Recent improvements in using NFSv4 ACLs with Samba

Alexander Werth IBM

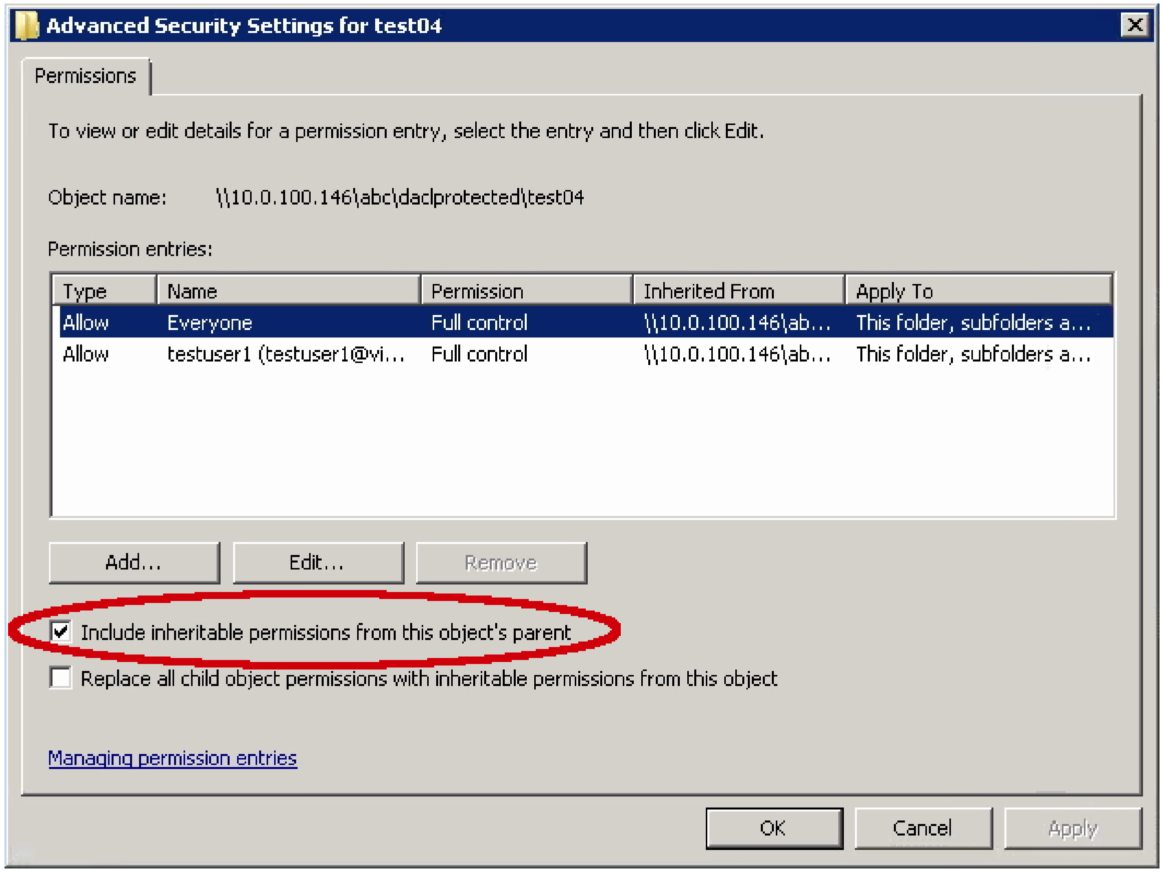
Agenda

Recent improvements in using NFSv4 ACLs with Samba

- **Detailed look at individual improvements.**
  - **ACL control flags**
  - **No propagate inherit flag**
  - **Permission improvements**
  - **Order of ACLs**
  - **Other and noteable older changes**

- **What's left?**

DACL protected bit



Advanced Security Settings for test04

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

Object name: \\10.0.100.146\abc\daclprotected\test04

Permission entries:

Type	Name	Permission	Inherited From	Apply To
Allow	Everyone	Full control	\\10.0.100.146\ab...	This folder, subfolders a...
Allow	testuser1 (testuser1@vi...	Full control	\\10.0.100.146\ab...	This folder, subfolders a...

[✓] Include inheritable permissions from this object's parent

[ ] Replace all child object permissions with inheritable permissions from this object

- Used to „break“ inheritance at a particular level in the folder hierarchy.
- Used by the windows client when changing ACLs.
- Stored in ACL control flags.
- Checkbox is unchecked if the DACL protected bit is set, or the inherited ACL entries are different from the inheriting parent ACL entries.

DACL protected bit in NFS4.1 and Windows

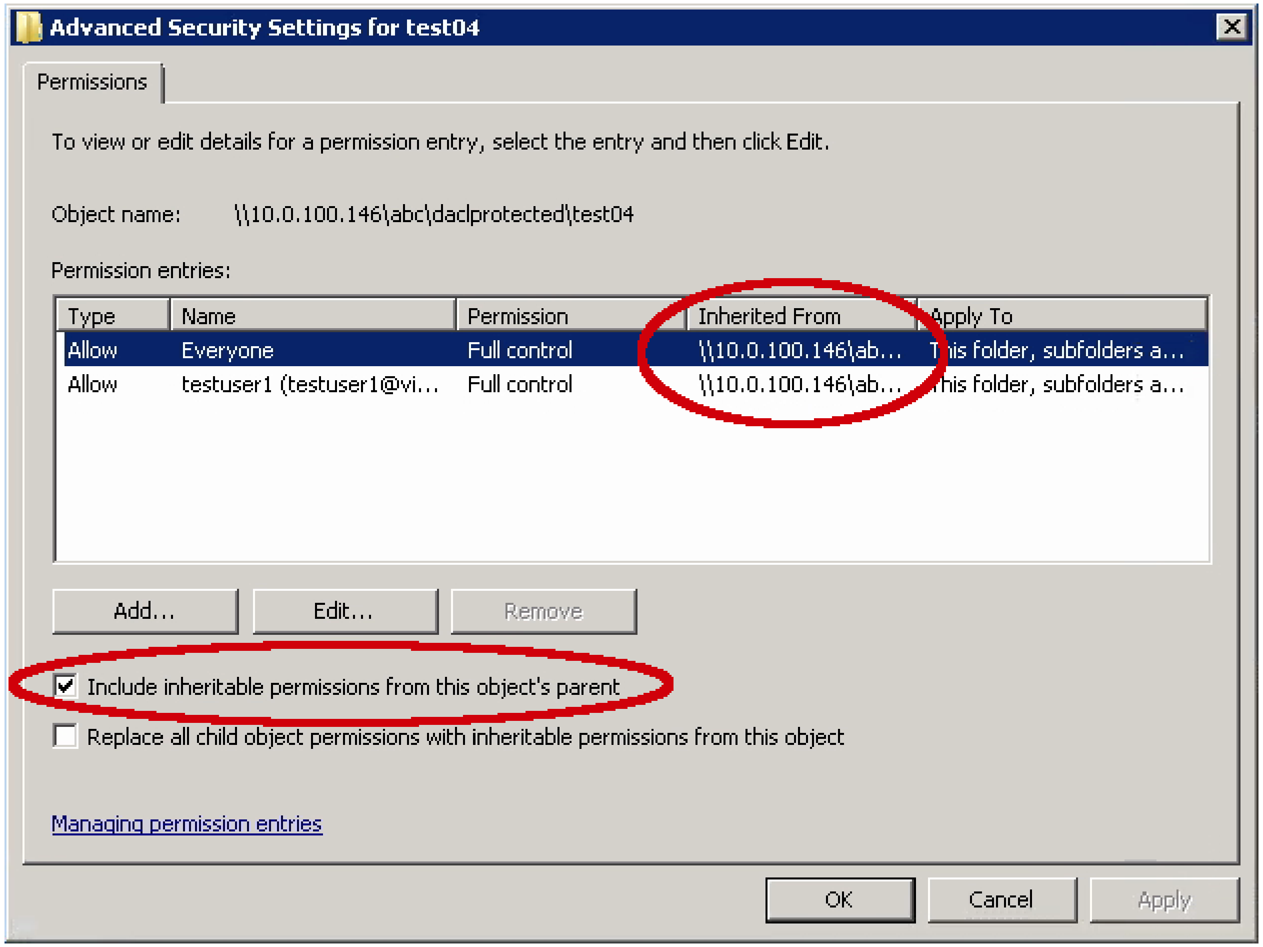
The DACL protected bit is part of the Security Descriptor and the NFS4.1 spec.

