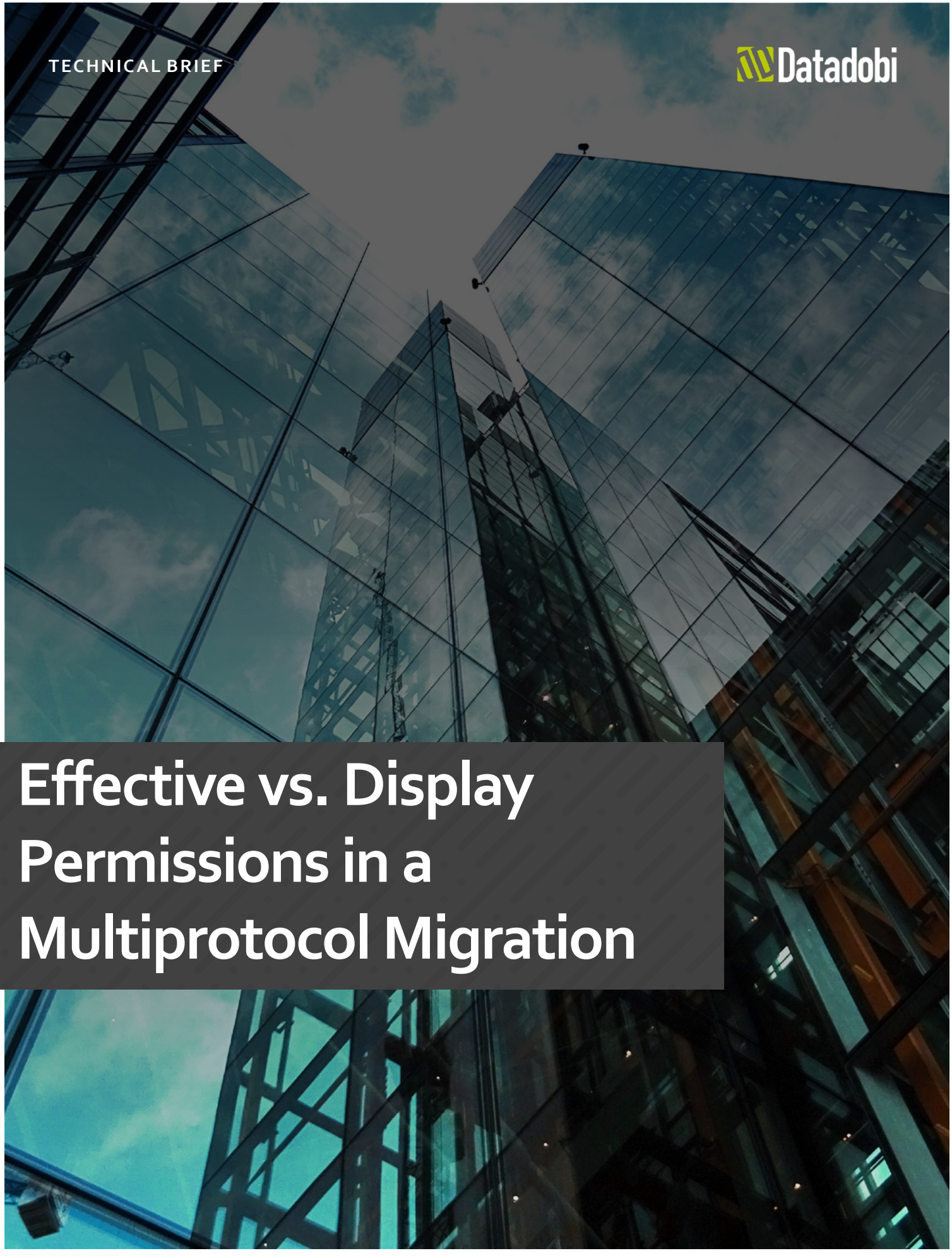

TECHNICAL BRIEF
Datadobi
Effective vs. Display Permissions in a Multiprotocol Migration

Copyright © Datadobi, All rights reserved.

Datadobi believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS." DATADOBI MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any Datadobi software described in this publication requires an applicable software license.

All other trademarks used herein are the property of their respective owners.

