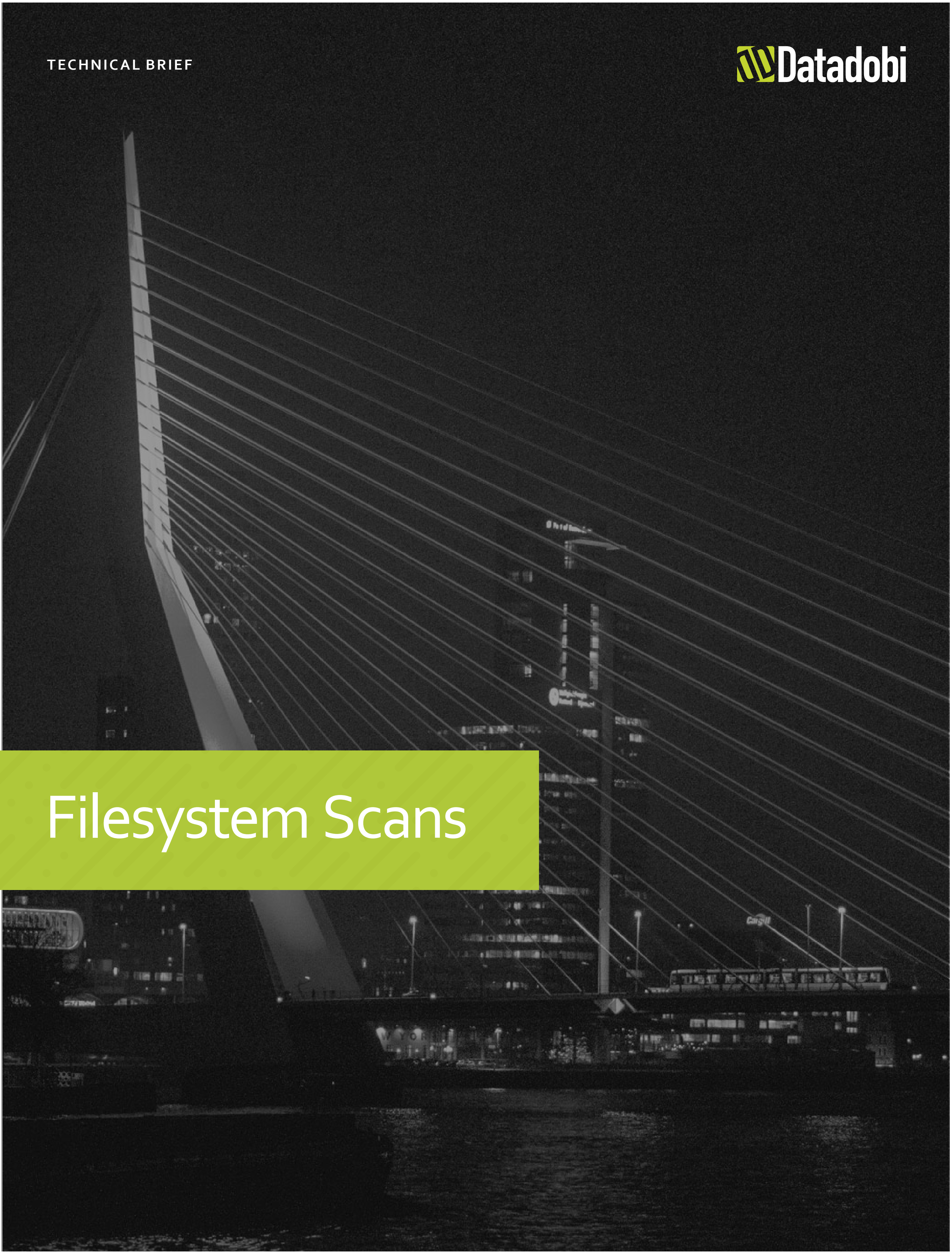
TECHNICAL BRIEF

Datadobi



Filesystem Scans

TABLE OF CONTENTS

INTRODUCTION

Datadobi software uses patented technology to synchronize file and object data between two systems or clouds. These systems can contain billions of files or objects, spread over various directories, filesystems, or buckets.

A key aspect of this synchronization is determining what data needs to be copied, updated, or deleted. This document describes how Datadobi software identifies what deltas exist between a source system and a destination system and how it executes this comparison in a fast, reliable, and scalable way.

We will show why “scanning” the filesystems or object storage buckets is the most reliable way to detect changes, and why other tools are simply too limited when compared to the modern capabilities of the Datadobi software suite.