ACL control flags from the NFS4.1 spec:
- ACL4_AUTO_INHERIT
- ACL4_PROTECTED
- ACL4_DEFAULTED

Only a few of the ACL control flags in the Security Descriptor are persistent and stored to disk.

Data passed through nfs4_acl.c common code to vfs modules for storage and retrieval.

What happens without the DACL protected bit 1/4?


Advanced Security Settings for test04

Permissions

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

Object name: \\10.0.100.146\abc\daclprotected\test04

Permission entries:

Type	Name	Permission	Inherited From	Apply To
Allow	Everyone	Full control	\\10.0.100.146\ab...	This folder, subfolders a...
Allow	testuser1 (testuser1@vi...	Full control	\\10.0.100.146\ab...	This folder, subfolders a...

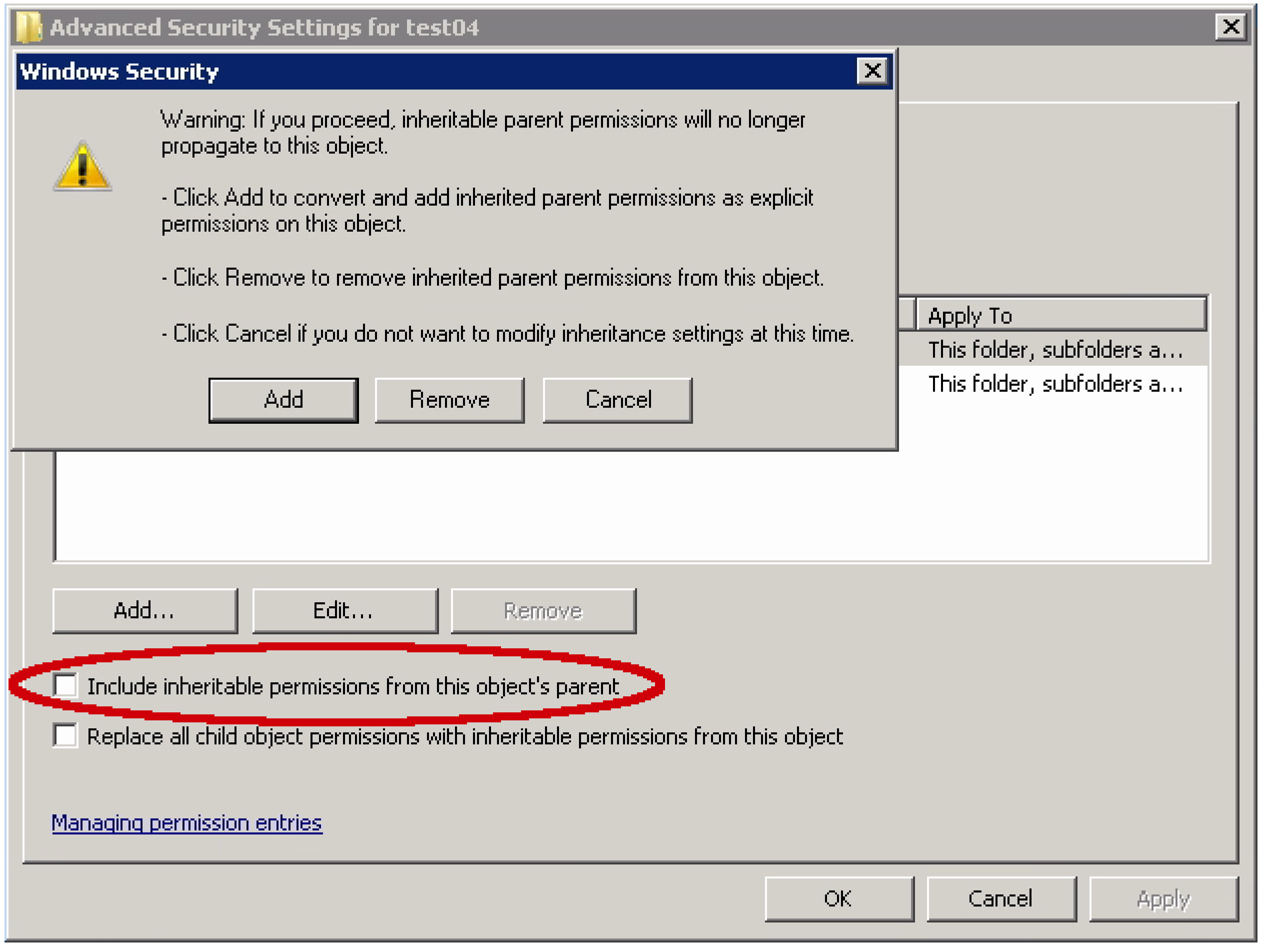
☑ Include inheritable permissions from this object's parent

☐ Replace all child object permissions with inheritable permissions from this object

Managing permission entries

- **Initially the checkbox is checked and ACL entries are inherited.**

What happens without the DACL protected bit 2/4?


Advanced Security Settings for test04

Windows Security

Warning: If you proceed, inheritable parent permissions will no longer propagate to this object.

- Click Add to convert and add inherited parent permissions as explicit permissions on this object.

- Click Remove to remove inherited parent permissions from this object.