CROSS-PLATFORM MULTIPROTOCOL MIGRATIONS

A typical NAS migration involves copying file-based content either over the SMB protocol or the NFS protocol. In each case the security of the filesystem objects (i.e., files, directories, symlinks, etc.) is governed by the security model associated with the access protocol – e.g., NTFS security for objects accessed over SMB and POSIX mode bits for objects accessed over NFSv3. In some cases, however, the NAS device has the capability to allow simultaneous access to common filesystem objects using either access protocol simultaneously – the industry refers to this ability as "multiprotocol" access. A migration involving data provisioned via the multiprotocol capabilities of the NAS device presents unique challenges when the migration involves a source and destination platform that are different. At Datadobi, we refer to this type of migration where the source and destination platform have different architectures as a cross-platform migration. The reason multiprotocol migrations are difficult in a cross-platform scenario is because there are no industry standards governing the underlying implementation of multiprotocol security. There is often confusion regarding this situation because while SMB is a standard and NFSv3 is a standard, the simultaneous multiprotocol access of filesystem objects using both protocols is governed by no standard whatsoever. Each vendor providing multiprotocol capability has created a proprietary method of maintaining security as it relates to the storage of permissions and the resulting evaluation and enforcement of access control.

As can be seen in Table 1, the way in which different NAS platforms store permissions and grant client access is unique to each platform. NetApp, for example, stores either NTFS or UNIX permissions but not both. Dell EMC's VNX, Unity, and PowerStore platforms running in "Native" access mode will store both NTFS and UNIX permissions separately while Dell EMC's Isilon/PowerScale implements a "Unified" permission model wherein both sets of permissions are combined into a single permission model. Each platform also has its own method of evaluating and enforcing access control depending on the type of permissions stored as well as the type of client protocol in use.

	NTAP MIXED	ISILON/PowerStore	NATIVE VNX, Unity, and PowerStore
User management	User mapping required.	User mapping required.	User mapping required.
Permission storage	Single set of permissions stored (NTFS or UNIX). If NTFS permissions on a file are changed, the storage system recomputes UNIX permissions on that file. If UNIX permissions or ownership on a file are changed, the storage system deletes any NTFS permissions on that file.	Unified set of permissions stored (NTFS + UNIX combined).	Dual set of permissions stored (NTFS permissions and UNIX permissions stored separately using 'Native' access policy).
Client access	NTFS Security descriptor or UNIX mode bits stored based on last client to modify permissions. If the last client was UNIX then NFSv3 mode bits or NFSv4 ACLs are stored. If the last client was an SMB client, then NTFS ACLs are stored.	Access token compared to Unified permissions.	Either NTFS security or UNIX mode bits will be used to grant access depending on the type of client issuing request.

Table 1 - Comparison of Multiprotocol Security Implementations

In every NAS migration it is critical to maintain the permissions as the data is copied from the source platform to the destination platform. If the data is copied but the permissions are not accurately transferred, then the migration cannot be considered successful. In cross-platform multiprotocol migrations, the challenge of accurately copying permissions and maintaining behavior is complicated by the fact that each platform has its own method for storing permissions. When we say "maintain behavior" we mean that consistent behavior between the two platforms will be such that when a given user attempts to access a file or directory object on the destination platform, they will either be granted or denied access in exactly the same fashion as on the source platform. In other words, the filesystem object will neither be more, or less, accessible on the destination platform as it was on the source. Due to the differences in multiprotocol security implementations across platforms, it is often difficult to maintain identical behavior between the disparate platforms because of the way that permissions must be copied between the source and destination.

When executing a cross-platform multiprotocol migration, DobiMigrate will capture the security metadata associated with each protocol. In other words, for each filesystem object that is encountered during copy operations, DobiMigrate will not only copy the filesystem object data/content but also the metadata from the source to the destination platform. These two distinct set of copy operations are performed first by copying the data via the NFS protocol, next copying the SMB metadata, and finally copying the NFS metadata.

Since both NFS and SMB protocols are used during the migration, there are two different types of clients accessing the same filesystem objects. Simultaneous access to the same filesystem objects via different protocols leads to potential discrepancies in how access permissions appear due to differences between "Effective permissions" and "Display permissions". As an example, we'll review migration using a NetApp filer as a source and an EMC Isilon as the destination.

