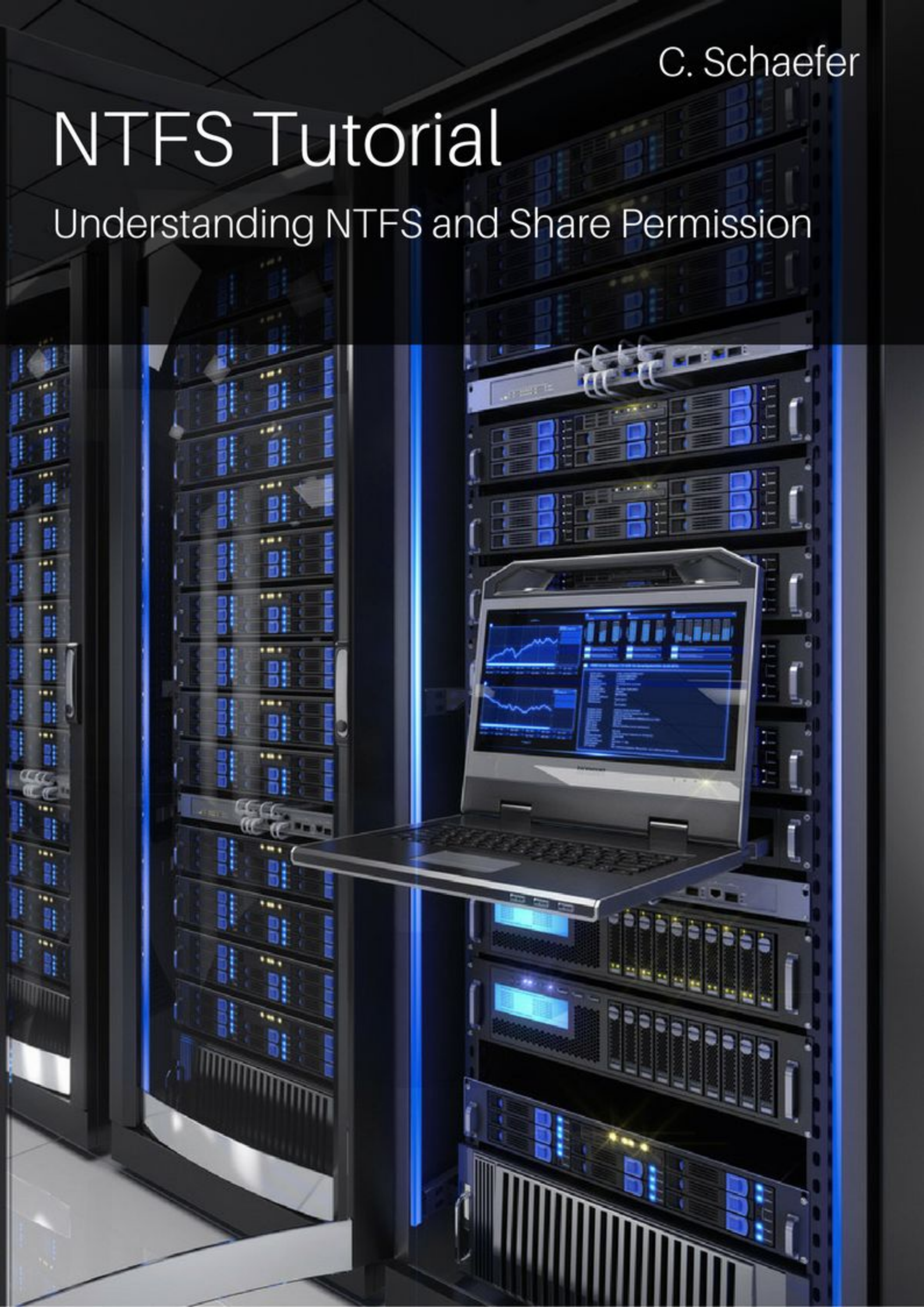


C. Schaefer

# NTFS Tutorial

Understanding NTFS and Share Permission



# Chapter 1

## Understanding NTFS Permissions

# Basic Concepts

## NTFS

NTFS, which stands for **New Technology File System**, is Microsoft's current file system for the Windows NT operating system. NTFS is the successor of Microsoft's previous systems, FAT and HPFS, and contains a wide range of improvements in terms of performance, extendibility, and security.

*The main differences between NTFS and its predecessors are:*

- FAT32 only supports individual files of up to 4GB in size. On the other hand, NTFS supports files of up to 16 EiB ( $16 \times 1024^6$  or  $2^{64}$  bytes).
- The most important difference you need to understand in order to follow this tutorial is that NTFS supports file permissions and introduced the concept of the **access control list** (ACL), a concept we will be explaining in more detail as we proceed.

## NTFS Permissions

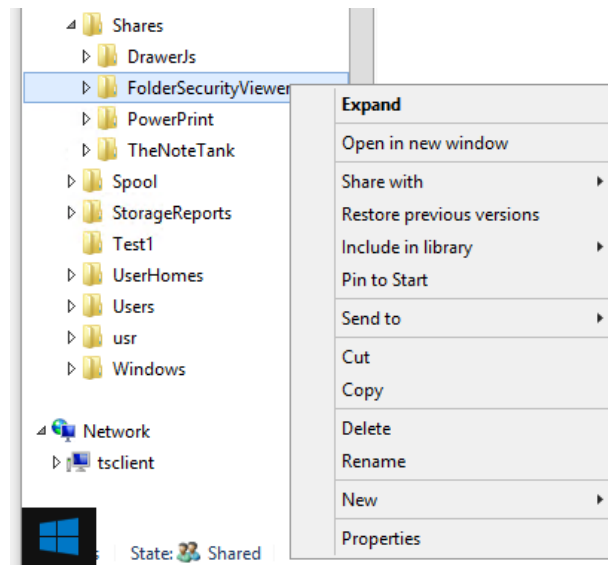
NTFS permissions determine who have access to files or folders. These permissions can be assigned to individual users or groups, but the best practice is to assign them to groups whenever possible. Permissions are set in the ACL.

## Access Control List (ACL)

The **access control list** (ACL) is the list of users or groups that have access to a certain object. An object can be a file or folder. Each entry in the ACL is known as an **access control entry** (ACE).

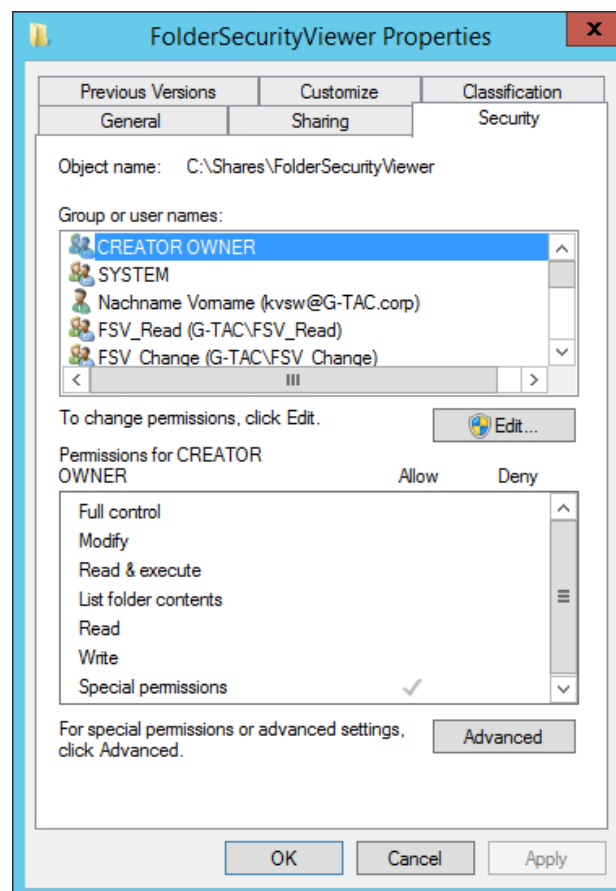
The users or groups in the ACL are known as **trustees**. Permissions can be allowed, denied, or audited.

To create, edit, or view access control lists, you right click on a file or folder then select "Properties" from the options displayed:



Menu options relating to a file or folder in Windows

Next, click on the “Security” tab to display the access control list (ACL) for the chosen file or folder.



An access control list (ACL) in the Windows Server 2012 R2 / 2016

# Understanding Permissions

Windows allows you to assign different types of permissions to an object. You can allow or deny such permissions. The types of permissions change depending on if you are working with a file or folder.

## NTFS Folder Permissions

You can assign permissions to a user or group for a specific folder and, thus, control their access level. How these permissions are propagated to subfolders and their respective files is controlled by **inheritance**, a concept we will explain in more detail as we proceed. Next table lists and describes all permissions that can be allowed or denied for a certain user or group.

Permission	Description
<b>Full Control</b>	Specifies whether a user or group has all available permissions for a folder.
<b>Modify</b>	Specifies whether a user or group can modify the contents of a folder. It is more restrictive than full control, as it does not allow users/groups to change permissions or take ownership of said folder.
<b>Read and Execute</b>	Specifies whether a user or group can read the data within a folder and execute the programs said folder contains.
<b>List Folder Contents</b>	Specifies whether a user or group can list the content of a folder. This does not allow users/groups to run any of the programs or read any of the data within the folder.
<b>Read</b>	Specifies whether a user or group can read the data within a folder. As opposed to "Read and Execute", if there is an executable file within the folder, the user or group will be unable to run it.
<b>Write</b>	Specifies whether a user or group can create files and folders, write data, and write attributes for a folder. The write permission implies the ability to read all data within the folder.
<b>Special Permissions</b>	Refer to TABLE 3 for the list and description of special permissions.

List of NTFS folder permissions



## NTFS File Permissions

You can assign permissions to a user or group for a specific file and, thus, control their access level. Next table lists and describes all permissions that can be allowed or denied for specific users or groups. NTFS file permissions take priority over NTFS folder permissions.

### *For example,*

if you have access to a folder, but an administrator denies access for a file within that folder, you cannot access that file even if you have the necessary permissions for its parent folder.

<i>Permission</i>	<i>Description</i>
<b>Full Control</b>	Specifies whether a user or group has all available permissions for a file.
<b>Modify</b>	Specifies whether a user or group can modify a file. It is more restrictive than full control, as it does not allow users/groups to change permissions or take ownership of said file.
<b>Read and Execute</b>	Specifies whether a user or group can read the contents of a file and execute the programs of said file.
<b>Read</b>	Specifies whether a user can read a file's data. As opposed to "Read and Execute", if the file in question is an executable file, the user or group will be unable to run it.
<b>Write</b>	Specifies whether a user or group can change the content or, in other terms, write data to a file. The write permission implies the ability to read all the data contained in a file.
<b>Special Permissions</b>	Refer to next table for the list and description of special permissions.

List of NTFS file permissions

<i>Permission</i>	<i>Description</i>
<b>Traverse Folder/ Execute File</b>	Traverse Folder allows a user or group to access a folder nested within a tree, even if parent folders in that tree deny said user/group access to the contents of those folders. Execute File allows a user or group to run a program.
<b>List Folder/ Read Data</b>	List Folder allows a user or group to see objects (files and folders) inside a folder. Read Data allows a user or group to open and view a file
<b>Read Attributes</b>	Allows a user or group to view basic attributes of an object (read-only, system, archive, and hidden).
<b>Read Extended Attributes</b>	Allows a user or group to view the extended attributes of an object. For example: the summary, author, title, and so on for a Word document. These attributes vary from program to program.
<b>Create Files/ Write Data</b>	Create Files allows a user or group to create new objects within a folder. Write Data allows a user or group to overwrite an existing file.
<b>Create Folders/ Append Data</b>	Create Folders allows a user or group to nest folders. Append Data allows a user or group to add data to an existing file, but not delete data within that file or delete the file itself.

Special permissions

## NTFS Access Limitations

Microsoft provides the following table to offer a more detailed understanding of what each permission can allow you to do. You should always refer to this table when assigning permissions. Try to assign the most restrictive possible permissions for each use case. A common bad practice in many IT business environments is to assign “full control” every time a user or group requests access to a file or folder.