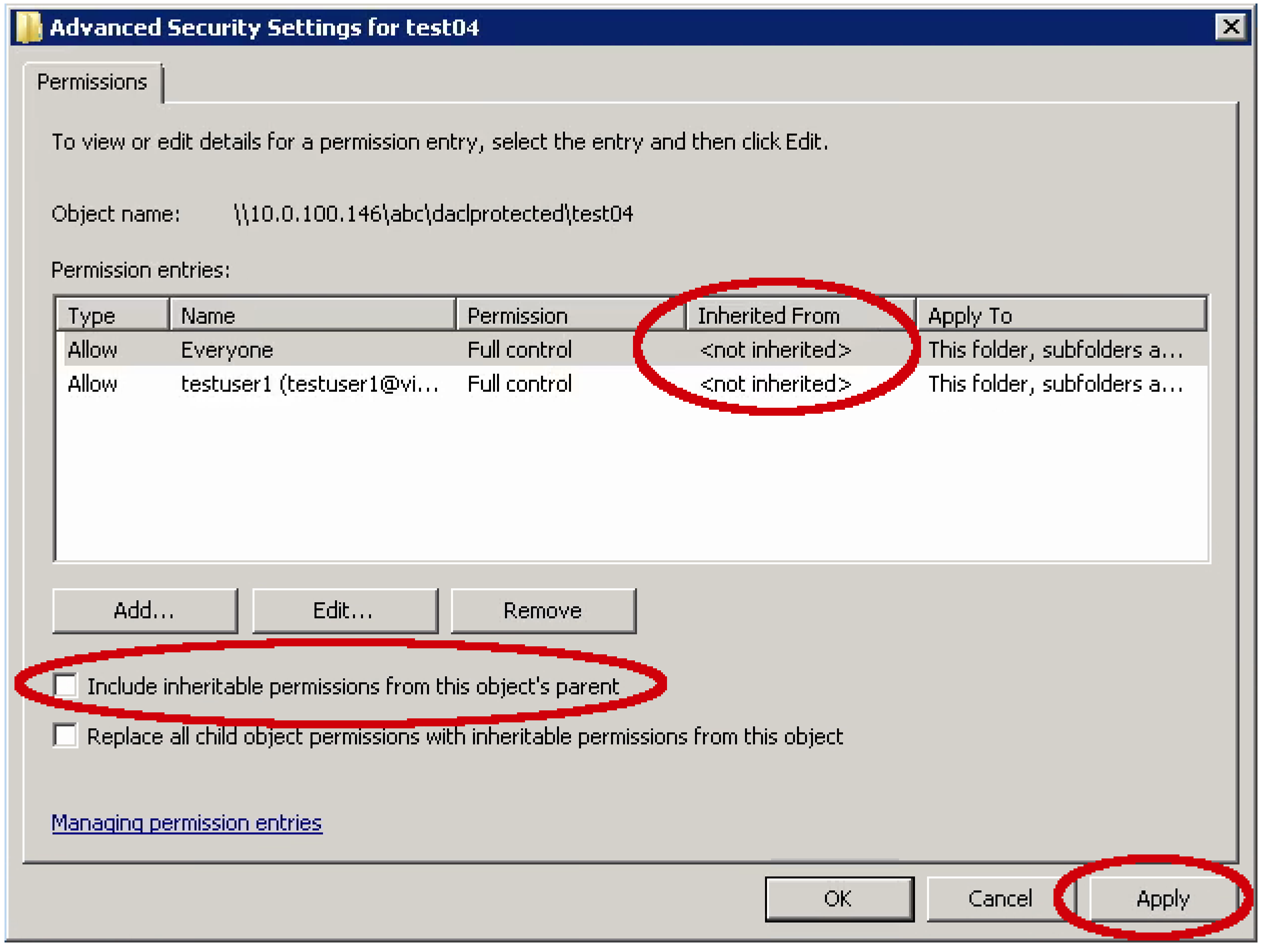
- Click Cancel if you do not want to modify inheritance settings at this time.

☐ Include inheritable permissions from this object's parent

☐ Replace all child object permissions with inheritable permissions from this object

- **Unchecking the checkbox opens a dialog to add or remove the inherited entries.**

What happens without the DACL protected bit 3/4?



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

Object name: \\10.0.100.146\abc\daclprotected\test04

Permission entries:

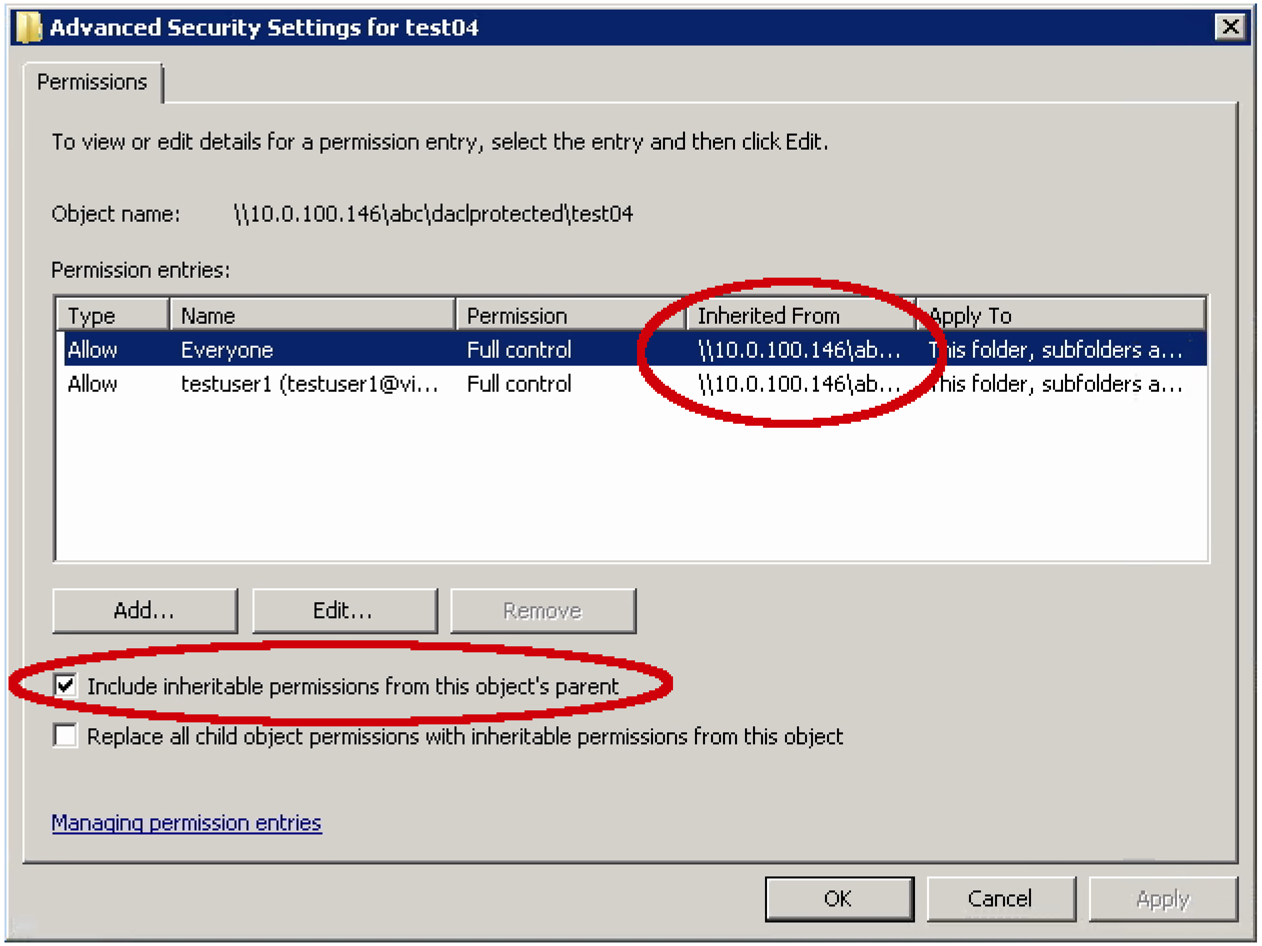
Type	Name	Permission	Inherited From	Apply To
Allow	Everyone	Full control	<not inherited>	This folder, subfolders a...
Allow	testuser1 (testuser1@vi...	Full control	<not inherited>	This folder, subfolders a...