EFFECTIVE AND DISPLAY PERMISSIONS DEFINED

The difference between "Effective permissions" and "Display permissions" within a multiprotocol environment such as a NetApp filer can affect the behavior of the dataset on the destination platform. This is because only one set of permissions are associated with a given file/directory object (see Table 1).

- "Effective permissions" are the actual permissions associated with the file or directory. They can be either NTFS ACLs or UNIX style POSIX mode bits.
- "Display permissions" are the permissions shown when querying the permissions of a file or directory using for example "ls -l" on UNIX or "Properties –> Security" on Windows.

NetApp systems support multiple security styles that control the type of **"Effective permissions"**:

- NTFS: All effective permissions on the files and directories are "NTFS security style" (ACLs).
- UNIX: All effective permissions are UNIX security style (POSIX mode bits for NFSv3).
- Mixed: Each file or directory has either NTFS-style effective permissions or UNIX-style effective permissions, but not both (see Table 1).

The security style has impact in two major areas: 1) Access control and 2) Displaying permissions.

1) ACCESS CONTROL

"Access control" in this context refers to the following:

- A user connects to a volume or qtree using a certain protocol (SMB or NFS).
- The user asks for certain permissions on a file or directory (read, write, etc.)
- The NetApp filer determines whether to grant or to deny the requested permissions.

The rules for determining access are as follows:

- When the security style matches the protocol (NTFS and SMB or UNIX and NFS) the effective permissions are used for access control.
- When the security style doesn't match the protocol, the credentials supplied by the protocol are mapped to credentials that match the security style. For example, a UNIX user requesting access on a file with NTFS permissions is mapped (via user mapping) to an equivalent NTFS user. The mapped NTFS user (SID/Group SID) is used to check for access using the effective permissions.

2) DISPLAYING PERMISSIONS

"Displaying permissions" equates to the following:

- A user or client connects to a volume or qtree using a certain protocol (either SMB or NFS).
- This user displays the permissions of a file or directory using 'ls -l' in UNIX or in the 'Properties –> Security' dialog on Windows.

For each bullet point listed above, the "Display permissions" are used. These are not necessarily the same as the "Effective permissions."

THE IMPACT OF EFFECTIVE VS. DISPLAY PERMISSIONS

It is important to understand that "Display permissions" are not used to control access to a given filesystem object. Access control is enforced by a combination of user mapping and comparison of the mapped user credentials to the "Effective permissions."

Two very important considerations:

1. "Display permissions" are NOT used to control access to the file or directory. Access control is done with a combination of user mapping and the "Effective permissions."
2. "Display permissions" are almost always MORE PERMISSIVE than "Effective permissions."

Earlier we discussed the various NetApp security styles (NTFS, UNIX, and Mixed); when the "Mixed" security style is used, the filer will store either NTFS permissions or UNIX permissions but not both. With "Mixed" security, a given folder or qtree can contain files with "Effective" NTFS permissions but also contain files with "Effective" UNIX permissions.

When migrating to NetApp you should select the protocol that matches the security style of the Qtree/Directory, and setup NetApp's user mapper to handle the other protocol.