Note that we are not discussing the methods by which array-based data synchronization is accomplished. Instead, we address the challenges posed when dealing with data copies across network-attached storage (NAS) platforms or object stores belonging to different product lines or from different vendors. When synchronizing data across heterogeneous platforms or clouds, array-based migration or replication cannot be used.

DATA COPY TIMELINE

A cross-platform data migration, protection, or other copy project can be broken down into a number of phases, starting with the seeding of the content from the source system into the destination. We refer to this seeding of data as the “first copy” operation.

Once the first copy is complete, the data copy engine settles into the “steady-state” phase. This is an iterative phase, whereby automated incremental copy operations are executed in order to keep the source and destination systems synchronized.

When performing data migrations, the speed of incremental copy operations is crucial since it will determine the length of your read-only maintenance window during the final cutover from source to destination. See **Figure 1** for the timeline of a typical enterprise data migration.

For data protection, the speed and scheduling of the incremental copies will directly impact your recovery point objective (RPO). It will also impact how long a planned failover takes.

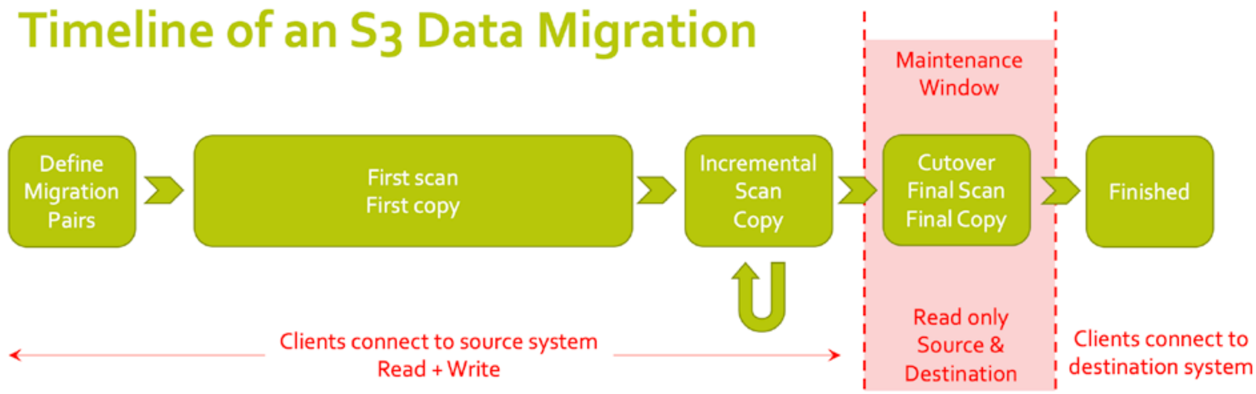


Figure 1 – Data copy timeline

The overall process of executing one single incremental copy consists of the following steps as depicted in **Figure 2**:

1. A scan happens on both the source and destination systems to capture the current state of all items in the filesystems or object stores. There can be billions of such items. For complex, multiprotocol NAS setups, it is necessary to scan the data using both the SMB and NFS protocols.
2. The difference (delta) between the source and destination scans is calculated. Together with the chain-of-custody journal that was created during previous iterations, the Datadobi software determines which operations are required to further synchronize the two systems.
3. Delta copy, update, and delete operations are then executed to synchronize the two systems. New files that have been added, files that have changed, and metadata updates, such as permission changes, file deletions, object metadata changes, etc., are all propagated to the destination system.

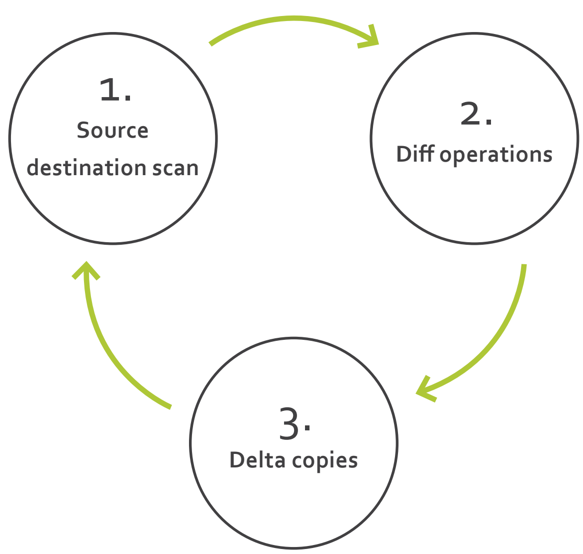


Figure 2 – Steady state/incremental copy phases

OTHER TOOL SCANS VERSUS DATADOBI

Although scanning is clearly the most effective method in determining what file, directory, or object changes have occurred, not all scans are created equal. Some are slow and single threaded, and some can be multithreaded but with no intelligence in how those threads are distributed across the filesystem or object store structure. Scans using other tools result in unacceptably slow performance, given today's ever-increasing number of items on shared NAS devices or object stores.