☐ Include inheritable permissions from this object's parent

☐ Replace all child object permissions with inheritable permissions from this object

- **Selecting to add the entries adds the same entries as not inherited. Apply can be selected now.**

What happens without the DACL protected bit 4/4?



Advanced Security Settings for test04

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

Object name: \\10.0.100.146\abc\daclprotected\test04

Permission entries:

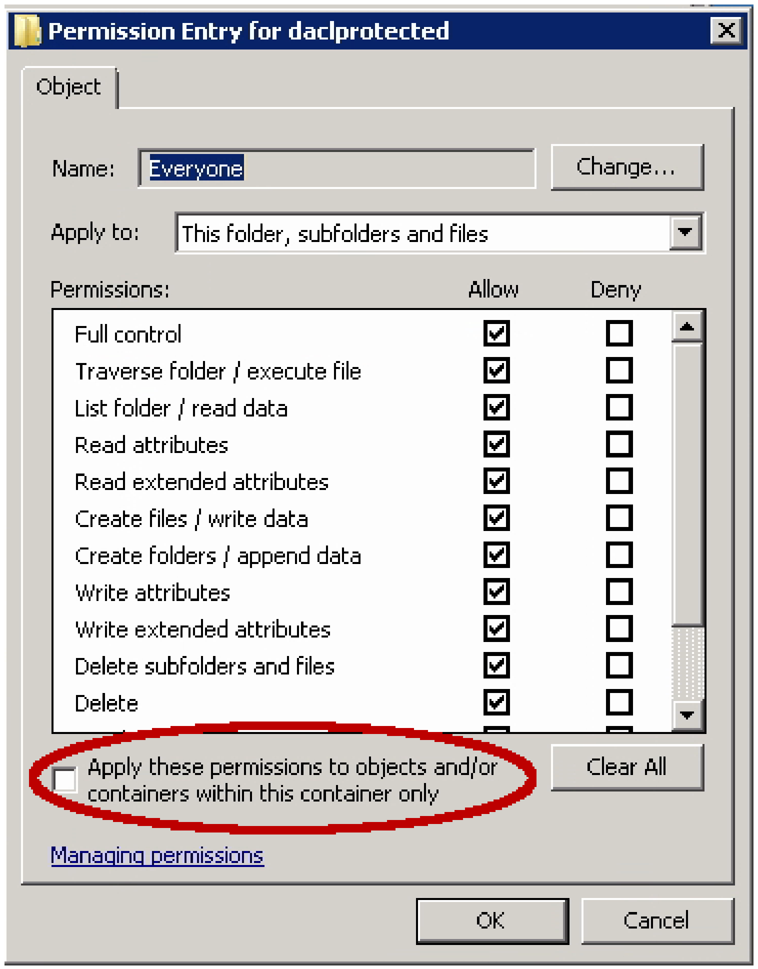
Type	Name	Permission	Inherited From	Apply To
Allow	Everyone	Full control	\\10.0.100.146\ab...	This folder, subfolders a...
Allow	testuser1 (testuser1@vi...	Full control	\\10.0.100.146\ab...	This folder, subfolders a...

[✓] Include inheritable permissions from this object's parent

[ ] Replace all child object permissions with inheritable permissions from this object

- **After selecting apply the changes are apparently not applied. The checkbox is again checked, the ACL entries are marked as inherited, and the remove button is again greyed out.**

No propagate inherit semantics implemented


Permission Entry for daclprotected

Object

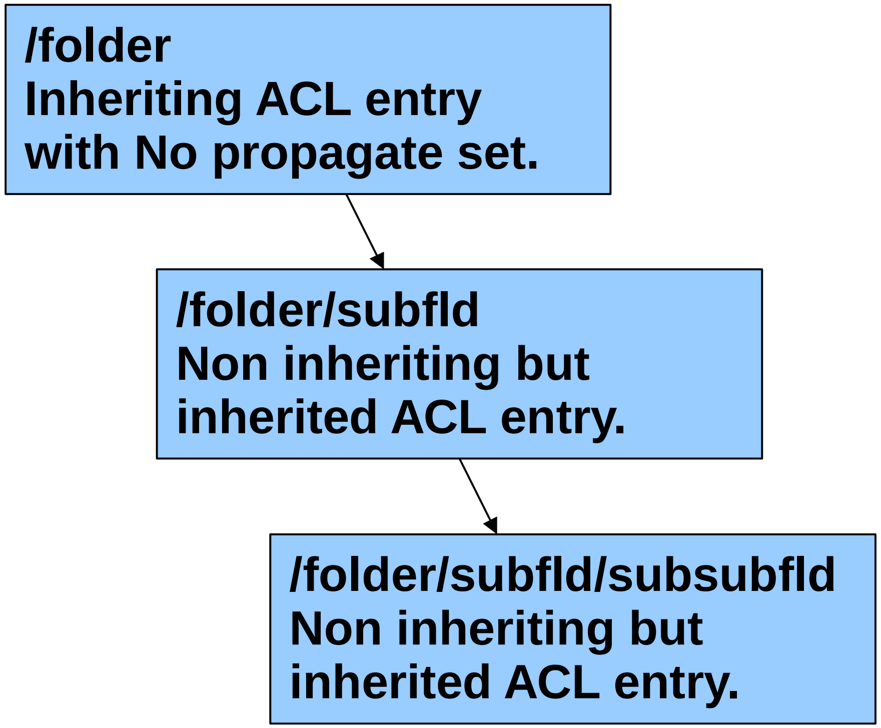
Name: Everyone

Apply to: This folder, subfolders and files

Permissions:	Allow	Deny
Full control	✓	☐
Traverse folder / execute file	✓	☐
List folder / read data	✓	☐
Read attributes	✓	☐
Read extended attributes	✓	☐
Create files / write data	✓	☐
Create folders / append data	✓	☐
Write attributes	✓	☐
Write extended attributes	✓	☐
Delete subfolders and files	✓	☐
Delete	✓	☐

☐ Apply these permissions to objects and/or containers within this container only

- If set the ACL entry will inherit only to the immediate subfolder.


/folder
Inheriting ACL entry with No propagate set.

/folder/subfld
Non inheriting but inherited ACL entry.

