

RE: Filecopy to network share

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.general/2006-08/msg00006.html>

- *From:* stcheng@xxxxxxxxxxxxxxxxxxxxxx (Steven Cheng[MSFT])
 - *Date:* Tue, 25 Jul 2006 05:45:57 GMT
-

Hello Michel,

Welcome to the MSDN newsgroup.

From your description, you're developing an .NET application which will programmatically access a network share folder and copy some files into it. Since the share folder is protected, you're encountering problems access it in code, correct?

Based on my experience, according to your scenario, you have the following two difficulties need to overcome:

1. Let your application(current thread) running under a specific security identity other than the default logon user (for winform or console application).
2. Generate an identity/account on your webserver(where the code runs) which can be used as our application's security identity, and this identity should be authenticatable on the remote network share's machine.

For #1, we can use the .net platform invoke to call win32 "LogonUser" api and impersonate our application code to run under the specific logon user identity. The following kb article demonstrate how to use managed code to perform impersonate(it applies to both desktop and asp.net application):

#How to implement impersonation in an ASP.NET application
<http://support.microsoft.com/kb/306158/en-us>

For #2, since the remote share is on a DMZ server (which has only local users and groups), we can not domain account to access it, however, the logonuser API can only access an account(credential) on local machine(for your scenario it's the domainA webserver) or domain. To resolve this, you need to create two duplicated account which have the same username and password on both machines(the domainA webserver and the DMZ webserver). Thus, on our domainA webserver, we can impersonate our application to run

RE: Filecopy to network share

under the "localmachine/duplicatedUser" account, and this account's Network Credential can be used to access the remote DMZ server(and its share folders). Also, you need to grant the permission for this duplicated account on the DMZ server so as to manipulate the share folder.

I've paste a simple test console application's complete code at the bottom of this message demonstrating the impersonate code(I've also include the code file in this message and you can get it if you're using OE reader to access the newsgroup).

Please feel free to let me know if you have anything unclear or any other questions on this.

Sincerely,

Steven Cheng

Microsoft MSDN Online Support Lead

=====

Get notification to my posts through email? Please refer to

<http://msdn.microsoft.com/subscriptions/managednewsgroups/default.aspx#notifications>.

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 project analysis and dump analysis

issues. Issues of this nature are best handled working with a dedicated Microsoft Support

RE: Filecopy to network share

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.

=====main program file=====

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Security.Principal;

namespace ImpersonateConsole
{
class Program
{
static void Main(string[] args)
{
Console.WriteLine("Before Impersonate, User: {0}",
WindowsIdentity.GetCurrent().Name);

if (ImpersonateHelper.ImpersonateValidUser("accountname",
"localmachine or domain name", "Password"))
{
try
{

Console.WriteLine("After Impersonate, User: {0}",
WindowsIdentity.GetCurrent().Name);

//add your remote file access code here

}
finally
{
ImpersonateHelper.UndoImpersonation();
}
}
else
{
Console.WriteLine("Impersonate failed.....");
}
}
}
```

RE: Filecopy to network share

RE: Filecopy to network share

```
}  
}  
}
```

=====helper class code=====

```
using System;  
using System.Collections.Generic;  
using System.Text;  
using System.Security;  
using System.Security.Principal;  
using System.Runtime.InteropServices;  
  
namespace ImpersonateConsole  
{  
    public class ImpersonateHelper  
    {  
        private static WindowsImpersonationContext impersonationContext;  
  
        public const int LOGON32_LOGON_INTERACTIVE = 2;  
        public const int LOGON32_PROVIDER_DEFAULT = 0;  
  
        [DllImport("advapi32.dll")]  
        public static extern int LogonUserA(String lpszUserName,  
        String lpszDomain,  
        String lpszPassword,  
        int dwLogonType,  
        int dwLogonProvider,  
        ref IntPtr phToken);  
        [DllImport("advapi32.dll", CharSet = CharSet.Auto, SetLastError =  
        true)]  
        public static extern int DuplicateToken(IntPtr hToken,  
        int impersonationLevel,  
        ref IntPtr hNewToken);  
  
        [DllImport("advapi32.dll", CharSet = CharSet.Auto, SetLastError =  
        true)]  
        public static extern bool RevertToSelf();  
  
        [DllImport("kernel32.dll", CharSet = CharSet.Auto)]  
        public static extern bool CloseHandle(IntPtr handle);
```

RE: Filecopy to network share

RE: Filecopy to network share

```
public static bool ImpersonateValidUser(String userName, String
domain, String password)
{
    WindowsIdentity tempWindowsIdentity;
    IntPtr token = IntPtr.Zero;
    IntPtr tokenDuplicate = IntPtr.Zero;

    if (RevertToSelf())
    {
        if (LogonUserA(userName, domain, password,
LOGON32_LOGON_INTERACTIVE,
LOGON32_PROVIDER_DEFAULT, ref token) != 0)
        {
            if (DuplicateToken(token, 2, ref tokenDuplicate) != 0)
            {
                tempWindowsIdentity = new
WindowsIdentity(tokenDuplicate);
                impersonationContext =
tempWindowsIdentity.Impersonate();
                if (impersonationContext != null)
                {
                    CloseHandle(token);
                    CloseHandle(tokenDuplicate);
                    return true;
                }
            }
        }
    }
    if (token != IntPtr.Zero)
        CloseHandle(token);
    if (tokenDuplicate != IntPtr.Zero)
        CloseHandle(tokenDuplicate);
    return false;
}

public static void UndoImpersonation()
{
    impersonationContext.Undo();
}

}
}
```

Attachment: ImpersonateHelper.cs

Description: Binary data

RE: Filecopy to network share

RE: Filecopy to network share

Attachment: **Program.cs**

Description: Binary data