Unfortunately, the word "scan" has certain historical connotations that should not be applied blindly to new products. A modern scan implementation, as found in Datadobi software, is highly efficient compared to the unsophisticated scans executed by legacy tools. In order to understand the different approaches, let's compare with legacy tools, such as Robocopy and rsync.

Rsync has existed since 1996, when a 9 GB hard drive was considered a large-capacity storage device. It was originally developed using a single-thread model, since most servers and workstations at the time were single CPU, and it was mainly used for backing up Unix workstations or very small Unix web servers. Fast forward to today, and rsync has added many features, but it is still a single-threaded application. It does not take advantage of today's multicore processors and large memory footprints. Scan performance is therefore extremely slow.

While multiple rsync instances are run in parallel, complicated shell scripts must be written to parse the filesystem structure and assign each portion to a unique instance of rsync. This is a brute-force approach that does not scale well and limits performance. Such an approach does not leverage any intelligence regarding the structure, breadth, and/or depth of the filesystem. Valid data could potentially remain uncopied since scripts will not detect new directories and/or volumes added to the source during the migration.

The tree structure of a filesystem will vary considerably, with some leaves of the tree being narrow and/or shallow, while others are extremely wide or deep. In a parallel rsync model, all leaves of the tree have the same weight and will be scanned serially by a single thread. In this model, thread usage is extremely uneven and inefficient.

Finally, large migrations can take multiple days or weeks. Every migration will encounter errors. There can be network errors, access or permission errors, transient errors in the storage system, temporary unavailability, or other errors that are expensive to root cause. It is important that the migration software can handle a large volume of errors, regularly retrying the operations that caused the error, and finally allowing the storage administrator to process groups of these errors (e.g., adjusting the permission on a subdir). Processing a sizeable set of migration errors requires a dedicated user interface and an admin-friendly operational flow.

In **Figure 3**, the rsync single-threaded model is not highly efficient in the context of a large filesystem with multiple levels of directories. The scan thread has to work its way through the directory tree in a serial fashion. Rsync is also limited to use with filesystems accessed over NFS, so it only covers a portion of modern NAS devices capable of servicing both NFS and SMB clients.

Rsync Scan Thread Usage

Fixed scan thread count = 1

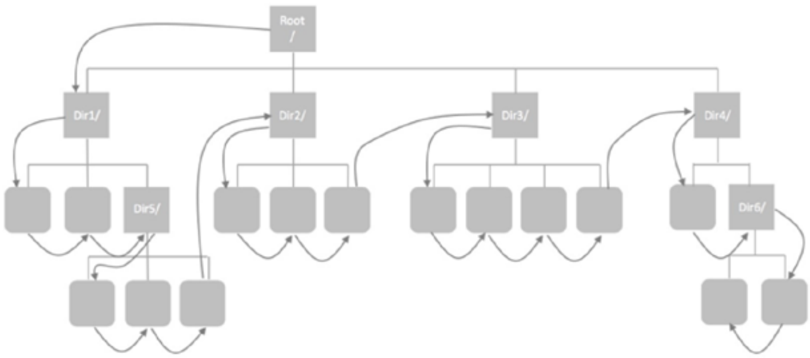


Figure 3 – rsync scan thread execution

Robocopy was introduced with Windows NT 4.0 in 1997 – again, when storage capacity was a fraction of what it is today. Like rsync, Robocopy's development has continued slowly over the years. While it has evolved to include the ability to run multiple copy threads (8 by default), only a single scan thread is utilized to update filesystem maps.

In **Figure 4** below, Robocopy launches a single thread that must serially traverse all directory structures across all filesystems. Robocopy is also limited to scanning NTFS filesystems, so, like rsync, it can only scan a portion of modern NAS devices capable of providing storage access to both SMB and NFS clients.