/folder/subfld/subsubfld
Non inheriting but inherited ACL entry.

No propagate inherit semantics implemented

The nopropagate inherit flag is part of the NFS 4.1 spec.

It's stored in the ACL entry (ACE) flags.

The flag is already passed to the NFS4 filesystems.

GPFS 4.1 supports the no propagate inherit flag.

The semantic to drop the inheriting flags on ACEs of new files is followed.

NTFS permissions required on Windows™ 2003 Server

ACL permission behavior / Operation	Traverse folder / execute file [6]	List folder / read data	Read attributes	Read extended attributes [7]	Create files / write data	Create folders / append data	Write attributes	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file	X												
List folder		X											
Read data from file		X	X	X									
Read attributes			X										
Create file					X								
Create folder						X							
Write data to file					X	X	X	X			X		
Write file/folder attributes							X						
Delete file/folder		P	X		P				P or	X			
Read file/folder permissions											X		
Write file/folder permissions												X	
Take file/folder ownership													X

X: Required   P: Required on parent folder

2013 Comparison of Samba & GPFS™ against Windows

ACL permission behavior / Operation	Traverse folder / execute file [6]	List folder / read data	Read attributes	Read extended attributes [7]	Create files / write data	Create folders / append data	Write attributes	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file		X	T										
List folder		X	Files										
Read data from file		X	X,T	X							X		
Read attributes			X,T										
Create file			T		X								
Create folder			T			X							
Write data to file		X	X,T		X	X	X	X			X		
Write file/folder attributes		X	T				X				X		
Delete file/folder		X,P	X,T		P				P or	X			
Read file/folder permissions		X	T								X		
Write file/folder permissions		X	T								X	X	
Take file/folder ownership		X	T								X		X

X: Required P: Required on parent folder T: Required for traversal

Read ACL permission not required for Samba internal processing

To perform access checks Samba must read the ACLs and construct a security descriptor.

With the vfs_gpfs module the ACLs for internal processing are read as root.

All of the cases where the read ACL permission was required are ok now.

ACL permission behavior Operation	Traverse folder / execute file [6]	List folder / read data	Read attributes	Read extended attributes [7]	Create files / write data	Create folders / append data	Write attributes	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file	🟥	X	T										
List folder		X	Files										
Read data from file		X	X,T	X							🟩		
Read attributes			X,T										
Create file			T		X								
Create folder			T			X							
Write data to file		X	X,T		X	X	X	X			X		
Write file/folder attributes		X	T				X				🟩		
Delete file/folder		X,P	X,T		P				P or	X			
Read file/folder permissions		X	T								X		
Write file/folder permissions		X	T								🟩	X	
Take file/folder ownership		X	T								🟩		X

GPFS fix to keep in use stat cache entries

Last years GPFS version dropped stat cache entries in case someone without the required priviledges tried to access them.

This caused problems for system calls by also dropping in use stat cache entries.

The current GPFS versions don't fail a stat cache entry revalidation based on a missing read attribute permission.

ACL permission behavior Operation	Traverse folder / execute file [6]	List folder / read data	Read attributes	Read extended attributes [7]	Create files / write data	Create folders / append data	Write attributes	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file	🟥	X	🟩										
List folder		X	Files										
Read data from file		X	X	X							X		
Read attributes			X										
Create file			🟩		X								
Create folder			🟩			X							
Write data to file		X	X		X	X	X	X			X		
Write file/folder attributes		X	🟩				X				X		
Delete file/folder		X,P	X		P				P or	X			
Read file/folder permissions		X	🟩								X		
Write file/folder permissions		X	🟩								X	X	
Take file/folder ownership		X	🟩								X		X

But Read attribute still required for inode lookups in path

For initial access the read attribute permission is still required for directories in the path. Unless it's already cached by someones else access.

Not as bad since it will fail at open time.

The topmost folders are usually in the stat cache and now they stay there.

The user specific directories usually already have the required permissions.

ACL permission behavior Operation	Traverse folder / execute file [6]	List folder / read data	Read attributes	Read extended attributes [7]	Create files / write data	Create folders / append data	Write attributes	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file	🟥	X	T										
List folder		X	Files										
Read data from file		X	X,T	X							X		
Read attributes			X,T										
Create file			T		X								
Create folder			T			X							
Write data to file		X	T		X	X	X	X			X		
Write file/folder attributes		X	T				X				X		
Delete file/folder		X,P	X,T		P				P or	X			
Read file/folder permissions		X	T								X		
Write file/folder permissions		X	T								X	X	
Take file/folder ownership		X	T								X		X

Read Data permission not required for some folder operations

Not all operations actually require an open file handle.

The directory open code now skips the Posix open in these cases.

This is similar to a previous implementation for files.

Most cases where the read data permission was required are ok now.

ACL permission behavior Operation	Traverse folder / execute file [6]	List folder / read data	Read attributes	Read extended attributes [7]	Create files / write data	Create folders / append data	Write attributes	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file	🟥	X	T										
List folder		X	Files										
Read data from file		X	X,T	X							X		
Read attributes			X,T										
Create file			T		X								
Create folder			T			X							
Write data to file		🟩	X,T		X	X	X	X			X		
Write file/folder attributes		🟩	T				X				X		
Delete file/folder		P	X,T		P				P or	X			
Read file/folder permissions		🟩	T								X		
Write file/folder permissions		🟩	T								X	X	
Take file/folder ownership		🟩	T								X		X

fstat is run as root

To list a directory a stat call for each entry is required.

This stat call is now run with root permissions.

This fixes cases where the directory view depends on the state of the stat cache.

Listing directories works fine now.

ACL permission behavior / Operation	Traverse folder / execute file [6]	List folder / read data	Read attributes	Read extended attributes [7]	Create files / write data	Create folders / append data	Write attributes	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file	🟥	X	T										
List folder		x	🟩										
Read data from file		x	X,T	X							X		
Read attributes			X,T										
Create file			T		X								
Create folder			T			X							
Write data to file		x	X,T		X	X	X	X			X		
Write file/folder attributes		x	T				X				X		
Delete file/folder		x,P	X,T		P				P or	X			
Read file/folder permissions		x	T								X		
Write file/folder permissions		x	T								X	X	
Take file/folder ownership		x	T								X		X

Optional strict execute permission checking

Samba 4 does check the execute permission if requested by th access mask.

New option „acl allow execute always“ can be used to get Samba 3.6 behavior.