<i>Special Permissions</i>	<i>Full Control</i>	<i>Modify</i>	<i>Read and Execute</i>	<i>List Folder Contents</i>	<i>Read</i>	<i>Write</i>
Traverse Folder/ Execute File	x	x	x	x		
List Folder/ Read Data	x	x	x	x	x	
Read Attributes	x	x	x	x	x	
Read Extended Attributes	x	x	x	x	x	
Create Files/ Write Data	x	x				x
Create Folders/ Append Data	x	x				x
Write Attributes	x	x				x
Write Extended Attributes	x	x				x
Delete Subfolders and Files	x					
Delete	x	x				
Read Permissions	x	x	x	x	x	x
Change Permissions	x					
Take Ownership	x					
Synchronize	x	x	x	x	x	x

NTFS access limitations

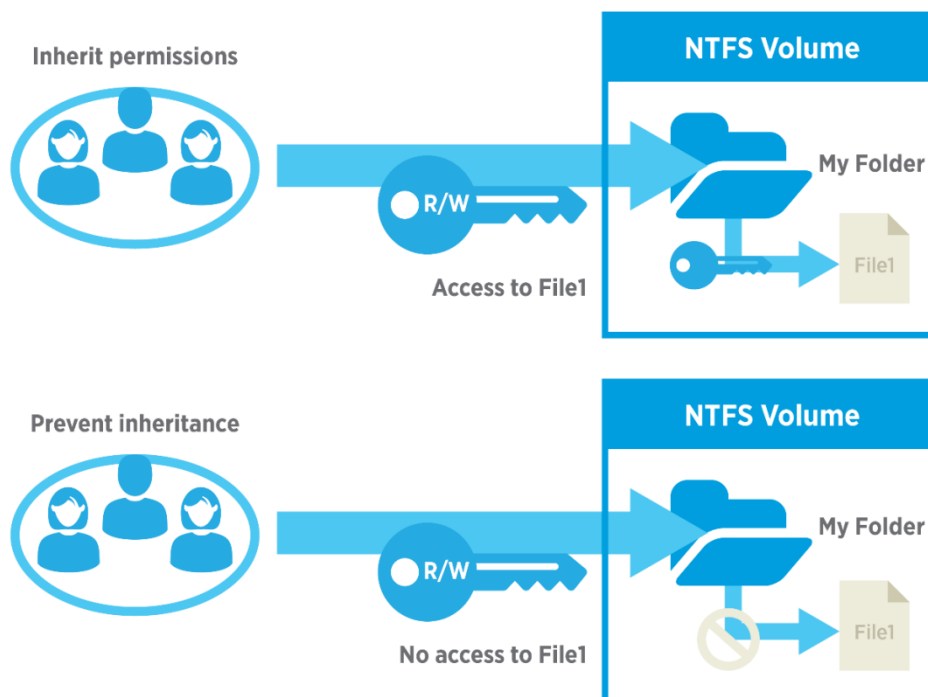


# Permission Inheritance

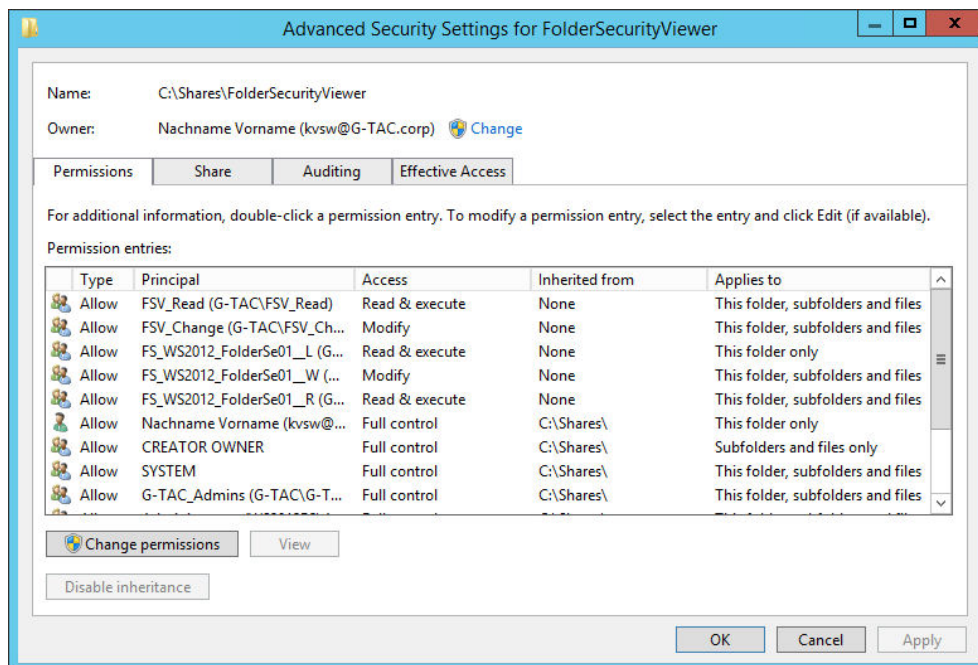
***There are two types of permissions in Windows NT environments:***

- Explicit: Permissions that are applied by default to an object upon its creation or by user action.
- Inherited: Permissions that are propagated to a child object. Inherited permissions facilitate the management tasks related to permissions assignment and ensure consistency among all the objects within a folder.

You must take into account that, by default, all objects created within the same folder inherit permissions from its respective parent folder. For example, if you create a folder called MyFolder, all subfolders and files within MyFolder will inherit its permissions automatically. In this order of ideas, MyFolder has explicit permissions and all subfolders and files in MyFolder have inherited permissions.



You can disable inheritance for any given file or folder by going to the security tab of its properties (as explained above) and clicking on **Advanced** and **Disable Inheritance**.



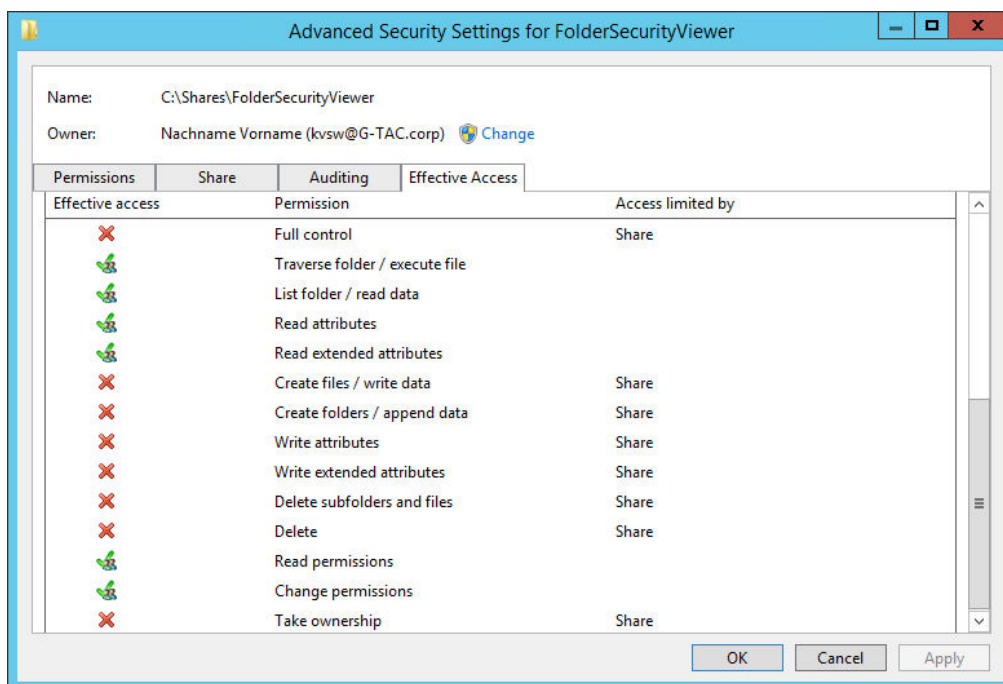
Disabling inheritance and replacing child object permissions in the Advanced Security Settings tab

When administrators and users start changing permissions and making regular changes, some files or folders can become inaccessible and users/groups that should have access to an object can lose their access. That's why you can go back, at any time, to the default inherited state of any prior time by choosing the option "replace all child object permission entries with inheritable permission entries from this object".

# Effective Access

The effective permissions tab, found in the Advanced Security Settings Editor in earlier versions of Windows, was replaced with a tab called effective access in Windows Server 2012 R2 and Windows Server 2016, which lets you choose not only the user or group accessing the file or folder, but also the device accessing that file or folder.

This tab provides an overview of all the permissions assigned to a user or group in regards to accessing a certain object. For example, if John has “read” permissions for MyFolder and belongs to a group with “write” permissions, the effective access tab will show you that John has both “read” and “write” permissions for MyFolder.



Effective access in Windows Server 2012 R2 and Windows Server 2016

# Dynamic Access Control (DAC)

Though the interface has been improved, many of the underlying concepts of NTFS permissions have not changed over the years. The most notable changes are that the effective permissions tab has changed and dynamic access control (DAC) has been introduced.

DAC does not replace NTFS permissions, but does extend the capabilities offered by NTFS permissions and share permissions.

## *For example,*

a user might have different permissions when they access a resource from their office computer than when they access that same resource using a laptop or over a virtual private network (VPN). In addition, access can be granted to a specific user only if said user's device meets the security requirements defined by administrators.



# Chapter 2

**Working with Permissions**

---

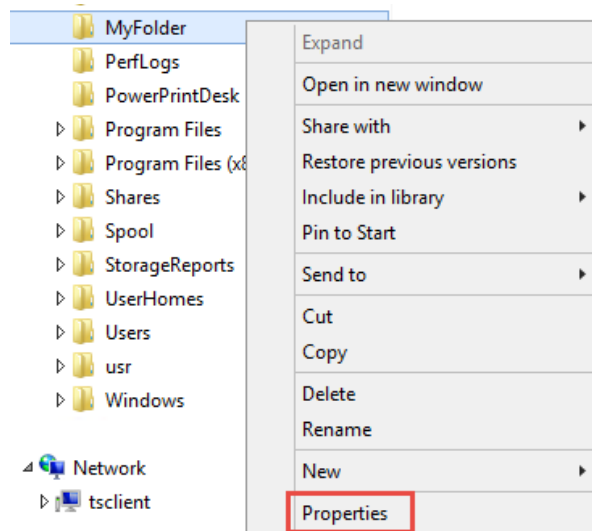
In previous chapters, we covered how to visualize an ACL (the current permissions of a filer or folder). Now you are going to learn how to manage these permissions. Throughout this section, we will continue to use MyFolder as our example folder.

---



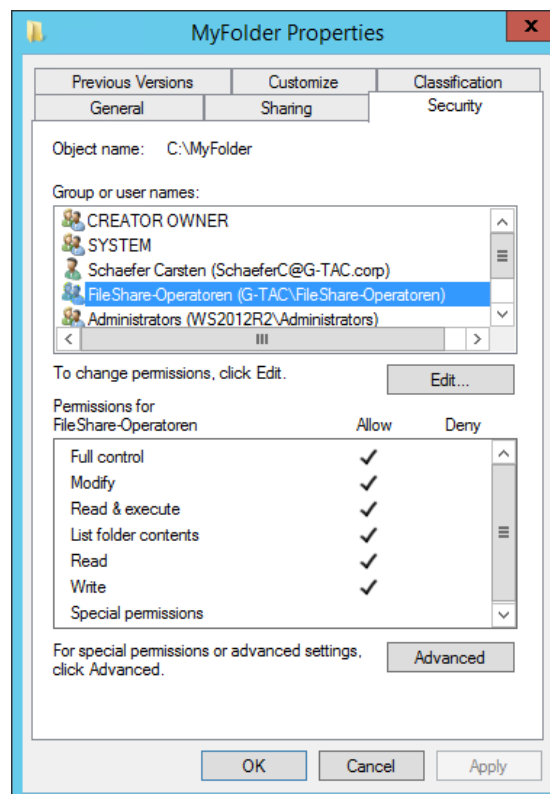
# How to Assign or Remove Permissions

First, locate the folder or file you want to grant permissions to. Right click that folder. Then click on “Properties”.



MyFolder Properties

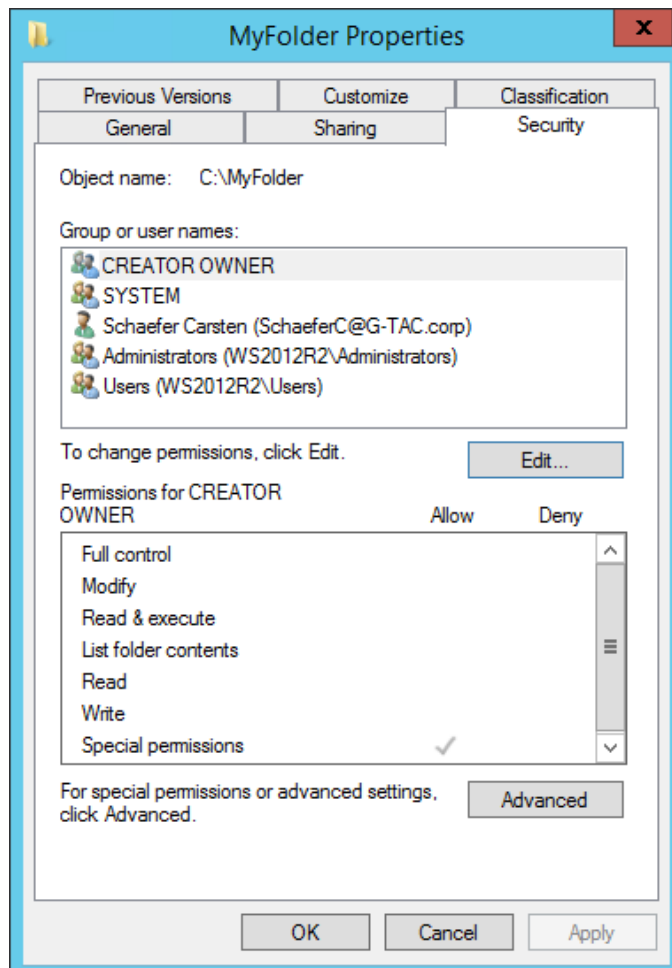
Click on the security tab to view the ACL for the folder. Under “Groups or user names” click the “Edit” button.