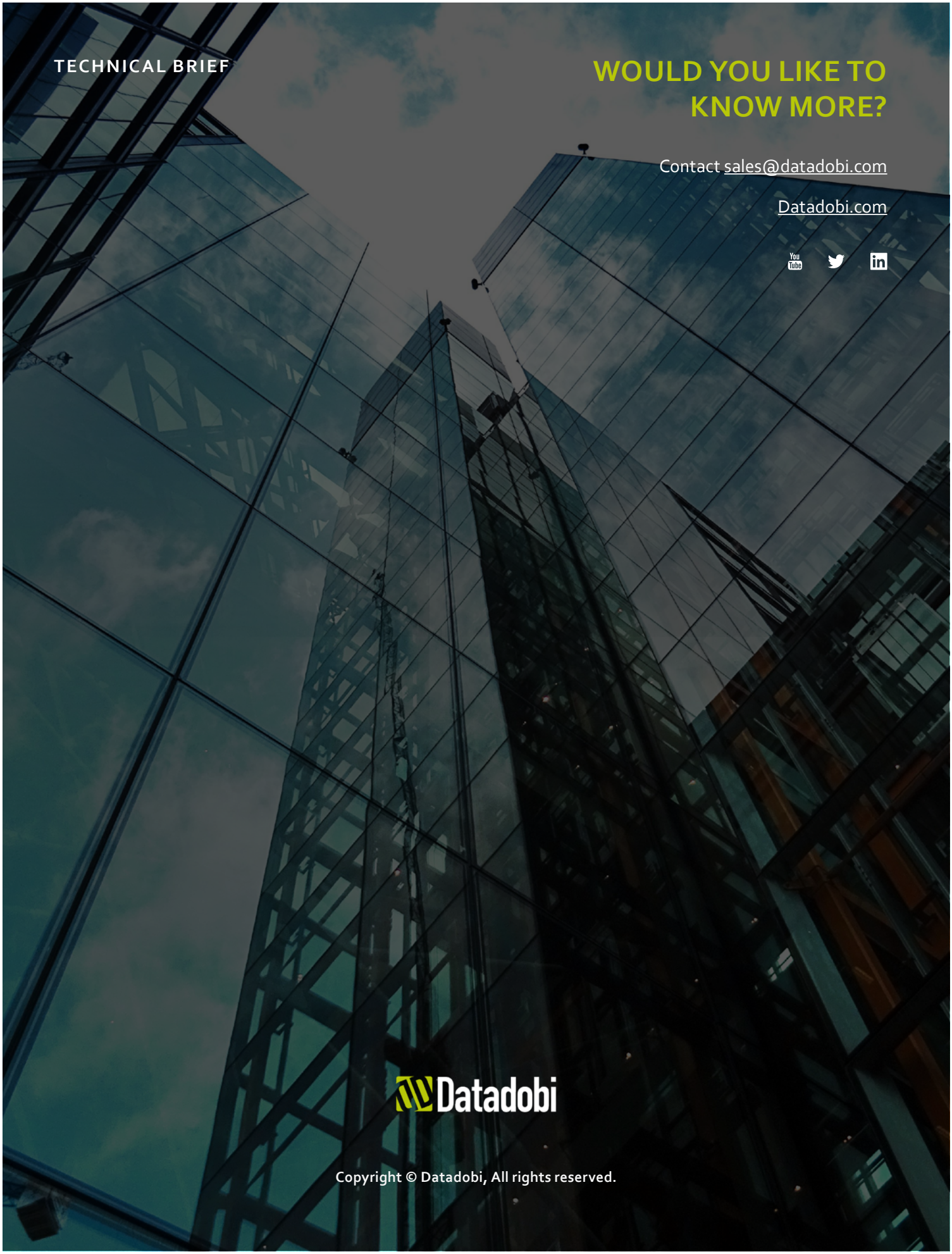
Behavior on the Isilon/PowerScale after a multiprotocol migration:

- Client access over SMB, the Isilon/PowerScale will use the "Effective" NTFS permissions.
- Client access over NFS, the Isilon/PowerScale will use the "Display" NFS permissions, which will likely grant more access than the "Effective" NTFS permissions had user mapping been performed.

As can be seen, the post-migration behavior between the two platforms can be different. Given this possibility, it is important to build time into any work plan for a cross-platform multiprotocol migration to test behavior and determine a remediation plan for instances where behavior between two different implementations proves to be different. In some cases, it is a valid question to raise as to whether the content provisioned via multiprotocol capabilities truly needs to be provisioned as such. In some situations, administrators have provisioned filesystems over multiprotocol when such access either a) isn't truly used or b) an assumption was made that at some point multiple types of clients would need to access the same filesystem data.

SUMMARY

Cross-platform NAS migrations can be difficult and cross-platform multiprotocol migrations only add to the level of difficulty due to the lack of any industry standards. DobiMigrate NAS migration software simplifies these migrations by providing the automation, error checking, and validation that is required to quickly execute these migrations. DobiMigrate's multiprotocol migration capabilities automate the difficult tasks usually encountered when using multiple scripting tools to execute these types of migrations. Even with DobiMigrate's automation capabilities, it is still important to understand how permissions will migrate and how behavior can change on different platforms given the lack of multiprotocol standards. Understanding how "Effective vs. Display" permissions factor into the implementation details between the source and destination platform will make for a more predictable and successful multiprotocol NAS migration.



TECHNICAL BRIEF

WOULD YOU LIKE TO KNOW MORE?

Contact sales@datadobi.com

Datadobi.com

Datadobi

Copyright © Datadobi, All rights reserved.