Realtime publishers

The Essentials Series: Fundamentals of Effective File Server Security

Enforcing File and Folder Security

sponsored by

SCRIPTLOGIC®

by Greg Shields

Copyright Statement

© 2009 Realtime Publishers. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtime Publishers (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtime Publishers or its web site sponsors. In no event shall Realtime Publishers or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtime Publishers and the Realtime Publishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtime Publishers, please contact us via e-mail at info@realtimepublishers.com.

Enforcing File and Folder Security

Files and folders…folders and files…data everywhere! It seems like even the smallest of business networks somehow aggregate mounds of those individual buggers each and every day. Collecting up on our file servers, it is the protection of that data that is arguably our primary job as IT administrators.

Yet as a fundamental part of our daily activities, managing file server security is one task we must accomplish using some of the least-capable tools in our administrative quiver. With little more than an Explorer shell and a few native command-line tools, we're charged with maintaining the security and availability of our business' most critical assets: *Its data.*

Effectively managing file server security is a three-fold responsibility: We must first ensure that rights and permissions are set correctly to give the correct people access and keep the wrong people out. Lacking specialized tools to assist, completing this task requires large amounts of mouse-work as we click through to set each file and folder's properties.

But setting security isn't efficient without a good visualization of the structures we have to work within. The second responsibility in effective file server security is in enumerating files, folders, and their assigned permissions. Lacking tools here, we're further forced into highly-manual activities that don't scale with the growth of our data.

The third facet of this responsibility arrives as an unfortunate side effect of how dynamic an IT environment really is. Once set, we must constantly and comprehensively audit that security to make sure it hasn't been changed. Ultimately, *keeping* the wrong people out is as much work as *locking* them out in the first place.

It is the goal of this Essentials Series to assist you with these three responsibilities. You know as well as the next IT professional that enforcing, enumerating, and auditing file server security can be a painful process without the right tools. This series will show you tools that are available right within the Windows operating system (OS). Along the way, you may discover that alternative solutions might come in handy for easing the strain of this daily activity.

Permissions in Two Flavors: Share and NTFS

Effectively enforcing file server security means monitoring permissions at both the level of NTFS as well as with each individual shared folder. For data that is accessed via a shared path, both sets of permissions determine a user's rights to work with files and folders in the share.

There is often some confusion as to what privileges are conveyed through each of these two permission structures. Part of this confusion is because both NTFS and shared folders leverage similar permission sets such as Full Control and Read. By design, NTFS permissions control the access to a particular file or folder. That access holds true no matter how the user navigates to the file or folder's location. Share permissions, however, apply only to objects that are accessed through a defined shared folder path.

Share Permissions

Share permissions are often considered the gatekeepers of a shared resource. Correctly implemented, they limit the boundary of access that a user can have on any item within the share. This extra security boundary enables a blanketed control over all file and folder access within the share, which is then tailored down further using the NTFS permissions applied to each individual file and folder. There are three levels of control exposed at the share level:

- The most basic access is *Read*. Read allows a user to be controlled by the Read & Execute, List Folder Contents, and Read NTFS permission groups.
- The next level of control at the share level is *Change*, which exposes all the additional NTFS permission groups except Full Control.
- The final level, *Full Control*, allows access to the remaining NTFS permission group of the same name.

It is important to remember share permissions do not grant any direct access to files or folders. They only enable a configured user or group to traverse through the share. The other half of permissioning is handled using NTFS permissions.

NTFS Permissions

Once through a shared folder, NTFS permissions further define which users can access which resources. These permissions control access to the file or folder itself as well as to its specific attributes. At their most basic level, NTFS permissions define whether a user or group can read, write, and ultimately delete a particular file or folder.

But these three rights are only the most basic of those that can be assigned. An additional set of "special permissions"—over a dozen in all and different based on each object's type—can be assigned as necessary. This granularity enables you to create complex permissioning structures where users can traverse folder structures but not read their contents or create files but not delete them. Exceptional care is necessary when working with these special permissions, as their incorrect assignment can create havoc for user access.