MyFolder ACL properties page

To remove a user, simply click on that user and press the “Remove” button followed by the “OK” button.

To add a user, click the “Add” button:

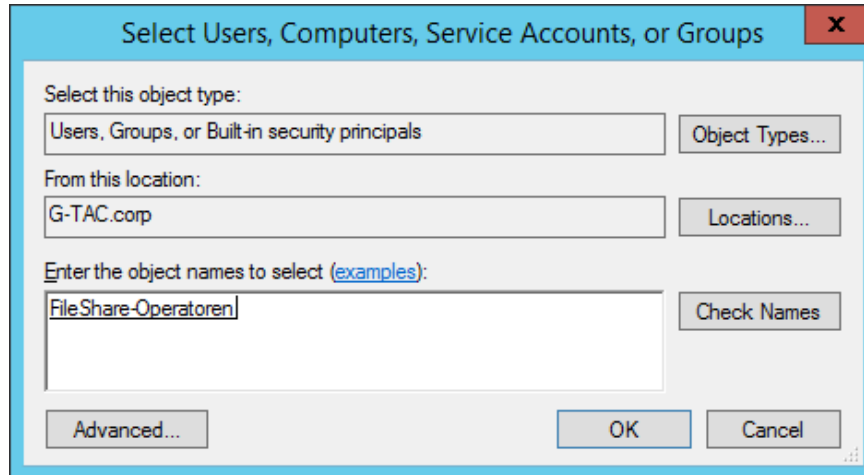


MyFolder permissions properties page

***Now you can select the User/Groups you wish to grant access to MyFolder. The options within the “Select Users or Groups” form are as follows:***

- Object Types: Allows you to filter what type of object you want to assign, in order to narrow your search.
- Locations: If you are on a Windows Network, you can choose between the local computer or Active Directory to search for network users in your organization.

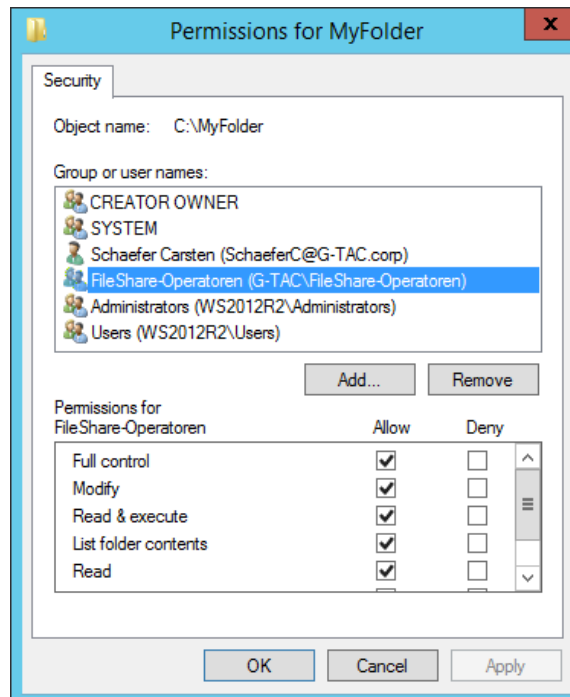
For our example, we are going to type “FileShare-Operatoren” in the “Enter the object names to select” textbox and then click the “Check Names” button, followed by the “OK” button.



Select Users or Groups prompt

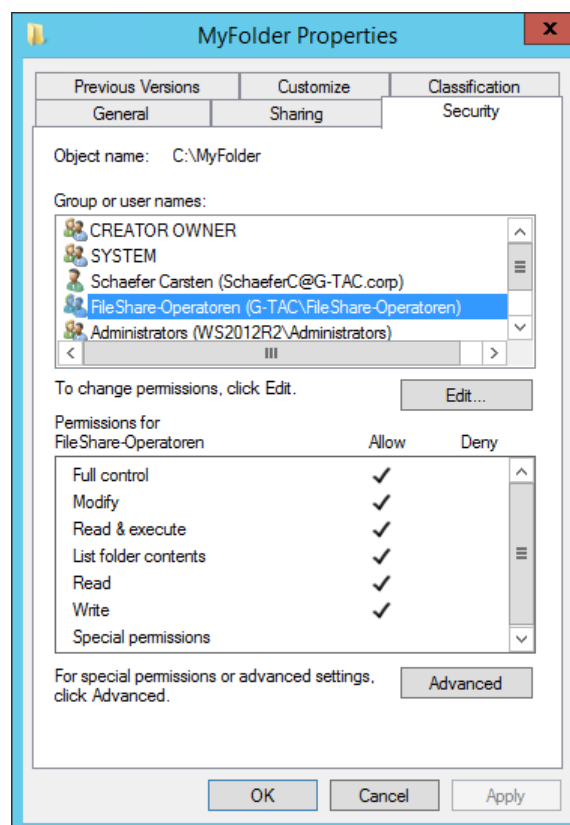
“FileShare-Operatoren” now appears on the “Permissions for MyFolder” page. Here you can choose which permissions to grant the user. Once you assign these permissions, click the “OK” button. For this exercise, we assigned the “Full Control” permission to the user from the “FileShare-Operatoren” group. You can also deny permissions using the “Deny” column.

***As a reminder, we do not recommend denying permissions to users. Instead, it is best to control user access through the groups which they belong to.***



Assigning permissions to the FileShare-Operatoren group

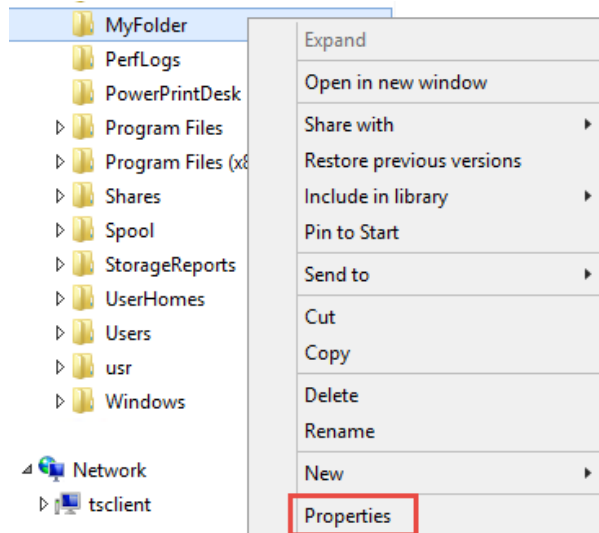
Group now has “Full Control” permissions within the ACL.



MyFolder ACL after assigning permissions to FileShare-Operatoren

# How to Assign Special Permissions

First, locate the folder or file you want to grant permissions to. Right click that folder. Then click on “Properties”.

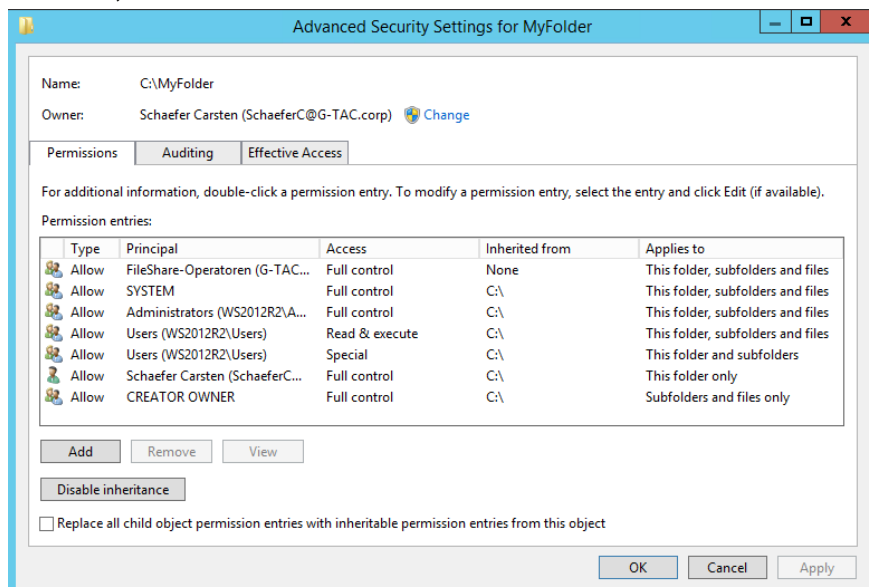


MyFolder properties

Click on the security tab to view the ACL for the folder. Then click on “Advanced”.

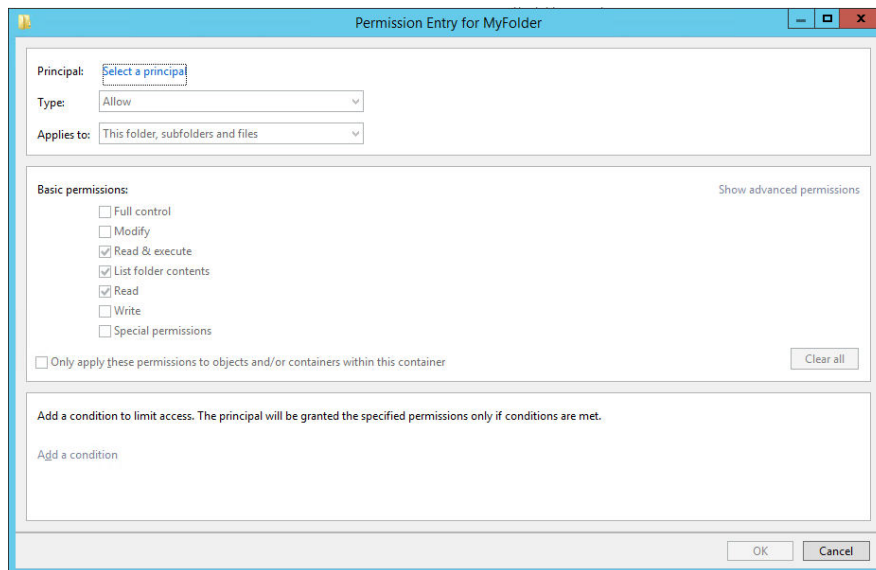
This is the “Advanced Security Settings” tab, which changed in Windows Server 2012 to provide an interface that is easier to understand and manage.

In the permissions tab, click on the “Add” button.



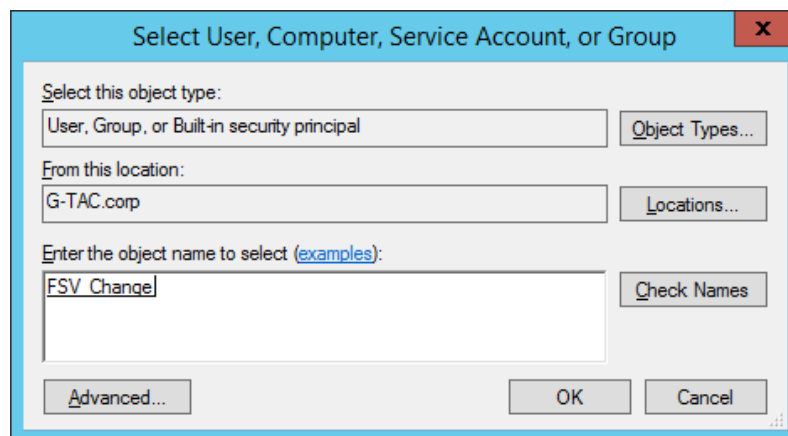
Advanced Security Settings for MyFolder

On the permissions entry page, click on “Select a principal”.



Permission Entry property page for MyFolder

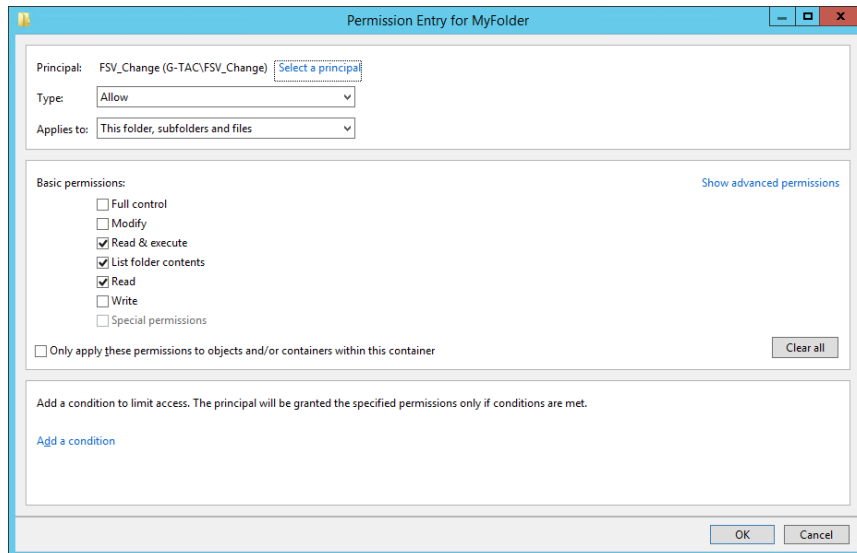
Choose the user or group you want to grant special permissions to. We will use group “FSV\_Change” for our example. Then click the “OK” button.