ACL permission behavior Operation	Traverse folder / execute file [6]	List folder / read data	Read attributes	Read extended attributes [7]	Create files / write data	Create folders / append data	Write attributes	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file	X	X	T										
List folder		x	Files										
Read data from file		x	X,T	X							X		
Read attributes			X,T										
Create file			T		X								
Create folder			T			X							
Write data to file		x	X,T		X	X	X	X			X		
Write file/folder attributes		x	T				X				X		
Delete file/folder		X,P	X,T		P				P or	X			
Read file/folder permissions		x	T								X		
Write file/folder permissions		x	T								X	X	
Take file/folder ownership		x	T								X		X

2014 Comparison of Samba & GPFS against Windows

ACL permission behavior Operation	Traverse folder / execute file [6]	List folder / read data	Read attributes	Read extended attributes [7]	Create files / write data	Create folders / append data	Write attributes	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file	🟩 X	🟥 X	T										
List folder		X	T										
Read data from file		X	X,T	X							🟩		
Read attributes			X,T										
Create file			T		X								
Create folder			T			X							
Write data to file		🟩	T		X	X	X	X			X		
Write file/folder attributes		🟩	T				X				🟩		
Delete file/folder		🟩 P	X,T		P				P or	X			
Read file/folder permissions		🟩	T								X		
Write file/folder permissions		🟩	T								🟩	X	
Take file/folder ownership		🟩	T								🟩		X

X: Required P: Required on parent folder T: Required for traversal

Out of order deny entries after chmod fixed

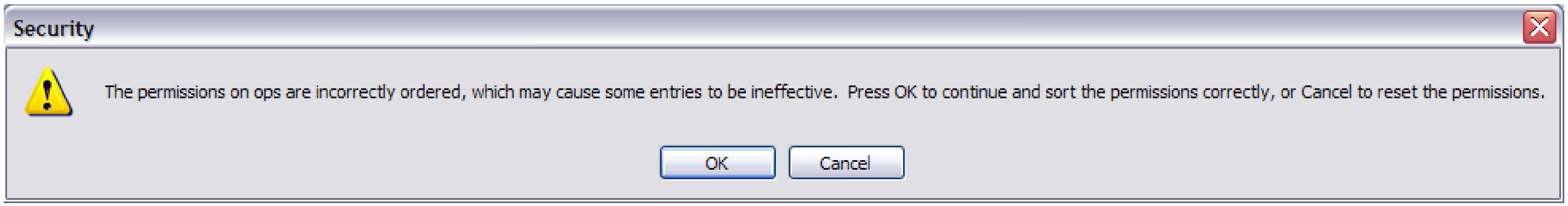
Mapping posix mode bits to NFS4 ACLs is not trivial

Posix group means everyone in that group except the owner

Other means everyone except the group or owner

Implemented using deny ACLs interleaved with allow ACLs

Windows complains about permission order

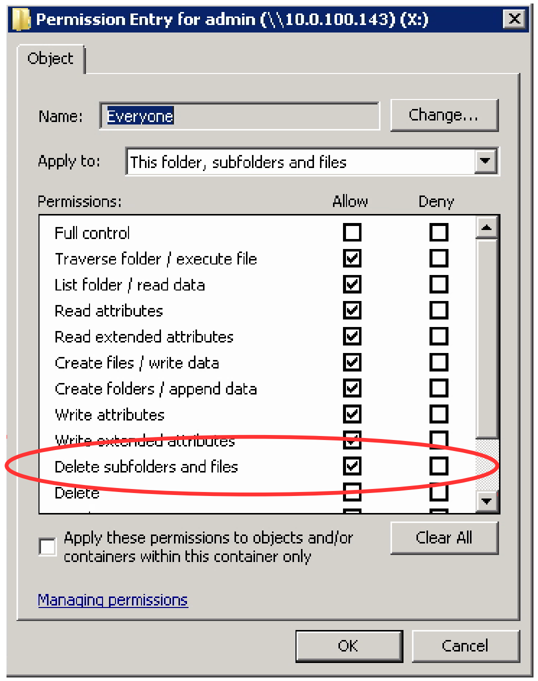

Security

The permissions on ops are incorrectly ordered, which may cause some entries to be ineffective. Press OK to continue and sort the permissions correctly, or Cancel to reset the permissions.

Example for wrong order of ACEs in ACLs

```
[root@st001.mgmt001st001 abc]# chmod 247 deny_order/
[root@st001.mgmt001st001 abc]# mmgetacl deny_order/
#NFSv4 ACL
#owner:root
#group:root
special:owner@:r-x-:deny
 (X)READ/LIST (-)WRITE/CREATE (-)MKDIR (-)SYNCHRONIZE (-)READ_ACL  (-)READ_ATTR
 (-)DELETE    (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR

special:owner@:-w-c:allow
 (-)READ/LIST (X)WRITE/CREATE (X)MKDIR (X)SYNCHRONIZE (X)READ_ACL  (X)READ_ATTR
 (-)DELETE    (X)DELETE_CHILD (X)CHOWN (-)EXEC/SEARCH (X)WRITE_ACL (X)WRITE_ATTR

special:group@:r---:allow
 (X)READ/LIST (-)WRITE/CREATE (-)MKDIR (X)SYNCHRONIZE (X)READ_ACL  (X)READ_ATTR
 (-)DELETE    (-)DELETE_CHILD (-)CHOWN (-)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR

special:group@:-wxc:deny
 (-)READ/LIST (X)WRITE/CREATE (X)MKDIR (-)SYNCHRONIZE (-)READ_ACL  (-)READ_ATTR
 (-)DELETE    (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (X)WRITE_ATTR

special:everyone@:rwx-:allow
 (X)READ/LIST (X)WRITE/CREATE (X)MKDIR (X)SYNCHRONIZE (X)READ_ACL  (X)READ_ATTR
 (-)DELETE    (X)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR
```

Add delete child to file if all other permissions are set


Permission Entry for admin (\\10.0.100.143) (X:)

Object

Name: Everyone

Apply to: This folder, subfolders and files

Permissions:	Allow	Deny
Full control	☐	☐
Traverse folder / execute file	☑	☐
List folder / read data	☑	☐
Read attributes	☑	☐
Read extended attributes	☑	☐
Create files / write data	☑	☐
Create folders / append data	☑	☐
Write attributes	☑	☐
Write extended attributes	☑	☐
Delete subfolders and files	☑	☐
Delete	☐	☐

☐ Apply these permissions to objects and/or containers within this container only

- **Some NFS4 filesystems don't allow to set the delete child permission on files.**
- **Without all bits set Windows will not mark the ACL entry as having full control.**
- **Need to set this ACL entry if everything else is set.**
- **The option "acl map full control" will enable this.**
- **Also interesting in case chmod 777 doesn't set this bit when the filesystem could store it.**

Allow folder owner to edit ACLs

In Windows the file owner is always able to set new ACLs.

Even without explicit permission Samba allowed to set new ACLs on files.

It was not possible to do the same on folders.

That's because the posix system call to open of files was skip if the access mask didn't demand any read or write data access.