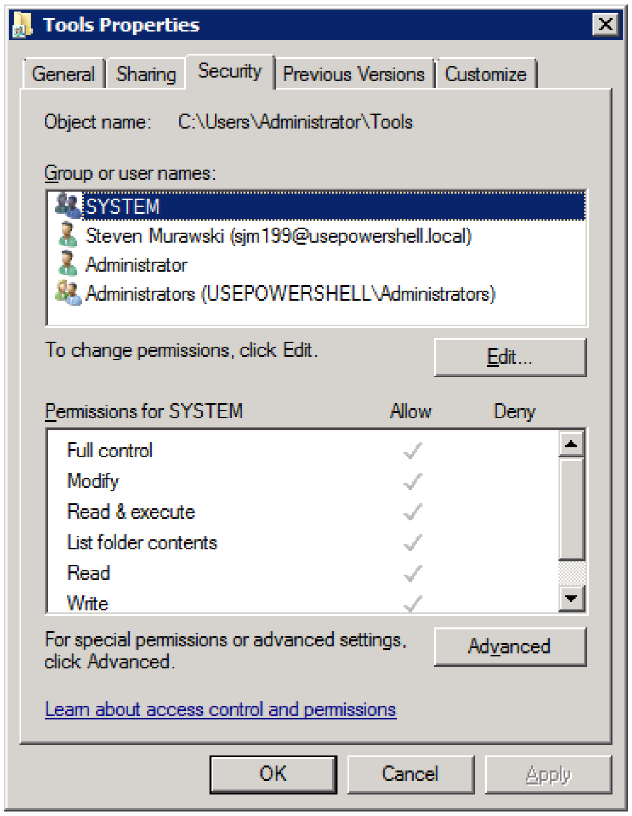
Complicating this situation even further, the permission structure for NTFS is cumulative. If a user is a member of several groups that have all been granted different permissions to a file or folder, the user will be granted the least-restrictive effective permission. As such, keeping a user out of a folder means not assigning its privileges to the user's user account as well as any of the groups to which the user is a member.

This cumulative behavior is also experienced with folder inheritance. With inheritance, a file or folder can receive its permissions from the folder above it in the tree. Thus, down-level files and folders can have their permissions automatically assigned from another folder that is one or many folders above in the file structure. Finding where those permissions are assigned can be a challenging activity using native tools alone. In complex environments with large folder structures, third-party solutions are often necessary to visualize and untangle what eventually becomes a spaghetti of inherited permissions.

Finally, the NTFS permission structure includes a powerful (and often painful) Deny permission. Whereas access that is enabled through the standard Allow permissions follows the rules previously mentioned, any Deny permission automatically and immediately overrides everything. This is useful for creating a blanket restriction for highly-secure folders but must be used carefully. If you've ever accidentally set the Deny permission for the Everyone group on a file or folder, you know how challenging this permission can be to wield effectively.

How to Set Permissions

NTFS permissions are defined within Windows Explorer by navigating to the target file or folder and viewing the Security tab of the object's properties (see Figure 1). The Security tab shows a list of accounts and groups that have access to the file. When a user or group is selected, the permission groups enabled for them are highlighted.


Tools Properties

General | Sharing | Security | Previous Versions | Customize

Object name: C:\Users\Administrator\Tools

Group or user names:

- SYSTEM
- Steven Murawski (sjm199@usepowershell.local)
- Administrator
- Administrators (USEPOWERSHELL\Administrators)

To change permissions, click Edit.

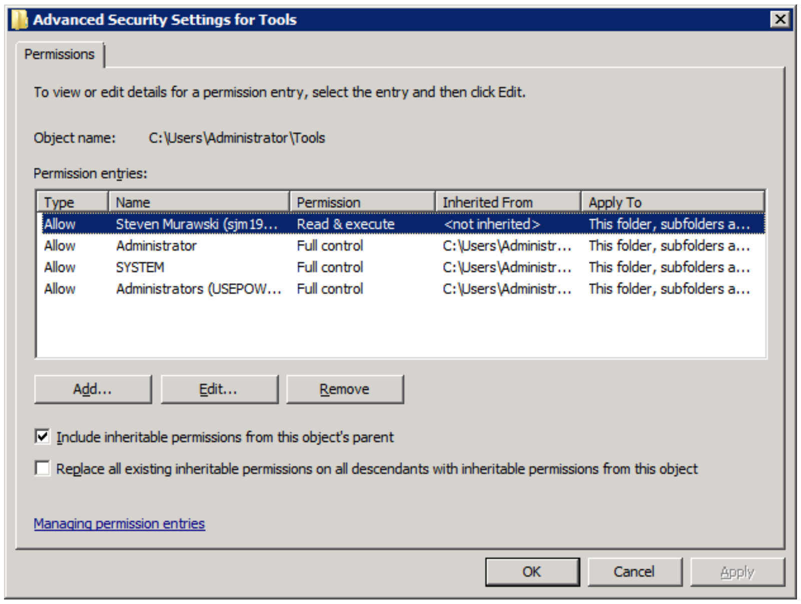
Permissions for SYSTEM	Allow	Deny
Full control	✓	
Modify	✓	
Read & execute	✓	
List folder contents	✓	
Read	✓	
Write	✓	

For special permissions or advanced settings, click Advanced.

Learn about access control and permissions

Figure 1: Viewing NTFS permissions.

Clicking Edit will allow you to modify the permission groups assigned or associate new groups. Clicking Advanced followed by the resulting Edit button enables access to edit Advanced Security Settings (see Figure 2).


Advanced Security Settings for Tools

Permissions

To view or edit details for a permission entry, select the entry and then click Edit.