Select User or Group prompt

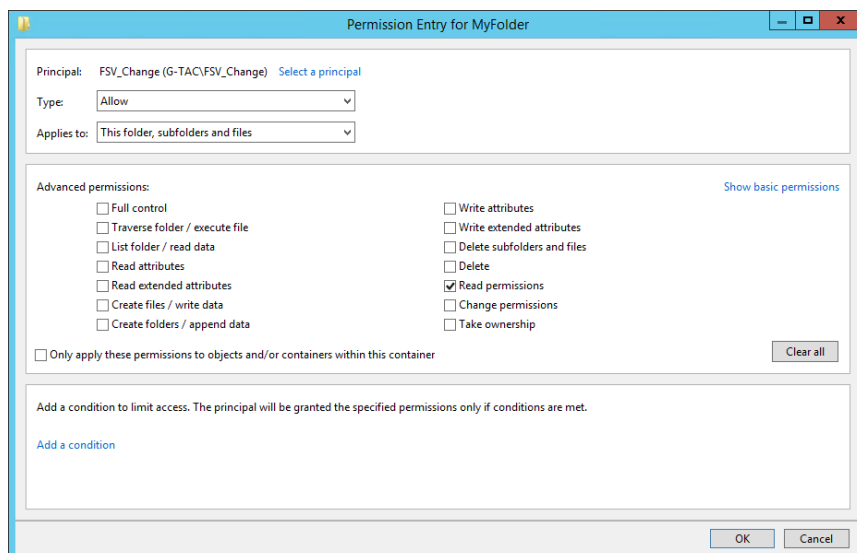


Once you have selected your user or group, by default, you are presented with the list of basic permissions. To see the list of advanced permissions, click on “Show advanced permissions”.



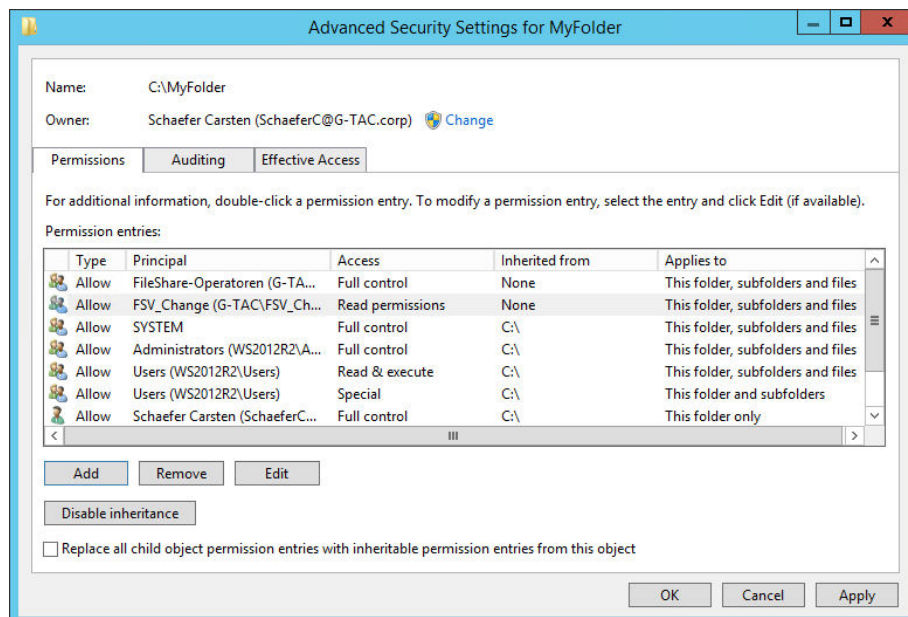
Enabling advanced permissions in the Permission entry for the MyFolder property page

Select the proper advanced permissions for the user and click the “OK” button. For more details about each advanced permission type, please refer to the previous chapter.



Choosing advanced permissions for FSV\_Change

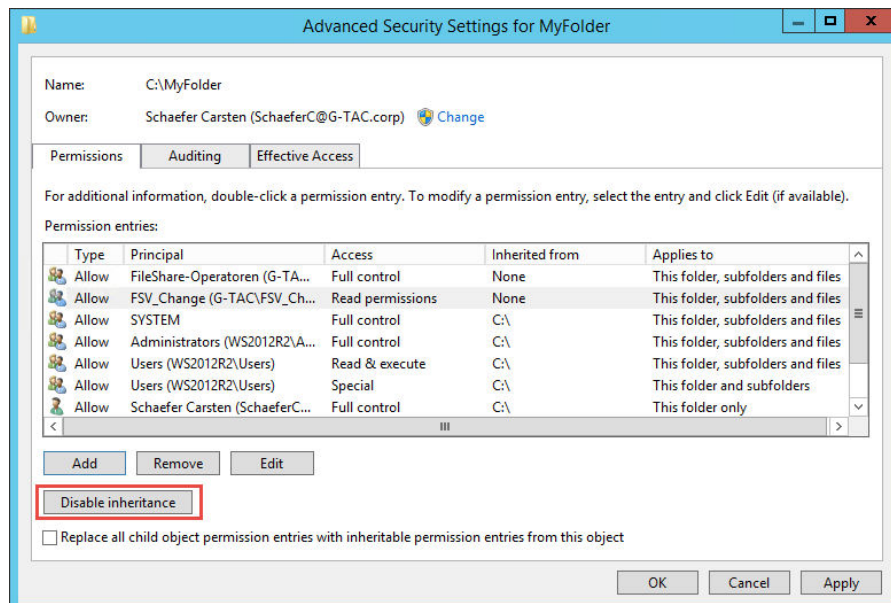
You will now see the advanced permissions assigned on the “Advanced Security Settings” page. Click the “OK” button to complete the process.



View of advanced permissions set for user Schaeferc

# How to Disable Inheritance

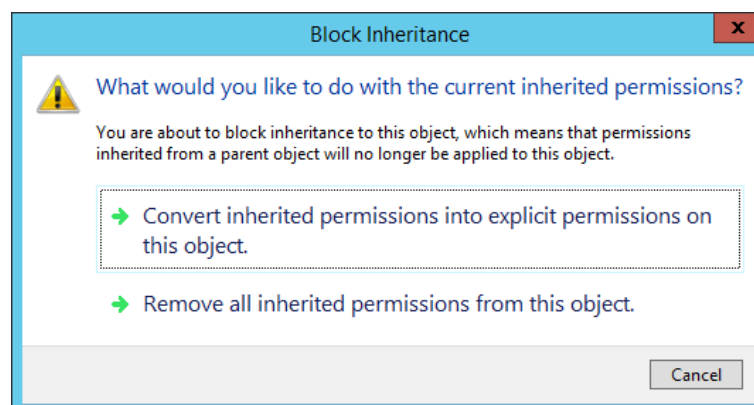
You can disable inheritance for any given file or folder by going to the Security tab within the properties of that file/folder and clicking on “Advanced” followed by “Disable inheritance”.



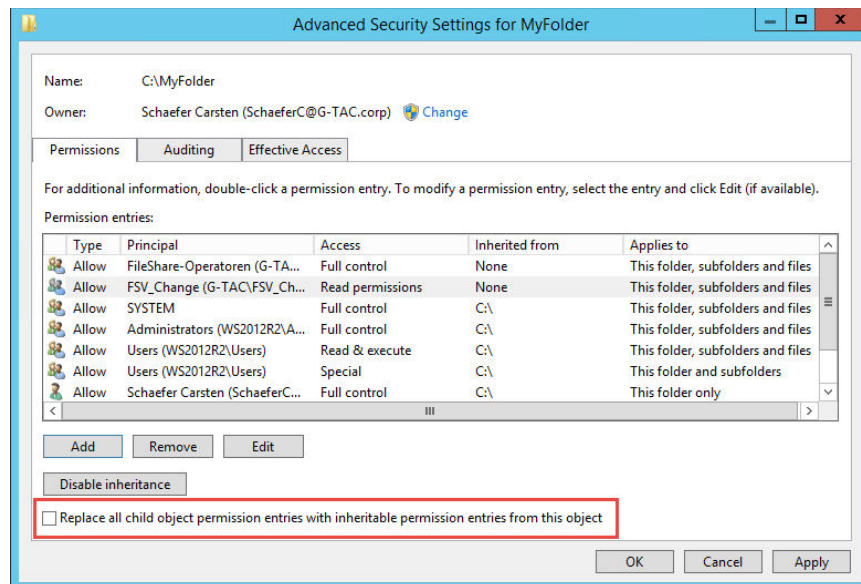
Disabling inheritance and replacing child object permissions in the Advanced Security Settings tab

In next step you have to choose one of the following options. The first option “Convert inherited permissions into explicit permissions on this object” will copy all inherited permissions and set them explicitly on this level. This would be the same as if you set all of these permissions manually.

The second option “Remove all inherited permissions from this object” will remove any permissions. You must be aware that you have to set permissions now by your own. Otherwise no permissions will be set on this folder anyway.



When administrators and users start changing permissions regularly, some files or folders can become inaccessible to users and groups that should have access. For this reason, you can, at any time, go back to the default inherited state by choosing “Replace all child object permission entries with inheritable permission entries from this object” as seen in next image.



# How to Override Folder Permissions with File Permissions

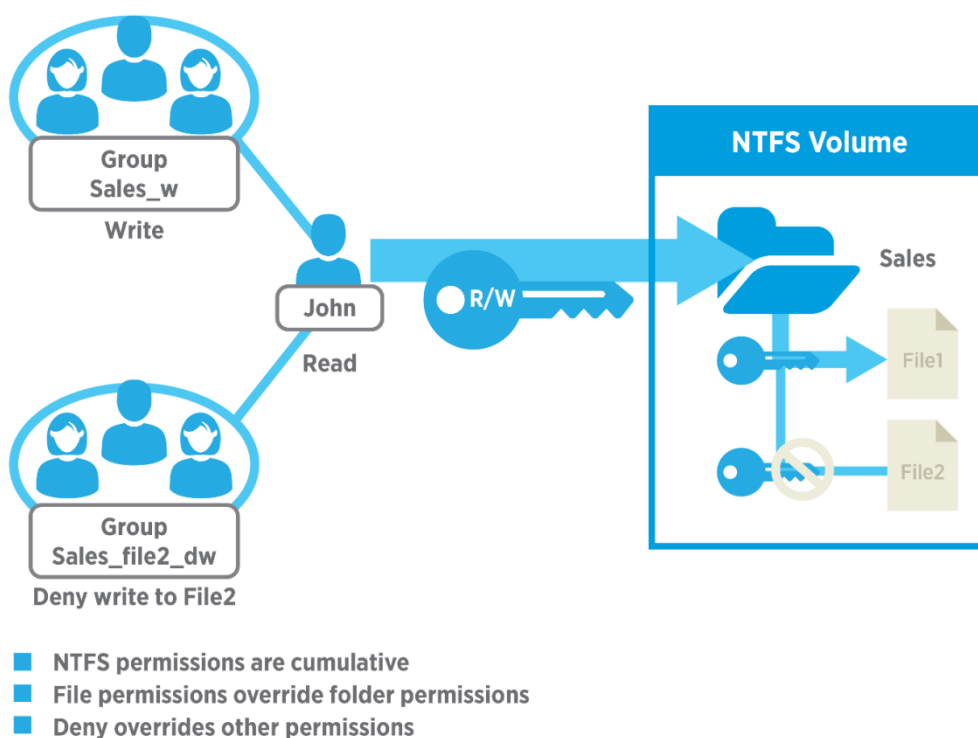
It is possible to override access to a file within a folder that you do not have access to. In our example, let's say that you do not have access to "MyFolder", but there is a specific file within MyFolder called "MyFile" that you need access to. You can receive access to only this specific file if you use the "Bypass Traverse Checking" security setting permission.

Bypass Traverse Checking is a setting that is assigned through Group Policy Settings. Therefore, we won't cover it in detail in this article.

# How to Override Folder Permissions with Deny Permission

It is possible to override any permissions with a Deny Permission. In our example, let's say that you do have modify access to "MyFolder" because of your membership in a specific security group, and you are a member of a group that is denying permissions to a specific file in that folder, you will not get access to this file.