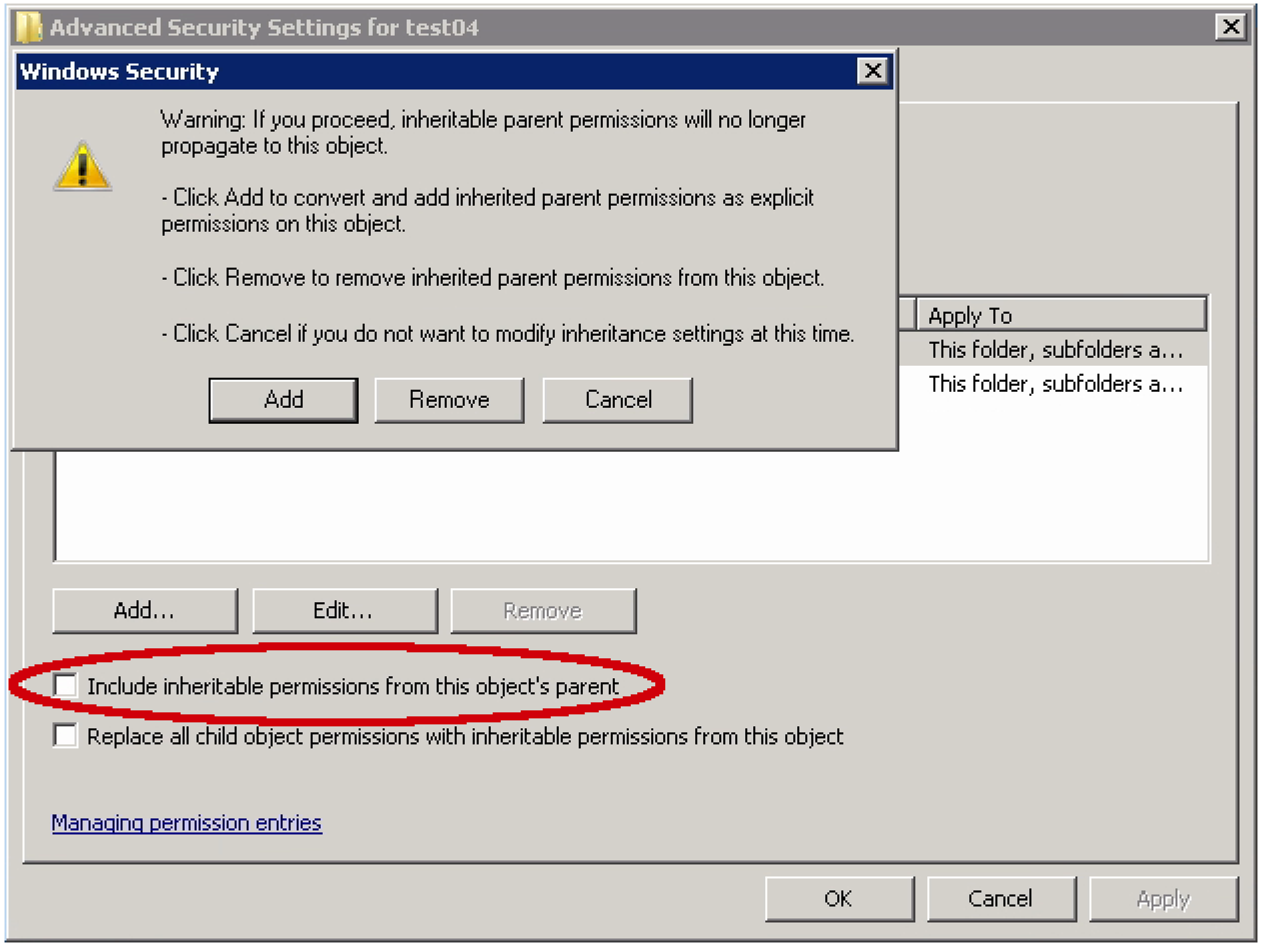
Similar code has been added to the directory open.

Allows folder owners to add new ACLs if someone deleted all ACLs of a folder.

Users can't lock themself out of their own files anymore. Less work for the administrators.

Should also help the xattr_acl module.

Allow folder owner to edit ACLs


Advanced Security Settings for test04

Windows Security

Warning: If you proceed, inheritable parent permissions will no longer propagate to this object.

- Click Add to convert and add inherited parent permissions as explicit permissions on this object.

- Click Remove to remove inherited parent permissions from this object.

- Click Cancel if you do not want to modify inheritance settings at this time.

☐ Include inheritable permissions from this object's parent

☐ Replace all child object permissions with inheritable permissions from this object

- **This made the issue of the DACL protected bit more serious: Since add didn't work a user might try and select remove next. And locked himself out due to this limitation.**

Older and other relevant changes

- **Support for CREATOR OWNER ACL entries with NFSv4 ACLs.**
- **Documentation updates**
- **Fix smbacls tool not to drop ACLs on user or group change**
- **New smbxcacls tool to directly process xattr acls.**
- **Use ID_TYPE_BOTH in ID mapping**

