response.
nsmgr.AddNamespace("a", "DAV:");

// Add the http://schemas.microsoft.com/mapi/proptag/
namespace, which
// is typically assigned the d: prefix in the XML response
body.
nsmgr.AddNamespace("d",
"http://schemas.microsoft.com/mapi/proptag/);

// Use an XPath query to build a list of the DAV:propstat
XML nodes,
// corresponding to the returned status and properties of
// the file attachment(s).
PropstatNodes = root.SelectNodes("//a:propstat", nsmgr);

// Use an XPath query to build a list of the DAV:href
nodes,
// corresponding to the URIs of the attachement(s) on the
message.
// For each DAV:href node in the XML response, there is an
// associated DAV:propstat node.
HrefNodes = root.SelectNodes("//a:href", nsmgr);

// Attachments found?
if(HrefNodes.Count > 0)
{
// Display the number of attachments on the message.
Console.WriteLine(HrefNodes.Count + " attachments
found...");

// Iterate through the attachment properties.
for(int i=0;i<HrefNodes.Count;i++)
{
// Use an XPath query to get the DAV:status node from
the DAV:propstat node.
StatusNode =
PropstatNodes[i].SelectSingleNode("a:status", nsmgr);

// Check the status of the attachment properties.
if(StatusNode.InnerText != "HTTP/1.1 200 OK")
{
Console.WriteLine("Attachment: " +
HrefNodes[i].InnerText);
Console.WriteLine("Status: " +
```

```
StatusNode.InnerText);
Console.WriteLine("");
}
else
{
Console.WriteLine("Attachment: " +
HrefNodes[i].InnerText);
Console.WriteLine("Status: " +
StatusNode.InnerText);

// Get the CdoPR_ATTACH_FILENAME_W MAPI property
tag,
// corresponding to the attachment file name. The
// http://schemas.microsoft.com/mapi/proptag/
namespace is typically
// assigned the d: prefix in the XML response
body.
PropNode =
PropstatNodes[i].SelectSingleNode("a:prop/d:x3704001f", nsmgr);
Console.WriteLine("Attachment name: " +
PropNode.InnerText);

// Get the CdoPR_ATTACH_EXTENSION_W MAPI property
tag,
// corresponding to the attachment file extension.
PropNode =
PropstatNodes[i].SelectSingleNode("a:prop/d:x3703001f", nsmgr);
Console.WriteLine("File extension: " +
PropNode.InnerText);

// Get the CdoPR_ATTACH_SIZE MAPI property tag,
// corresponding to the attachment file size.
PropNode =
PropstatNodes[i].SelectSingleNode("a:prop/d:x0e200003", nsmgr);
Console.WriteLine("Attachment size: " +
PropNode.InnerText);

Console.WriteLine("");
}
}
}
else
{
Console.WriteLine("No attachments found.");
}

// Clean up.
ResponseStream.Close();
Response.Close();

}
```

```
catch(Exception ex)
{
// Catch any exceptions. Any error codes from the
X−MS−ENUMATTS
// method request on the server will be caught here, also.
Console.WriteLine(ex.Message);
}
```

Best Regards,

James Scanlon

.

- Prev by Date: ***Ressource management application***
- Next by Date: ***Re: Ressource management application***
- Previous by thread: ***Ressource management application***
- Next by thread: ***Public folder automation question***
- Index(es):
    - ***Date***
    - ***Thread***