Denying permissions overrides any other permissions a user might have. Consider, this is not the recommended method of controlling access to resources.



## Example with Deny Permissions

In Figure above, John is member of Group Sales\_W and Group Sales\_File2\_DW. For folder Sales John will inherit Write permissions from Group Sales\_W. And John will inherit Deny Write permissions for File2 from Group Sales\_File2\_DW.

The results are: John can read and write to File1. He can also read File2, but cannot write to this file because he is a member of Group Sales\_File2\_DW, which grants Deny Write permission to this file.





# Chapter 3

**Share Permissions**

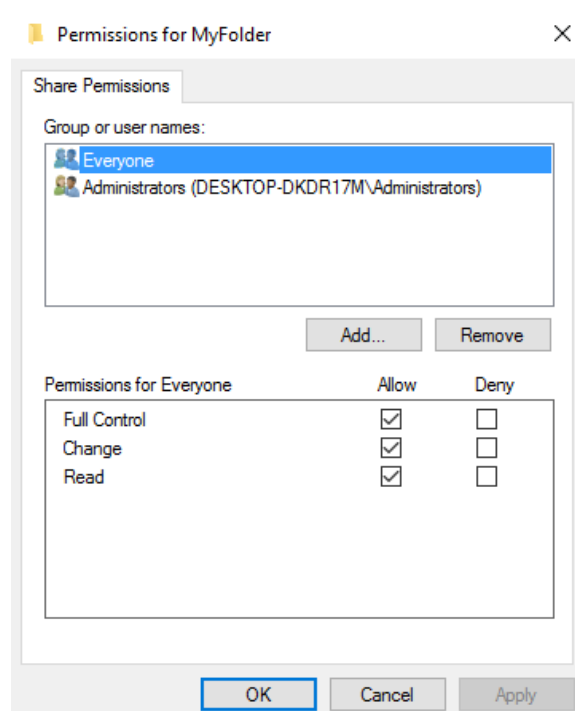
# Understanding Share Permissions

Shared folders are used to provide other users, on your Windows network, access to the contents of those folders. You can only share folders, not individual files.

Share permissions are only applied when a shared folder is accessed over the network. It is a common misconception to think the process works in a different way. When you log into a Windows machine locally (even if a file or folder is shared to other users within the network), every time you access an object, NTFS permissions apply and not share permissions. It does not matter how restrictive share permissions have been set up, if you have access to the object and you are logged into the workstation or server that “owns” the file or folder, you will be granted access.

## *There are three types of share permissions:*

- Full Control: Allows the user to read/execute/write/delete the contents of the folder and manage the folder permissions.
- Change: Allows the user to read/execute/write/delete the contents of the folder, but does not allow the user to modify its permissions.
- Read: Allows the user to read the contents of the folder and its files.



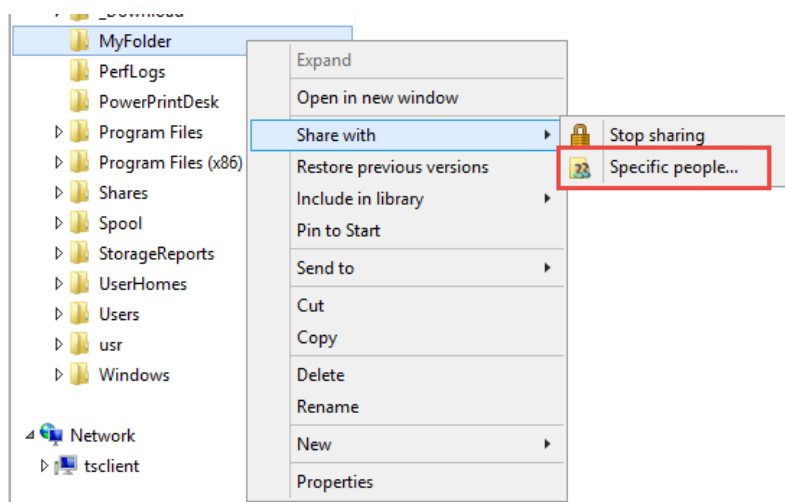
Share permissions for MyFolder

# How to Share Folders

Although the concept behind sharing folders has not changed, there are now two ways to share folders after the release of Windows Server 2012. In previous versions of Windows, the process was isolated and you had to access the folder first or login to the server that owned the folder in order to share it. Windows Server 2012 provided a new interface and a more centralized way of achieving this goal by using the Server Manager. This section will cover both options.

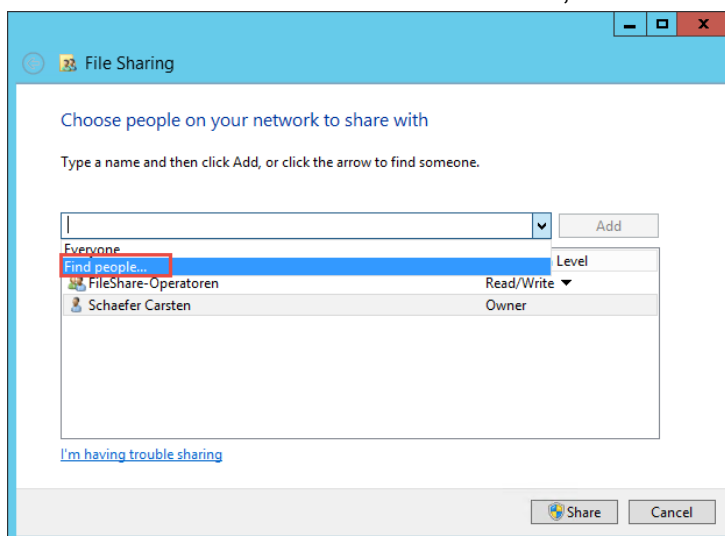
## Sharing Folders – The Traditional Way

Right click the folder. Then click on “Share with” followed by “Specific people”.



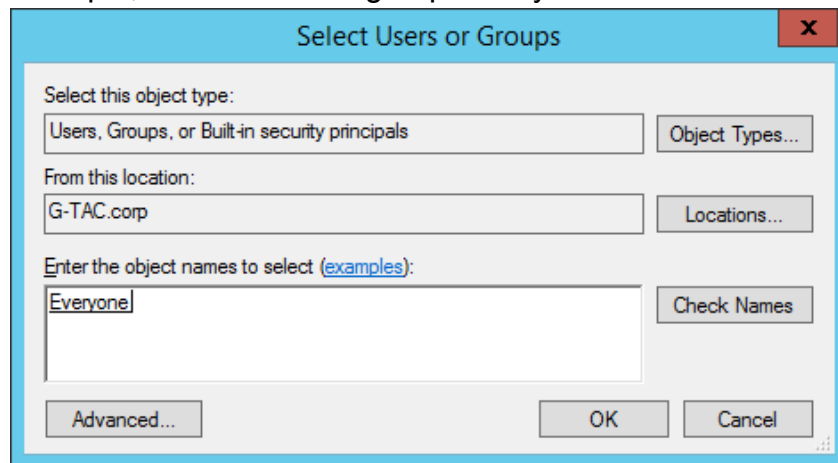
“Share with” Windows menu

Next, you will be presented with the following form, in which you select which users will be granted permission to access the folder. Click on the arrow, followed by “Find people”.



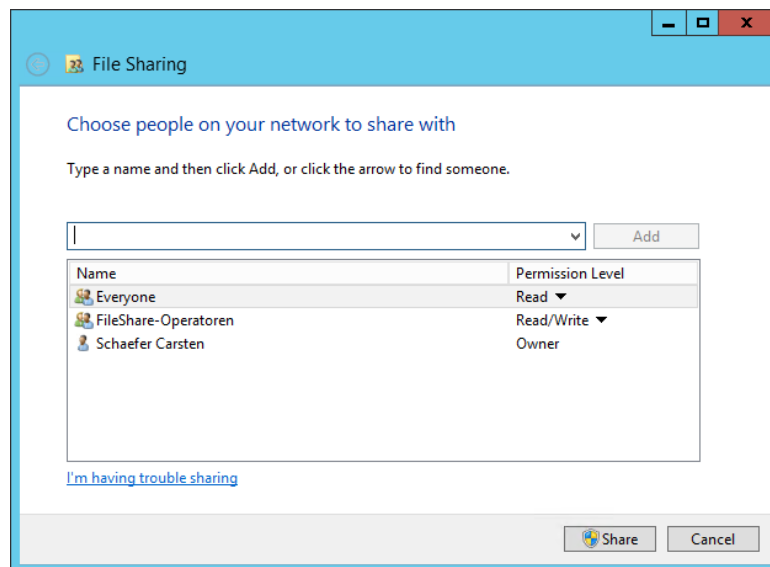
File Sharing properties page

Now you can select the users who will have access to the folder. Type their name and click on the “Check Names” button. Once you are done adding users, press the “OK” button. For this example, we will use the group “Everyone”.



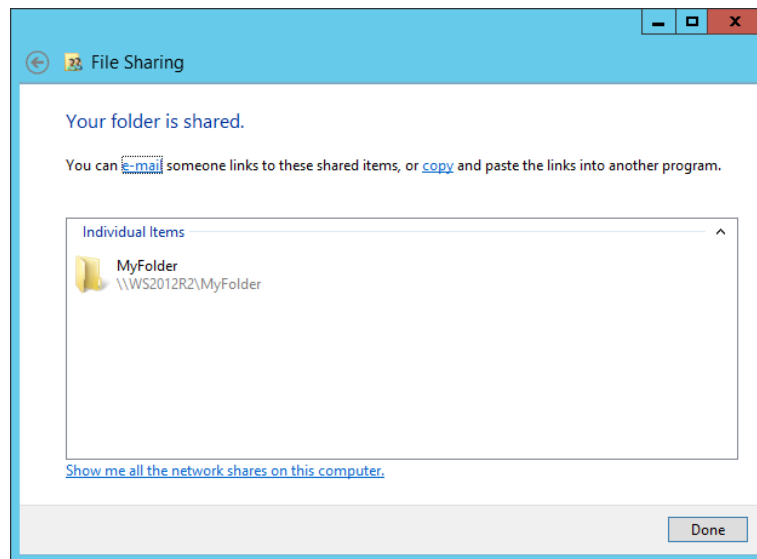
Select users or groups

Click on the “Share” button.



File Sharing properties page

Your folder is now shared to the users/groups that you chose during the previous steps. You can email the link to this folder to the respective users or copy and paste the link as you please.



Shared folder confirmation

This is the link we obtained for our example folder after clicking “copy”:

*MyFolder* (file://WS2012R2/MyFolder)

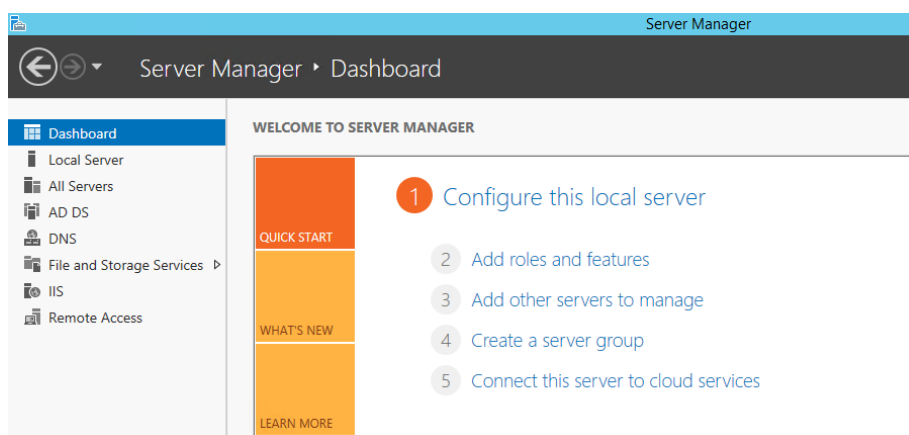


## Sharing Folders – The Server Manager Way

The Server Manager, a graphical interface that integrated many tasks initially done in separate and unorganized ways, was first introduced as a concept in Windows Server 2008. However, it wasn't until the release of Windows Server 2012 that this concept was fully implemented and improved. Now, the Server Manager is frequently the starting point for any task that needs to be performed on the server.