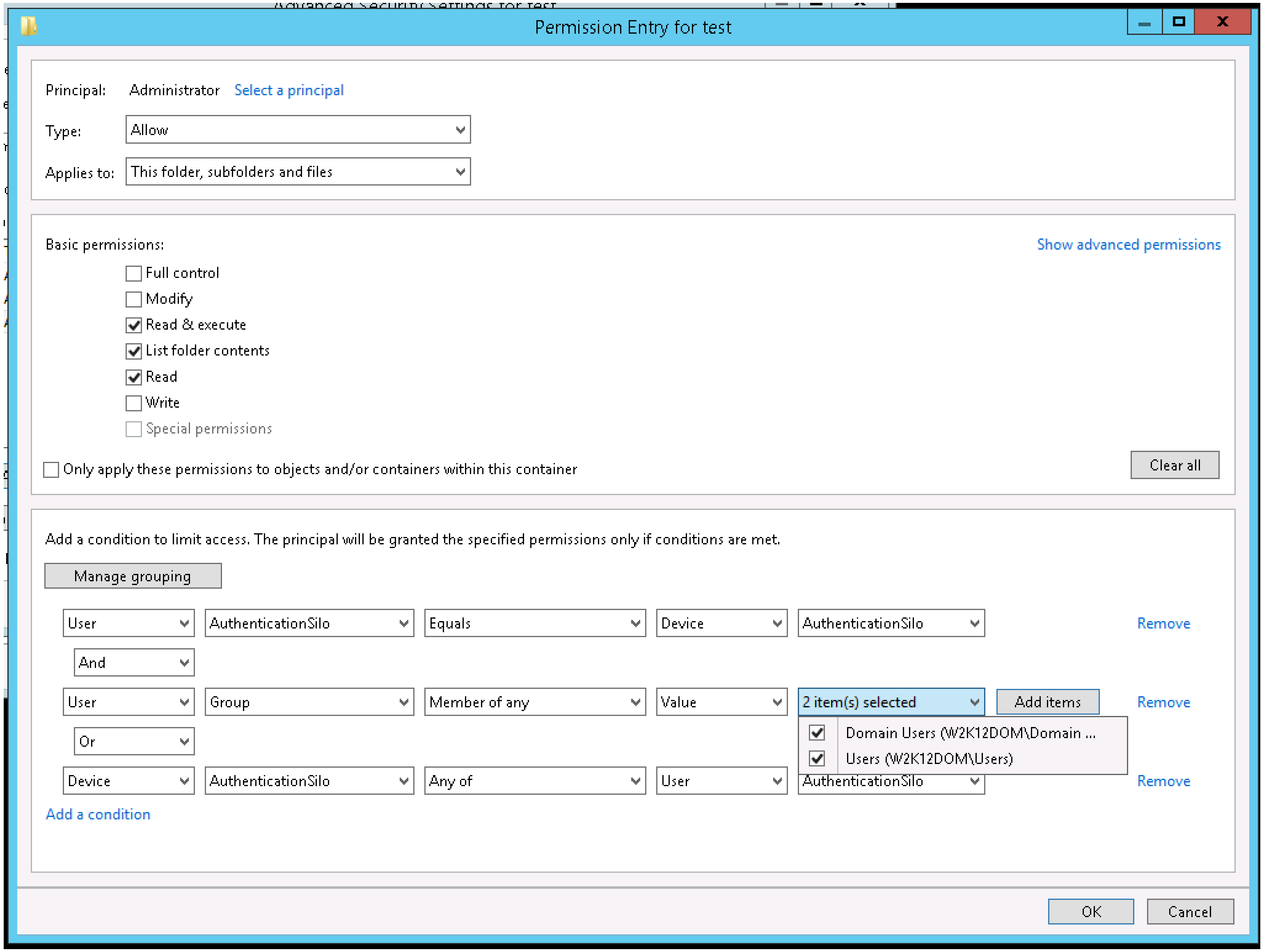
Remaining issues

- **Read/Write attribute**
- **System ACLs (audit and alarm ACLs, performance)**
- **Execute permission**
- **Read attribute required even with BTC enabled.**
- **Use xattr module and NFS4 modules**
- **Dynamic ACLs**

Stacked xattr module

- **Slower due to additional calls**
- **Support for appliances using the SD as data store**
- **No improvement in permission compatibility**

Dynamic ACLs



Permission Entry for test

Principal: Administrator Select a principal

Type: Allow

Applies to: This folder, subfolders and files

Basic permissions: Show advanced permissions

- [ ] Full control
- [ ] Modify
- [x] Read & execute
- [x] List folder contents
- [x] Read
- [ ] Write
- [ ] Special permissions

[ ] Only apply these permissions to objects and/or containers within this container

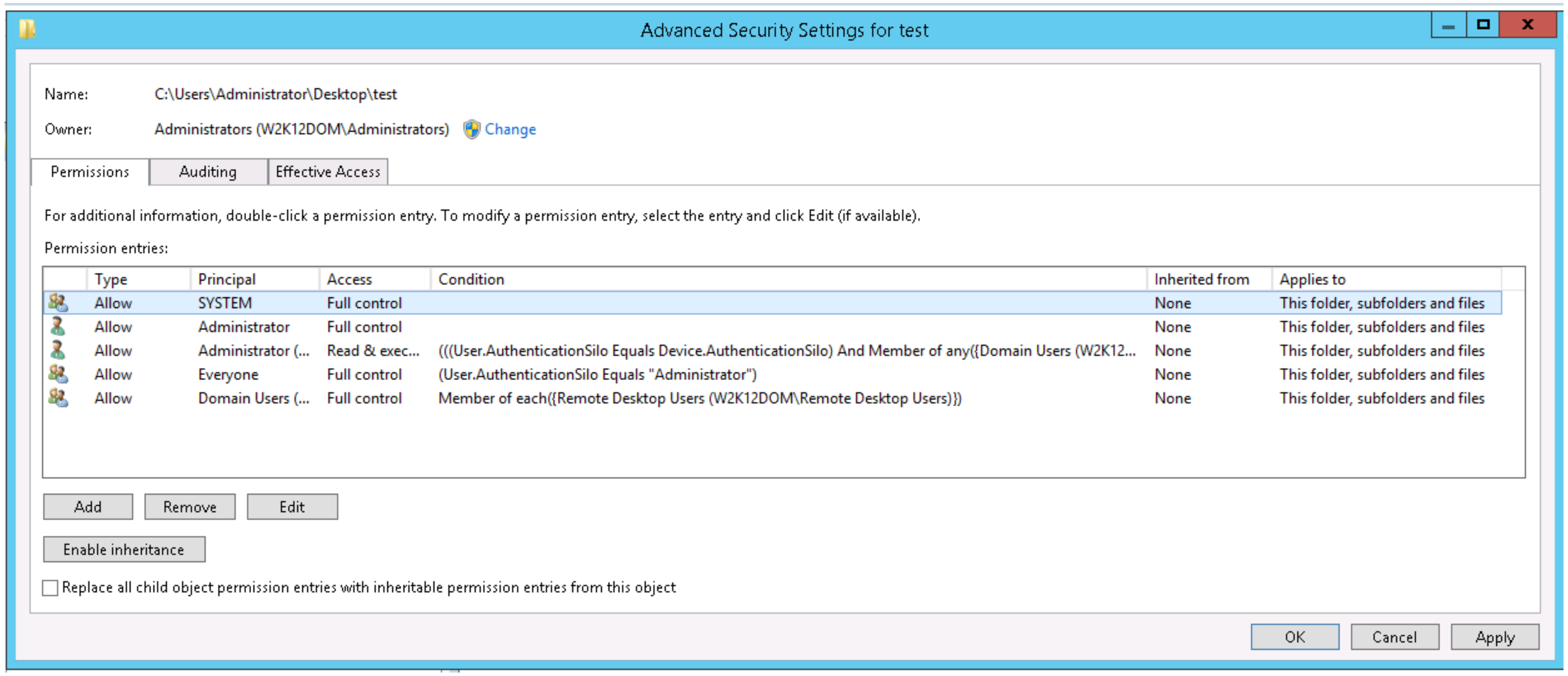
Add a condition to limit access. The principal will be granted the specified permissions only if conditions are met.

User	AuthenticationSilo	Equals	Device	AuthenticationSilo	Remove
And					
User	Group	Member of any	Value	2 item(s) selected	Remove
Or					
Device	AuthenticationSilo	Any of	User	AuthenticationSilo	Remove

- [x] Domain Users (W2K12DOM\Domain ...
- [x] Users (W2K12DOM\Users)

Dynamic ACLs

- **New enhancement to ACL binary format that can't be mapped to NFSv4**
- **New incentive to support that at least partially in Samba and use xattr to store it**
- **NFSv4 modules could drop or ignore unsupported ACL entries**
- **No additional access when the ignored entries are allow rules**



Name: C:\Users\Administrator\Desktop\test

Owner: Administrators (W2K12DOM\Administrators) Change

For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

Permission entries:

Type	Principal	Access	Condition	Inherited from	Applies to
Allow	SYSTEM	Full control		None	This folder, subfolders and files
Allow	Administrator	Full control		None	This folder, subfolders and files
Allow	Administrator (...	Read & exec...	(((User.AuthenticationSilo Equals Device.AuthenticationSilo) And Member of any({Domain Users (W2K12...	None	This folder, subfolders and files
Allow	Everyone	Full control	(User.AuthenticationSilo Equals "Administrator")	None	This folder, subfolders and files
Allow	Domain Users (...	Full control	Member of each({Remote Desktop Users (W2K12DOM\Remote Desktop Users)})	None	This folder, subfolders and files

☐ Replace all child object permission entries with inheritable permission entries from this object

Thanks for listening

- **Questions?**

3
1