

Re: Getting seteuclid/setegid functionality out of Windows

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.win32.programmer.kernel/2005-07/msg00246.htm>

- *From:* "Ivan Brugiolo [MSFT]" <ivanbrug@xxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 11 Jul 2005 12:21:18 -0700
-

This is a typical case of multi-tier application that requires preservation of identity across multiple authentication hops.

In the ideal implementation, the web client supplies to the web server the identity of the user. This can be accomplish via a client certificate, via some password-request via SSL, or, in a pure Windows Authentication Authority system, the IE browser can extract the identity from the current subject context.

Once the authenticated client hits the server, the server grabs the identity, and uses that in every operation performed on client behaf. This is done by both IIS.5 and the ASP.NET worker processes (configuration details may vary if it's a web service or not).

When you have more than one authentication hop, such as the web server accessing some file on a third machine, you need delegation. The users credential must be delegatable, and the web server machine must be trusted for delegation in AD. With delegated credentials, the web server can access the file system on a remote machine. All of this can happen without ever writing home brewed authentication and access-check code.

Now, I'm not completely sure and familair on how a Java service handles authentication in a windows environment. If all the features that your scenario requires are supported by the infrastructure (Jave + RMI), then you should need to do nothing besides configuring the pieces.

If your authentication infrastructure loses the identity, then you need to come out with some home brewed solution. Assuming you can get an impersonation token at delegation level somewhere in the system, you can pass that across one authentication hop by using `InitializeSecurityContext/AcceptSecurityContext/ImpersonateSecurityContext`.

Re: Getting seteuid/setegid functionality out of Windows

You would need to move the security blobs across some boundary via some mechanism of your choice, but, the delegated identity can be reconstructed in a different machine. This mechanism is naturally as weak as the the transport over which you pass the security blobs.

The temptation of faking a "uid" via NtCreateToken is short-legged in this scenario, because the token would have no LUID, no logon session, and it would be useful only on the local machine authority scope, that is not suitable for DFS (unless you make the Web server, the DFS server, the DFS target and the Domain Controller the same box, that would defeat the multi-tireness of the scenario.)

--

This posting is provided "AS IS" with no warranties, and confers no rights. Use of any included script samples are subject to the terms specified at <http://www.microsoft.com/info/copyright.htm>

"tanis" <tanis@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message news:9D3B432A-E258-4E79-9FD3-680739DC18D8@xxxxxxxxxxxxxxxxxxxx
> I have been posting on the visual c++ newsgroup on this issue and was told
> a
> few times that this was the probably the proper place to post this issue.
>
> Here's the situation:
>
> 1) I'm basically using Java as the front-end to my application. The
> back-end is some services that run on remote machines, namely Windows 2000
> machines in this case, that do some system-level stuff as a response a
> remote
> request from a client. Java front-ends these services on the remote
> machines
> and RMI is being used for the client/server activity over TCP/IP.
>
> 2) In my application, DFS is being used, as well as other file specific
> activity. I need to have that all coded up in Visual C++ (which is done).
> Now, via JNI, I have the remote Java services able to communiatate to the
> C++
> to do this and that.
>
> 3) The remote services are running the Java services under a specific
> high-level user account. So, when it asks to do something on the file
> system
> via JNI, the ultimate C++ process is run as if the high-level user account
> had requested it.
>
> 4) This leaves me a bit vulnerable if a low-level user asks to do
> something
> on a file system that they'd normally not have access to do. So, for

Re: Getting seteuid/setegid functionality out of Windows

- > example, if a user wants a list of files on a remote file system via my
- > tool... I wanted to be able to "become that user" on the remote file
- system
- > and programatically request the list of files. This way, it will adhere
- to
- > any permissions set-up. Same goes if the user wanting to delete or create
- a
- > file.
- >
- > 5) In Linux/Unix, you can use seteuid/setegid to "become the effective
- user"
- > in the C++ on the remote machine and then do a stat or whatever and see if
- it
- > works as that user. In Windows, I'm not so sure how to go about doing
- that
- > without a password yet.
- >
- > 6) This is where the password issue can get funky. Now in theory, I guess
- I
- > can ask the user when they start my GUI for their password and store it
- > somewhere and pass it over when a remote request is needed. But that gets
- a
- > bit crazy when users wanna use the command-line to rapid fire requests.
- >
- > Are there other ways to "check" access of a user for things like delete,
- > create and list files without having to be that user?
- >
- > Thanks for anything and everything.

• **References:**

- ◆ **[Getting seteuid/setegid functionality out of Windows](#)**

◇ From: tanis

- Prev by Date: **[Re: MapViewOfFile Vs CreateProcess](#)**
- Next by Date: **[Re: Change redirection of stdout of child process](#)**
- Previous by thread: **[Re: Getting seteuid/setegid functionality out of Windows](#)**
- Index(es):
 - ◆ **[Date](#)**
 - ◆ **[Thread](#)**