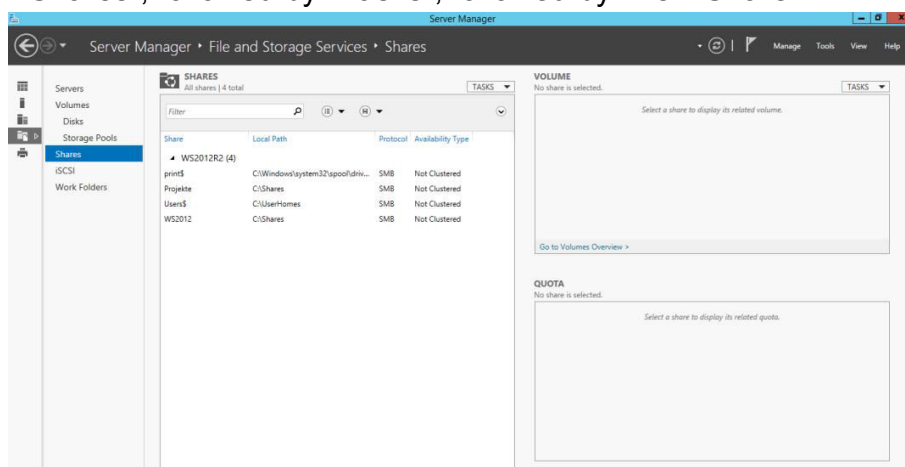
Sharing folders are now fully integrated into the Server Manager, which provides a more centralized means of sharing folders among different users and from different servers, although the concept remains the same.

First, open the Server Manager. On the lefthand side, you will see all the roles your server provides. Click on “File and Storage services”. (Note: you need to have previously installed this role using the “Add roles or features” assistant.)



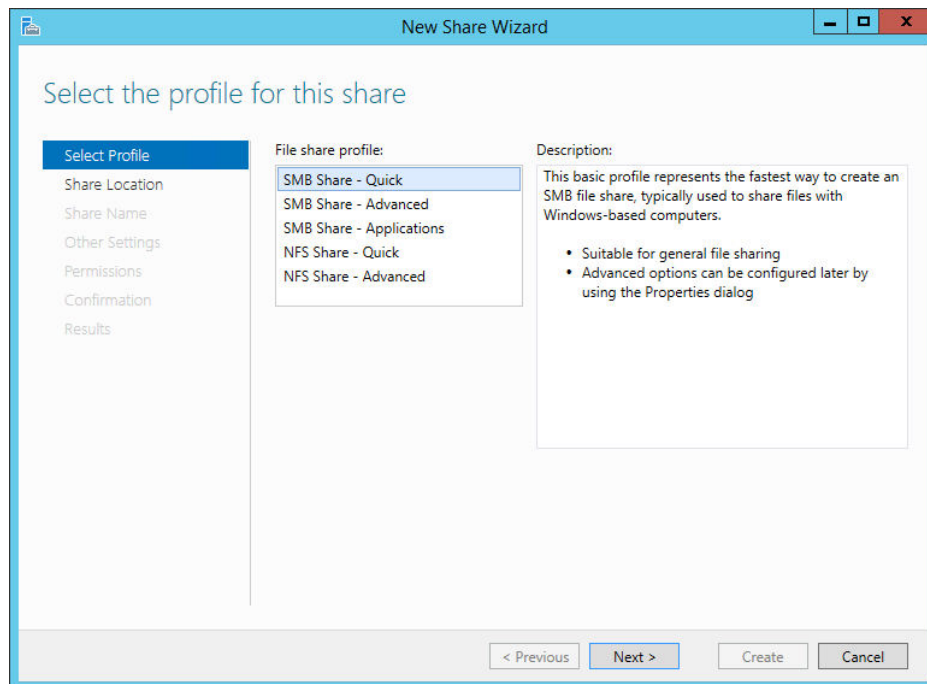
Server manager

Then click on “Shares”, followed by “Tasks”, followed by “New Share”.



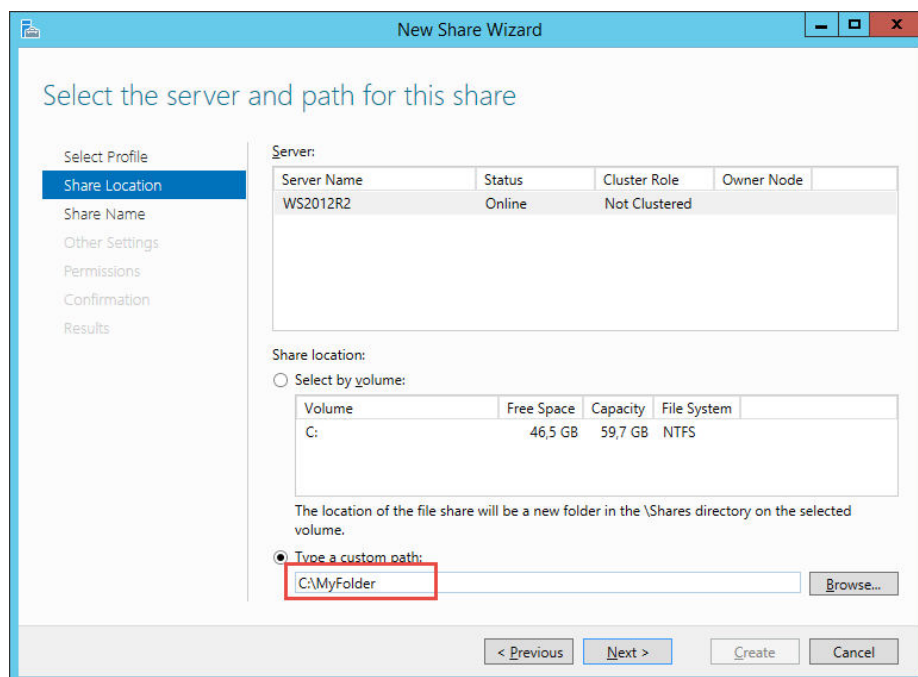
New Share option in the File and Storage Services manager

Now you will choose what you want to share. Each option has a description on the righthand side. For the purpose of this article, choose “SMB Share – Quick” and click “Next” button.



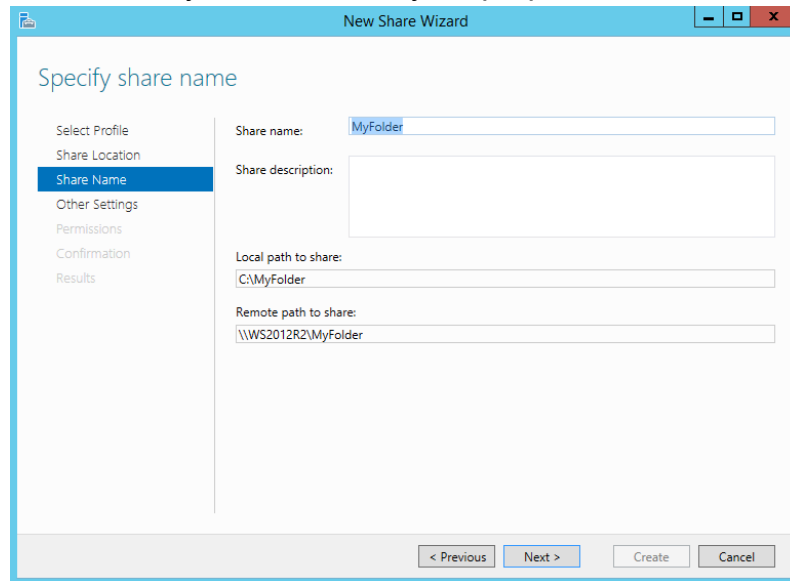
New share wizard

Now select the volume and/or the path for the folder that you want to share and click the “Next” button.



Selecting server and path

Type a name for your shared folder. By default, the share name is the name of the actual folder, but you can choose any name that fits your purposes. Click the “Next” button.

The screenshot shows the 'New Share Wizard' window with the 'Specify share name' step selected in the left-hand navigation pane. The main area contains the following fields: 'Share name:' with the text 'MyFolder', 'Share description:' (empty), 'Local path to share:' with the text 'C:\MyFolder', and 'Remote path to share:' with the text '\\WS2012R2\MyFolder'. At the bottom, there are four buttons: '< Previous', 'Next >', 'Create', and 'Cancel'.

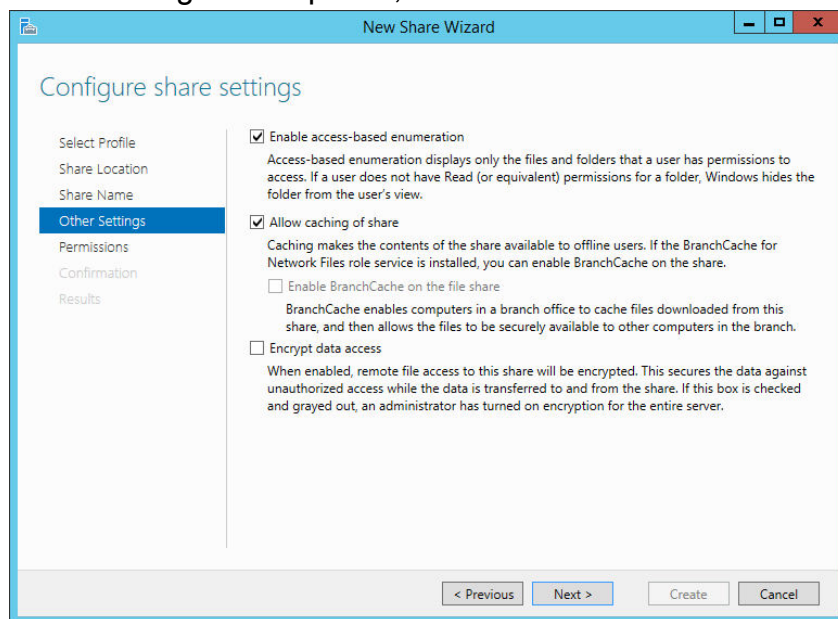
Specify share name

You are now presented with the “Other settings” form. Here you can choose some new features to use with your share that were not present in previous Windows Server versions (or that were only available by downloading separate Windows Server packages). The form contains a brief description of each setting, which will be summarized below to provide better understanding.

***Keep in mind that these options were made to improve security and reliability, thus it is best to choose all of them as we have done in our example.***

- **Enable Access-based Enumeration:** Use this setting when you want to prevent users from seeing other folders besides those they have access to. For example, if you create a folder that contains all “Home” directories containing their personal files for several users, you might want to allow each user to only see the list of folders within their own folder.
- **Allow Caching of Shares:** Use this setting to provide users with an offline copy of their folder. This is particularly helpful if you run into network issues that prevent users from accessing the network path for their folder.
- **Encrypt Data Access:** This setting improves the security of your network shares by encrypting all data in transit. In the case that it is intercepted while being transferred to or from the user workstation to the shared folder, the content will be encrypted and become inaccessible to other sources.

Once you are done selecting these options, click the “Next” button.



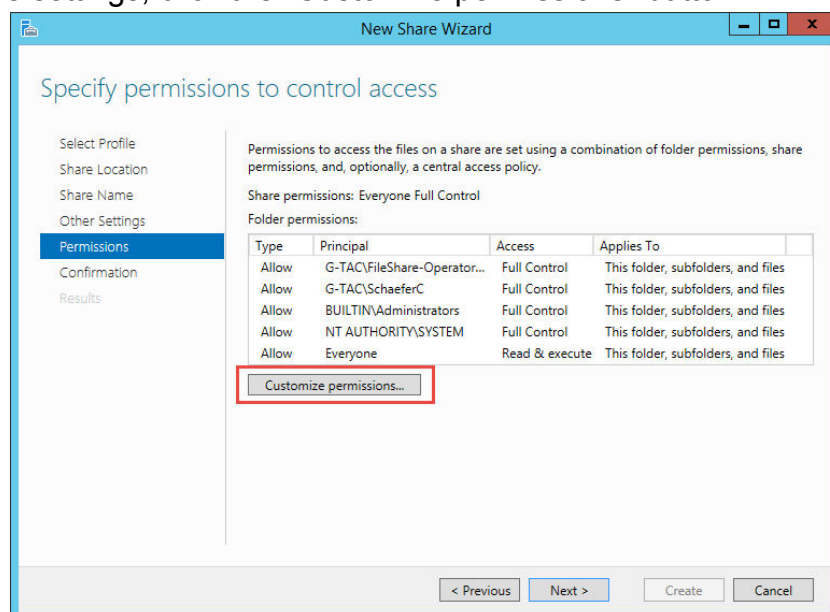
Share settings

The next step is to configure permissions. This step centralizes the setup for both NTFS and share permissions to a single screen.

Remember that, by default, your folder is shared to the group “Everyone” with “Full Control”. This is shown in the tag “Share permissions: Everyone Full Control”.