Robocopy Scan Thread Usage
Fixed scan thread count = 1

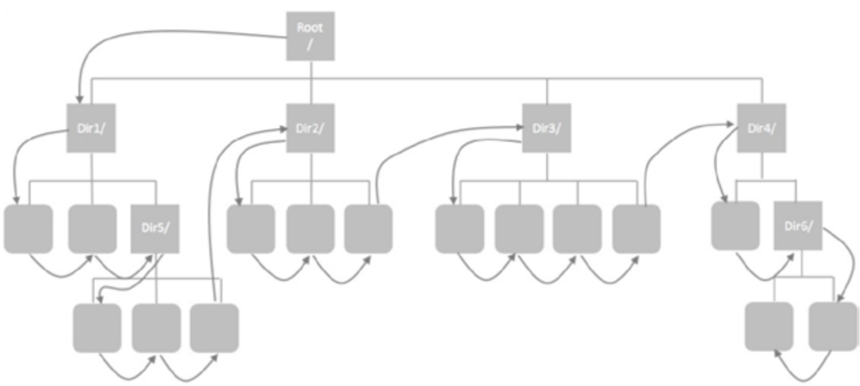


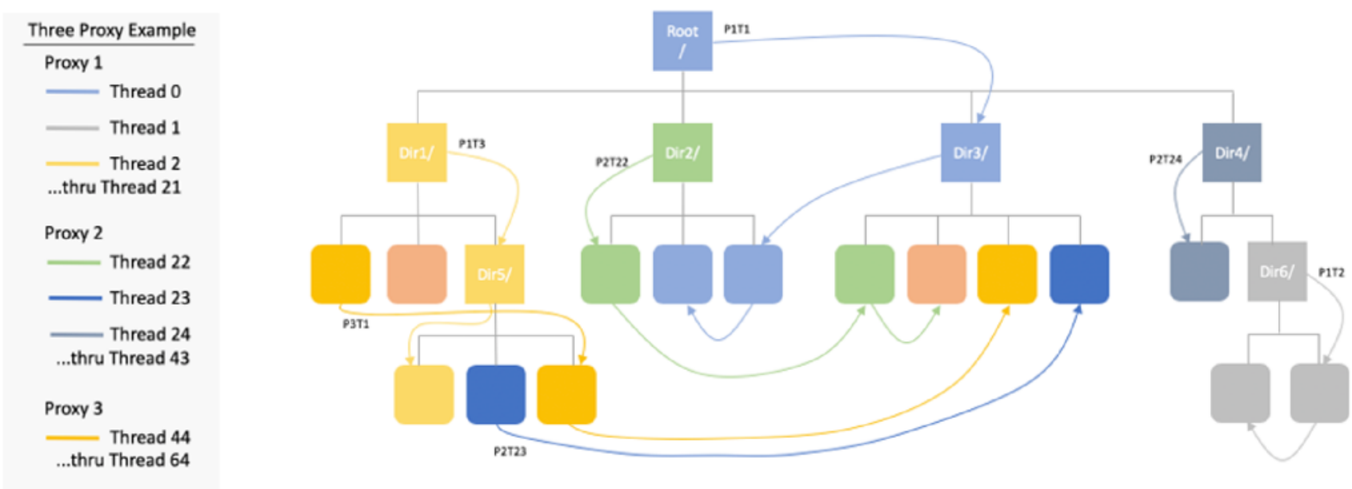
Figure 4 – Robocopy scan thread execution

Turning to the Datadobi software suite, scans are significantly faster than legacy tools due to its ability to not only launch a larger number of threads of execution, but to do so more intelligently.

Conceptually, we create a queue and then execute a number of individual worker threads, with each one pointing to various directories. **Figure 5** shows a simplistic illustration of how the Datadobi software suite scans the filesystem objects contained within each directory and does this in parallel. Each thread is processing portions of the overall filesystem simultaneously. Additionally, since the Datadobi's software stack includes both SMB and NFS proxies, both NTFS and Linux filesystems can be scanned in parallel – rsync and Robocopy are limited to either Linux or NTFS filesystems, respectively, and scan the filesystem in a serial fashion.

DobiMigrate®/DobiReplicate® Scan Thread Usage

Default scan thread count = 64 (threads distributed across running proxies)



Distributed threads process directory structures in parallel.

Figure 5 – Datadobi thread execution

Finally, for object scanning, a different approach has to be taken. Object storage does not have the concept of nested directories; instead, a flat namespace is available. Scanning this namespace single threaded from beginning to end would take a very long time. Therefore, Datadobi software deploys a divide-and-conquer approach to split the object namespace and parallelize the scanning. The algorithms are optimized to deal with constraints of the object storage systems and Unicode character encodings, in order to maximize the scanning speed.