Object name: C:\Users\Administrator\Tools

Permission entries:

Type	Name	Permission	Inherited From	Apply To
Allow	Steven Murawski (sjm19...	Read & execute	<not inherited>	This folder, subfolders a...
Allow	Administrator	Full control	C:\Users\Administr...	This folder, subfolders a...
Allow	SYSTEM	Full control	C:\Users\Administr...	This folder, subfolders a...
Allow	Administrators (USEPOW...	Full control	C:\Users\Administr...	This folder, subfolders a...

☑ Include inheritable permissions from this object's parent

☐ Replace all existing inheritable permissions on all descendants with inheritable permissions from this object

Managing permission entries

Figure 2: Editing advanced security settings.

This wizard allows you to associate users or groups with permissions on the targeted object. It also allows you to control how inherited permissions affect the object and any sub-objects. Editing permissions from there allows access to the special permissions.

Yet these are all tools you already use today. You already use Windows Explorer to accomplish at least the basics of permissioning through the GUI. What you need are automation solutions that make this process easier. Let’s start by looking at some common command-line solutions that are available right out of the box.

Automating Permissioning

Windows Server 2008 ships with several command-line tools to help manage NTFS and share permissions. These tools help alleviate some of the scalability and consistency limitations experienced with Windows Explorer. Three in particular are useful right out of the box:

- Share permissions can be managed through the *net share* command.
- NTFS permissions can be managed with *icacls.exe*, also known as Improved Change ACLs.
- Microsoft’s investment in Windows PowerShell further extends scripting support through cmdlets such as Set-ACL for working with NTFS permissions.

Net Share

Creating shares from the command line via “net share” is a fairly straightforward process. To create the share, you must pass the share name and the path to the share:

```
Net share MyFolderShare=c:\Users\Administrator\Tools
```

Net share can also be used to grant permissions to users and groups when establishing the share using the /grant switch:

```
Net share MyFolderShare=c:\Users\Administrator\Tools
/Grant:MyDomainSysAdmins@mydomain.local,CHANGE /Grant:steve@mydomain.local,FULL
```

As the previous example shows, the /Grant switch can be used more than once to add multiple users or groups. Be aware that “net share” will not allow you to modify permissions on an existing share.

Icacls.exe

Icacls.exe is a command-line tool for managing both permissions and file integrity levels. It is natively available in Windows Server 2008, Windows Vista, and Windows 7, and arrives as an evolution of the cacls.exe command first available in Windows NT 4.0. Icacls.exe offers rich permissioning support for files and folders, enabling an administrator to automate the application of both simple as well as special permissions.

Its challenge is in its syntax. Although it is powerful in what it can do, applying permissions using icacls.exe involves a complex aggregation of switches and inheritance operators. For example, let's assume you need to remove the existing inherited permissions from the C:\Shared folder. At the same time, you want to directly apply the Read permission to the Domain Users group and the Full Control permission for File Administrators. To accomplish this, you would use the following icacls.exe syntax:

```
Icacls C:\Shared /inheritance:r /grant:r “Domain Users”:(OI)(CI)R /grant:r “File
Administrators”:(OI)(CI)F
```

Once through its learning curve, icacls.exe's true power arrives by stringing together long series' of these commands to automate permissioning at multiple levels. Once scripted, you can re-run the script over and over again to reapply permissions to your file structure.

Note
A thorough explanation of scripting with icacls.exe can be found at http://technet.microsoft.com/en-us/magazine/2009.07.geekofalltrades.aspx.

PowerShell

The final and arguably most powerful native tool is PowerShell, Microsoft's new command-line automation framework. PowerShell provides a cmdlet called Set-ACL that can be used alone or incorporated within a larger script for managing NTFS permissions. Set-ACL is best used when modifying an existing permission set, whether that permission set is from the object you want to modify or another object. The Set-ACL solution is also the most verbose of the command-line options:

```
$AclToModify = Get-ACL –Path ‘c:\Users\Administrator\Tools’
$NewPermission = New-Object
System.Security.AccessControl.FileSystemAccessRule(“MyLocalDomain\Steve”,”Modify”,
”Allow”)
$ AclToModify.AddAccessRule($NewPermission)
Set-ACL –Path ‘c:\Users\Administrator\Tools’ –ACLObject $AclToModify
```

Going Beyond Native Toolsets

As you can see, the native toolsets available in Microsoft Windows create a framework for the enforcement of permissions. Yet each of these tools arrives with their own challenges, requiring either heavy mouse-work or a deep knowledge of scripting to be truly useful.

As the primary job of every IT administrator, managing data on file servers requires a lot of time and effort. As such, other non-native solutions may be necessary as the size of your data scales. These tools enable the graphical visualization of permissions structures through built-in discovery, reporting, and analysis engines. With the right views in place, administrators can leverage built-in automation that defines and maintains widespread permissions based on best practices. If the management of your file servers is a strain, consider looking to external solutions for enforcing your file and folder security.