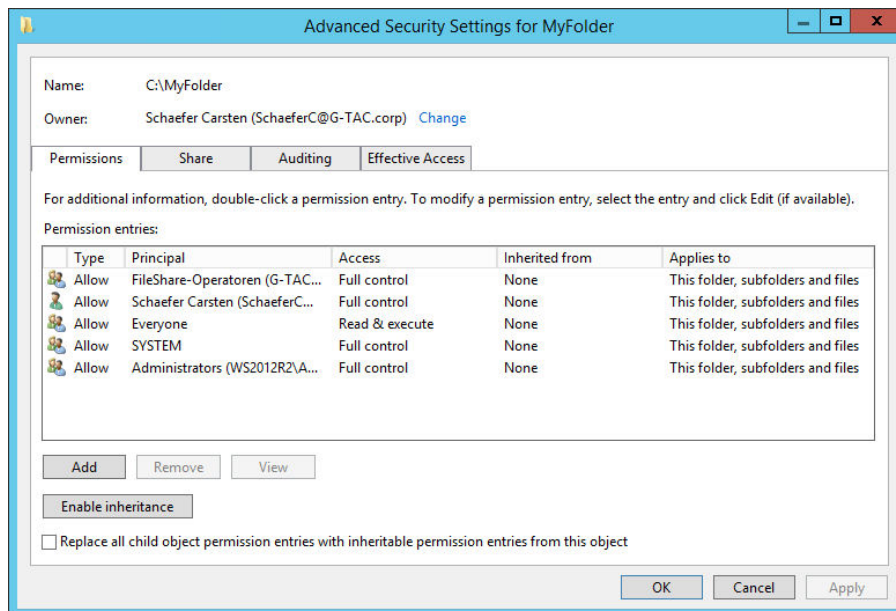
Next, you can see a list of the current NTFS permissions for the folder. Remember that, by default, the folder will inherit permissions from its parent folder.

To change these settings, click the “Customize permissions” button.



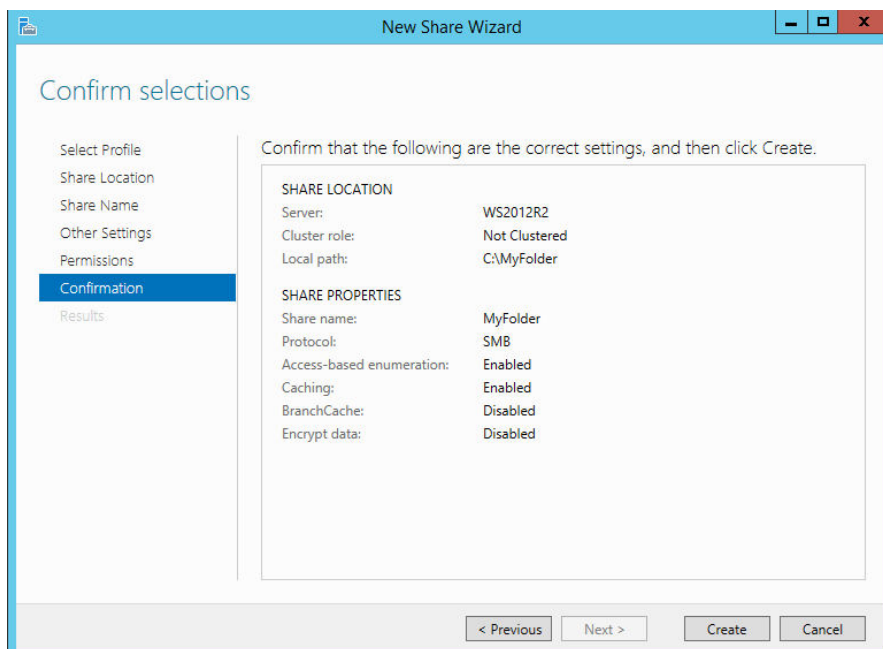
Setting share permissions

Notice this is the same “Advanced Security Settings” properties page that we saw in previous chapters under the “Share” tab, where you can set share permissions for users or groups, thus combining NTFS and share permissions in a single graphical user interface. Once you are ready to set up your permissions, click the “OK” button followed by the “Next” button.



Advanced Security Settings properties page

Finally, you are provided with a summary of the share you just set up. To finish the process, click the “Create” button.



Create share confirmation page

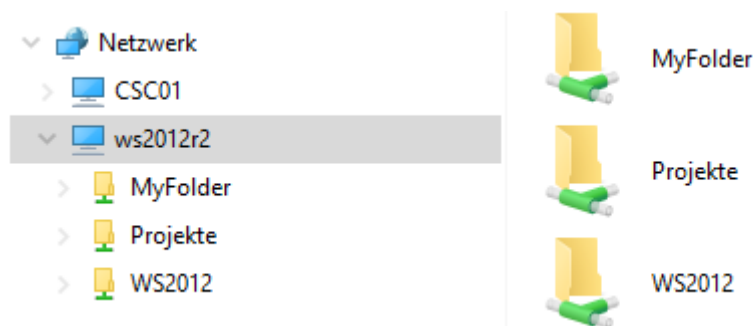
# How to List Shares

In previous editions of Windows (both for Servers and Workstations), once you shared a folder, a hand would be displayed next to that folder's icon.

This changed in Windows Server 2012 for Windows 8 and Windows 10.

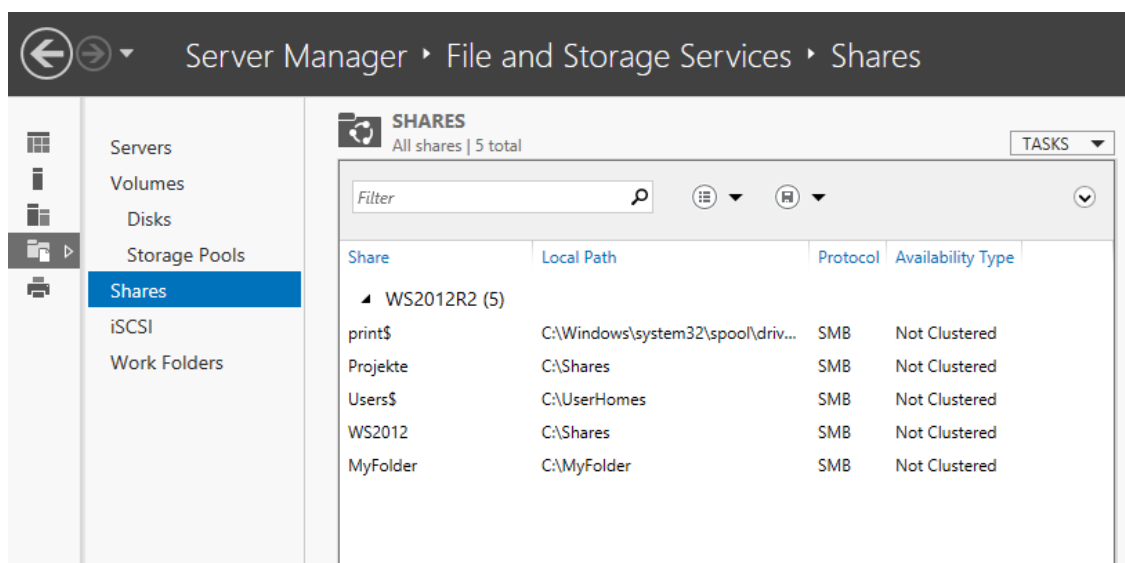
**For workstations, there is now only one way to list your shared folders. For servers, there are two different options.**

- 1) For workstations/servers: Go to "Windows File Explorer". Click on "Network" followed by the name of your server/workstation to display all shared folders.



Shares list in Windows Explorer

- 2) For servers: Go to the Server Manager. Click "File and Storage Services" followed by "Shares" to list all shared folders.



Shares list in Server Manager



# Chapter 4

**Daily Operations**



# NTFS Permissions vs. Share Permissions

## Difference between NTFS Permissions and Share Permissions

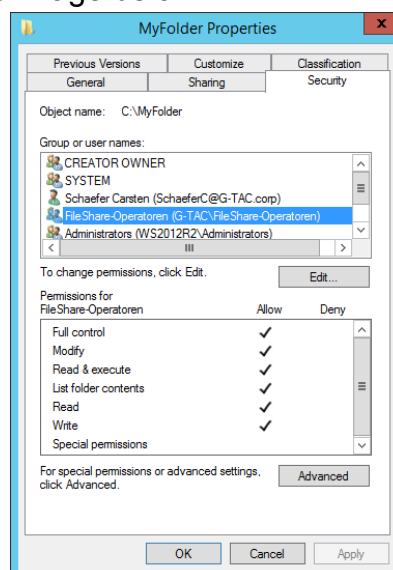
Share permissions are applied when a shared folder is accessed over a network. It is a common misconception to think that share permissions works in a different way. When you log in locally to a Windows machine (even if a file or folder is shared to other users within your network), every time you access an object locally, NTFS permissions apply and share permissions do not apply. It does not matter how restrictive share permissions have been set up on your network, if you have access to the object and you are logged into the workstation or server that “owns” the file or folder, you will be granted access.

## Combining NTFS Permissions and Share Permissions

When using share permissions and folder permissions please keep in mind, that you can apply different NTFS permissions to each folder within a shared folder. Working this way will ensure a permission strategy for each kind of data located in an appropriate folder structure.

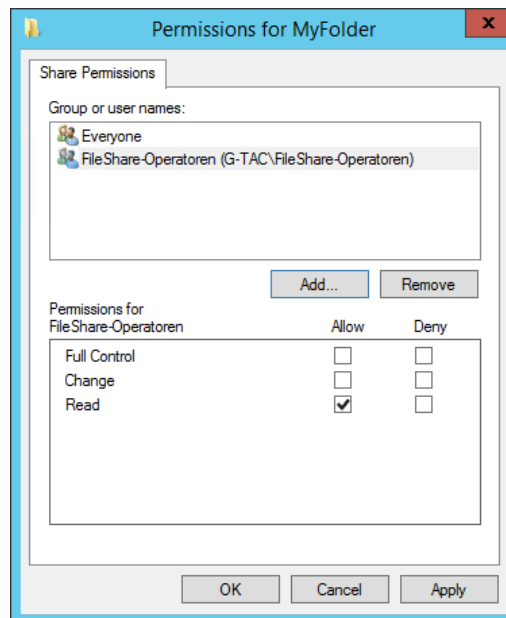
A frequently asked question when managing Windows Server environments is: once you combine share permissions with NTFS permissions, how do these two types of permissions work together? The answer is rather simple and helps you determine the most effective form of permission for a shared folder. Both sets of permissions get applied and the more restrictive of the two takes precedent. To give you a better idea, take a look at the below example.

You give “Full Control” NTFS permissions to the “FileShare-Operatoren” group for a folder called MyFolder, as seen in the image below:



Full Control permissions for MyFolder

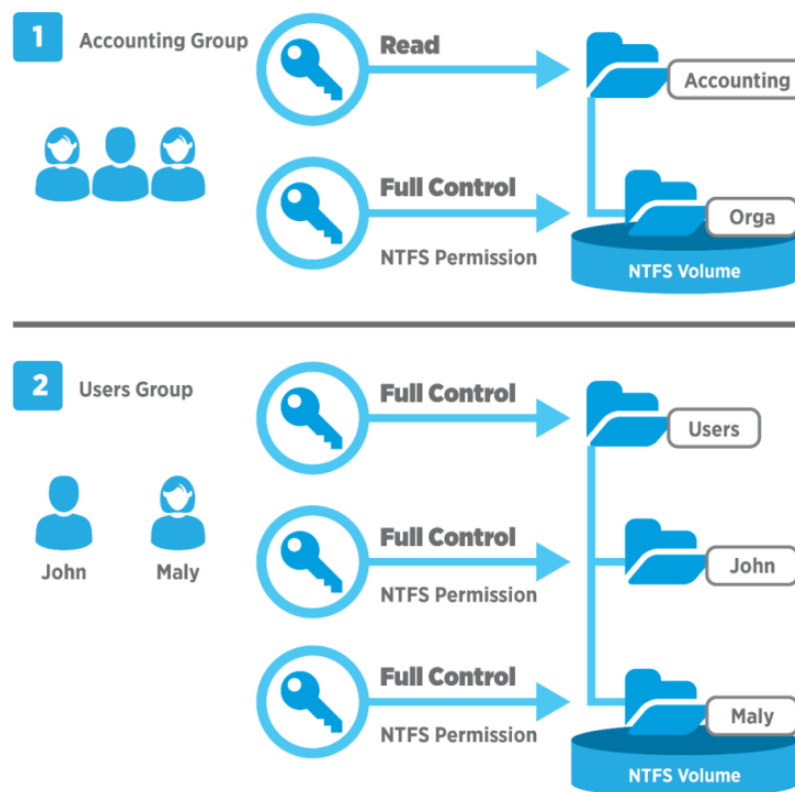
If you share MyFolder within the Windows Network to the “FileShare-Operatoren” group using “Read” permissions and a user that belongs to this group tries to access the folder from the network, that user will only have “Read” access and not “Full Control”. However, if that user then goes to the workstation or server where MyFolder is allocated, he will be granted “Full Control” permissions.



Read Only share permissions for MyFolder

### 3 Examples of Combining Share Permissions with Folder Permissions

In the next two examples we have shared folders on NTFS volumes. These shared folders contain subfolders that have also been assigned NTFS permissions.