SUMMARY

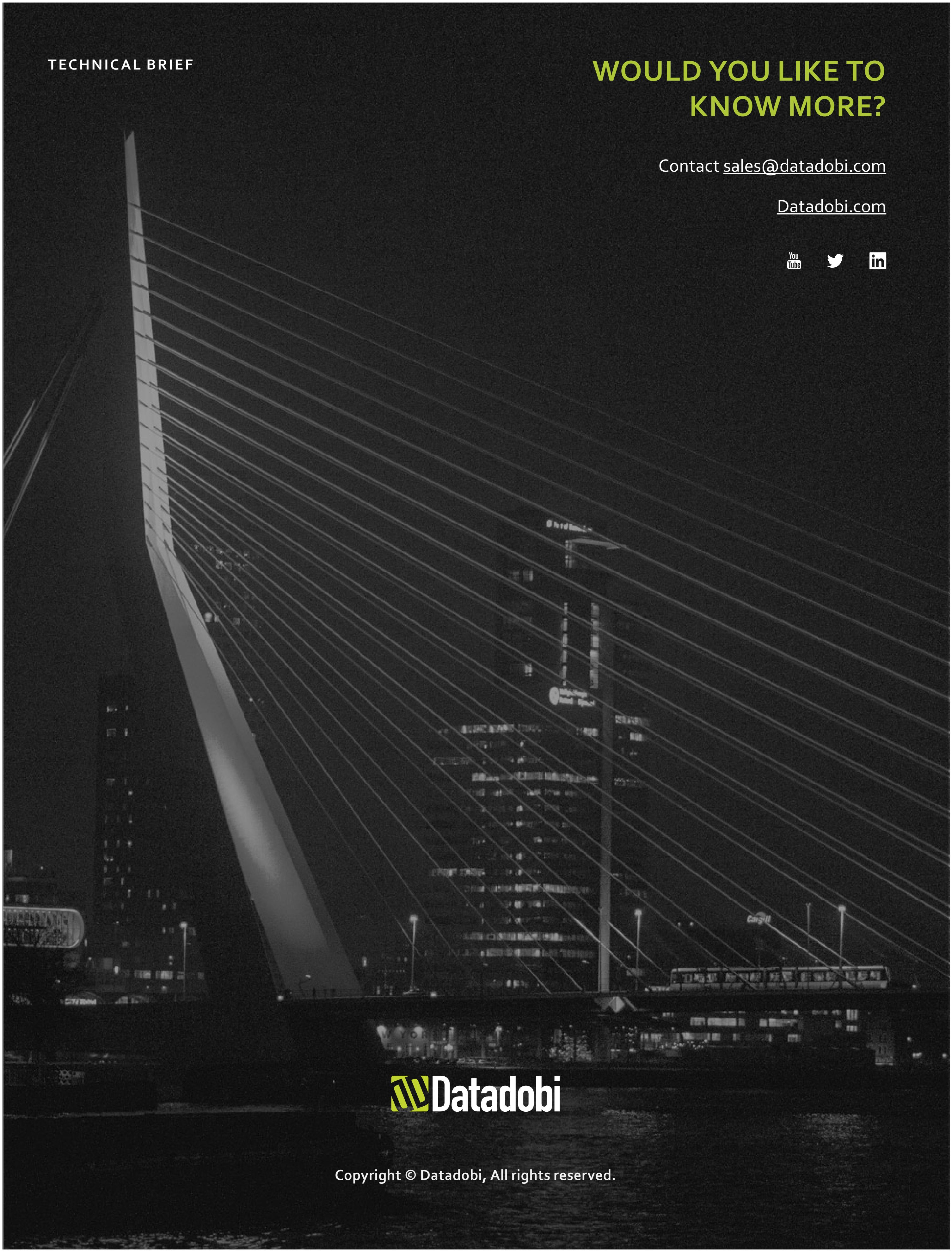
Cross-platform file or object synchronization requires highly innovative software to produce very fast scan times. The Datadobi software suite is significantly faster than other tools and works with any NAS or object platform (including standalone servers). It is extremely accurate, with no dependency on various platform APIs to report and distribute filesystem change notifications.

Additionally, SMB, NFS, and object platforms are handled with a single tool that requires no scripting and offers a scalable workflow for the administrator managing the migration or protection. Results provided by highly efficient parallel scans between platforms guarantee delta operations are quickly generated to maintain synchronization between the source and destination.

WOULD YOU LIKE TO KNOW MORE?

Contact sales@datadobi.com

Datadobi.com



Datadobi