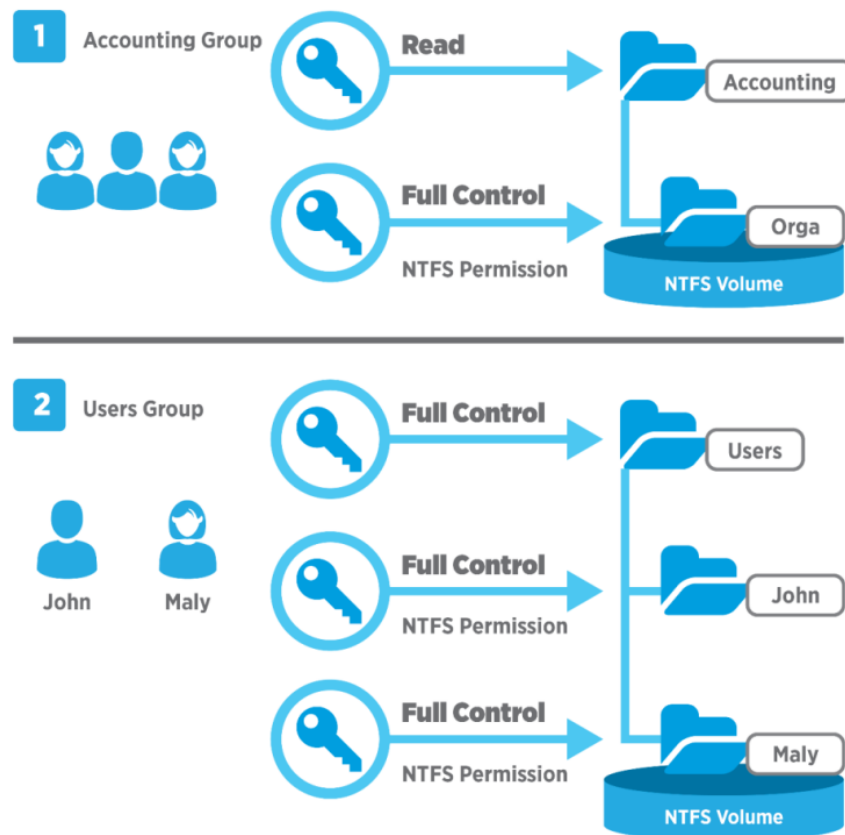


Combined Share and NTFS Permissions

#### First example:

- Accounting folder is shared.
- The Accounting group has the shared folder "Read" permission for this folder and the NTFS "Full Control" permission for the Orga subfolder.

The **effective permissions** for any member of the Accounting group for the subfolder called Orga is Read.



Combined Share and NTFS Permissions

### Second example:

- Users folder contains home folders for each user, here John and Maly.
- Both home folders contains data accessible only to the user for whom the folder is named.
- The Users folder has been shared and the Users group has “Full Control” permission for the Users folder.
- John and Maly have the NTFS “Full Control” permission for their home folder only and no NTFS permissions for other folders.
- Boths are members of the Users group.

The **effective permissions** for John and Maly for their own home folder are Full Control. But John has no access to Maly’s home folder and Maly has no access to John’s home folder.

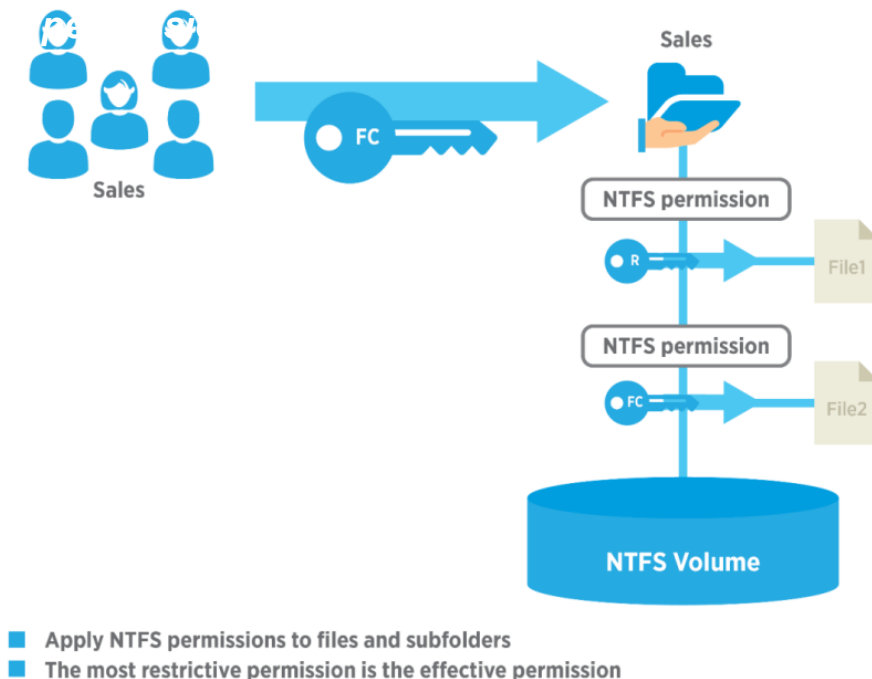
### Third example:

The group Sales has these permissions:

- NTFS Permissions Full Control for shared folder Sales
- NTFS Permissions Read for File1
- NTFS Permissions Full Control for File 2

The **effective permissions** are:

- The member of this group are granted only Read access to File1 because it is the most restrictive permission.
- And they are granted Full Control to File2 because both permission assignments are at the same level.



Effective NTFS Permissions

# Copying and Moving Files and Folders

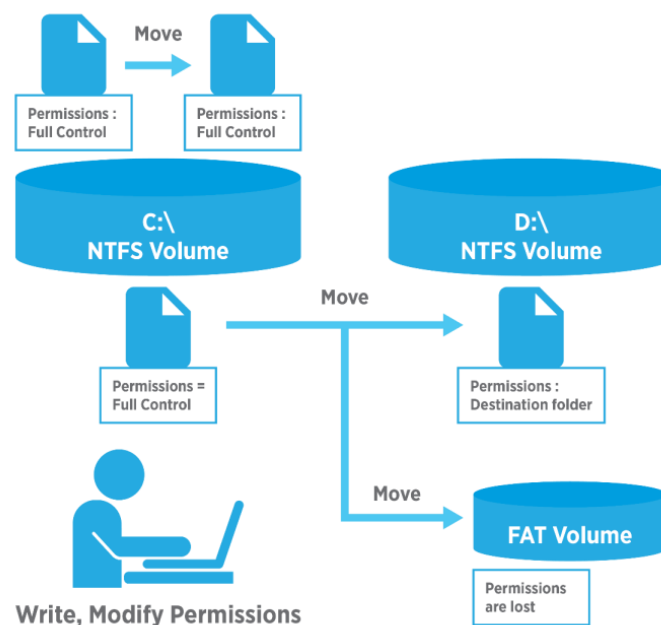
Now that we know how to manage permissions for folders or files, it's a good time to ask ourselves a question: What happens to permissions if I copy or move the files or folders? The answer is: it depends.

To give you a clearer explanation, consider the following three scenarios. Let's assume that you are going to copy "D:\MyFolder" and let's assume that "D" has an NTFS format.

## Scenario A: Copy D:\MyFolder to E:\ (E:\ is a FAT volume)

### Results:

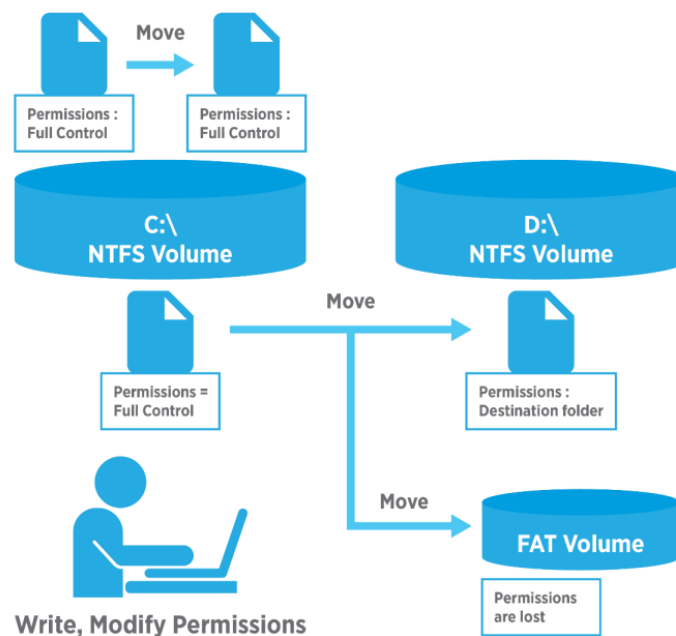
- When you copy files or folders to FAT volumes, the folders and files lose their NTFS permissions because FAT volumes don't support NTFS permissions.



### Scenario B: Move D:\MyFolder to D:\MyFiles

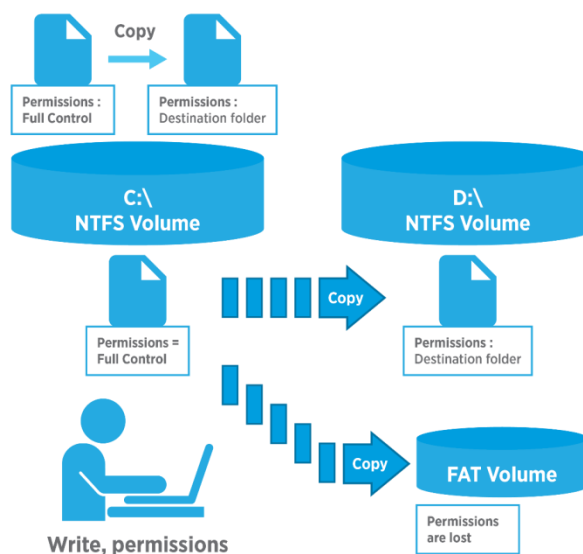
#### Results:

- The file or folder retains its original permissions.
- You must have the “Write” permission set up for the destination folder to move files and folders into that folder.
- You must have the “Modify” permission set up for the source file or folder. The “Modify” permission is required to move a file or folder because Windows 2000 deletes files and folders from the source folder after they are copied to the destination folder.
- You become the creator and owner.



**Scenario C: Copy D:\MyFolder to F:\MyFolder (F:\ is an NTFS volume)****Results:**

- The file or folder inherits the permissions of the destination folder.
- You must have the “Write” permission set up for the destination folder to move files and folders into that folder.
- You must have the “Modify” permission set up for the source file or folder. The “Modify” permission is required to move a file or folder because Windows XP Professional deletes files and folder from the source folder after they are copied to the destination folder.
- You become the creator and owner.





## Powershell and NTFS Permissions

Automation and scripting become more important for system administrators with every Windows Server version. One of the key concepts behind Windows Server 2012 is the capability to do almost everything you can do on a GUI. Using Powershell scripts and NTFS permissions are no exception.

The following table lists the most common use case scenarios that a system administrator can have and how to perform operations such as using scripts without going to the common graphical user interface (GUI). This is just to get you started on managing ACL's with Windows Powershell. Remember, these scripts can become as complicated as you want them to be.

Task	Powershell Script
Reading Permissions of a Single File or Folder	<pre>((Get-Item D:\MyFolder).GetAccessControl('Access')).Access</pre>
Modifying User Permissions on a Folder	<pre>\$HomeFolders = Get-ChildItem C:\Homefolders -Directory foreach (\$HomeFolder in \$HomeFolders) {     \$Path = \$HomeFolder.FullName     \$Acl = (Get-Item \$Path).GetAccessControl('Access')     \$Username = \$HomeFolder.Name     \$Ar = New-Object     System.Security.AccessControl.FileSystemAccessRule(\$Username,     'Modify', 'ContainerInherit, ObjectInherit', 'None', 'Allow')     \$Acl.SetAccessRule(\$Ar)     Set-Acl -path \$Path -AclObject \$Acl }</pre>

Basic Powershell scripts to manage NTFS permissions

## Best Practices

- A common practice by many businesses is to share a folder by giving full access to a group made up of everyone, then control who can access that folder using NTFS permissions.
- Always try to share folders with groups instead of individual people, as this makes administration tasks far simpler.
- To consolidate administration and group files into application, data, and home folders, centralize all home and public folders separately from your applications and operating system. Doing so provides the following benefits: a) permissions may only be assigned to folders, not individual files and b) backing up will be less complex because you will not need to back up application files, as all home and public folders will be consolidated in one location.
- When you assign permissions for working with data or application folders, assign the “Read & Execute” permission to the Users group and Administrators group. This will prevent application files from being accidentally deleted or damaged by users or viruses.
- Always assign the most restrictive permissions that still allow users to perform required tasks. For example, if users only need to read information in a folder and should never delete or create files, assign the “Read” permission.
- Organize your resources so that folders with the same security requirements are located within one folder. For example, if users require “Read” permission for several application folders, store those folders within a single folder. This will allow you to share that larger folder instead of sharing each individual application folder.
- Use intuitive share names so that users can easily recognize and locate resources. For example, for the Application folder, use “Apps” as the share name. You should only use share names that can be used across all client operating systems.

