Understanding Storage Intermediaries

Helen Hockx-Yu
Office of Information Technologies
University of Notre Dame
USA
hyu3@nd.edu

Don Brower
Hesburgh Libraries
University of Notre Dame
USA
dbrower@nd.edu

Abstract – Storage intermediaries are software, and sometimes hardware appliances that act as a link between applications and storage media, performing a range of tasks, such as protocol translation, caching, compression or even encryption. This paper describes storage intermediaries and their key functions that librarians and archivists should be aware of, as these introduce technical dependencies that can impact digital preservation.

Keywords – archival storage, storage gateway, storage caching layer, Cloud storage, tape storage

Conference Topics – Scanning the New Development; Building the Capacity & Capability;

I. INTRODUCTION

Storage intermediaries are a class of software, often in combination with hardware, that sit between applications and data on storage media, performing a variety of functions. There are many types of these intermediaries. Some translate one kind of file system to another. Others add features that optimize data transfer and storage. For example, they can translate storage protocols into widely used network file systems such as Network File System (NFS) or Server Message Block (SMB) to work with Cloud-base object storage. Or they could cache a subset of frequently used data, so that future requests for it can be served faste from this transient storage layer, rather than from the slower underlying primary storage location. Data compression and deduplication are also common functionalities. Some storage intermediaries can even encrypt data before writing it to storage media and manage the encryption keys.

Two common examples of storage intermediaries are Cloud storage gateway such as the AWS Storage Gateway File Gateway, and tape storage gateway such as Spectra Logic's BlackPearl Converged Storage System.

The AWS Storage Gateway File Gateway connects on-premises applications to AWS' Cloud storage such as S3 through a virtual machine or a purpose-built appliance, and caches data locally for low-latency access. Its protol conversion or translation allows files stored as S3 objects to be accessible via standard protocols such as NFS and SMB, appearing as mapped network drives on end users' machines [3].

Spectra Logic's BlackPearl Converged Storage System [10] connects with tape storage and also allows data to be replicated to Azure, Amazon S3 and other Clouds. BlackPearl uses on-premises appliances for caching and keeping a system database that contains a list of all objects it stores in various media/locations. Again it does protocol conversion, between NFS/SMB to LTFS (Linear Tape File System) and S3, so that applications can seamlessly consume what is stored on tapes and in the Cloud.

While the extent varies, storage intermediaries introduce a range of technical dependencies that we rely on to restore data from the storage media and present it in a file system for the consumption of applications. Compression and deduplication for example, are both reversible processes but the details of the algorithms involved are often opaque. The reversion cannot be

17th International Conference on Digital Preservation
iPRES 2021, Beijing,China.

DOI: 10.1145/nnnnnnn.nnnnnnn

performed independently without the intermediaries that compressed or broke up the files initially. Dependencies introduced by storage intermediaries challenge some of our best practices, yet have not caught much attention by the digital preservation community. They need to be understood better so that the impact on digital preservation is not overlooked.

The remainder of the paper describes the key issues related to storage intermediaries that librarians and archivists need to be aware of, as well as recommendations for addressing these in the context of long term digital preservation.

II. Storage Intermediaries: Transforming files

Many things happen to files between the point of being sent off for storage, and being restored from storage for use. Storage intermediaries perform certain transformations to the files in order to get the job done. These include changes to the structure of the files and the value of associated metadata, and the processes may even introduce additional (hidden) files that are necessary to support communication over protocol barriers or rendering of files.

A. File Transformations

Storage intermediaries may not store files in the same way as they are presented. To improve the efficiency of transfer and storage, data may be stored compressed, and "deduplicated", where identical blocks of data are only stored once. File names may also be replaced with wildly different names, where storage intermediaries rely on other metadata to map the presented name to the stored name. Common approaches are to use a hash value or a UUID as the stored name. This technique is often used by intermediaries that provide version control and snapshots, since two versions of a file cannot be saved under the same name on the back end storage.

Files may also be split, or "chunked", into smaller segments (called "blobs"), so that they can be packaged up and streamed to the end storage target as efficiently as possible. This is not something presented to the user and files will always be rebuilt from the blobs when a restore is requested. Chunking or blobbing is often used for the purpose of deduplication, a technique for eliminating repeating data. It may also be used with intermediates that provide caching and tiering between local and Cloud content. Data is moved across different media types and placed on storage that is most suitable based on performance needs. Only the pieces being actively used are stored in the (local) cache. For example NetApp Cloud Tiering chunks files in 4k blobs and stores "stale" blobs in the Cloud, instead of complete, stale files. So you could have 20 blobs of making up a file stored on premises and the remaining 2000 stored in the Cloud.

B. "Phantom" files

Hidden files (or folders) are files (folders) that are not displayed by default by desktop file browsers. Common examples of these files are the metadata files created by file browsers, for example the ".DS_Store" files created by the MacOS Finder. Another example is "thumbs.db", which is generated automatically by Windows to store thumbnail images of files to speed up Windows Explorer's thumbnail view. These files are file browser specific, not meant to be opened manually, but referenced by operating systems.

Many Storage intermediaries support access to stored data as a network file system, such as NFS and SMB. This presents stored data as a network drive, accessible using native file browsers. While providing convenience, when write operations are permitted, hidden and invisible files could be added to the storage location inadvertently. These "phantom" files are insubstantial in relation to the material intended to be stored and they change whenever someone views a folder. They could accumulate if unmanaged, flood your collection and become a headache for data migrations.

C. Filesystem Metadata and in-file metadata

Filesystem metadata is external metadata associated with a file, such as file size, attributes, permissions, timestamps like creation date, last accessed date and modification date, and a lot more. This is not stored with the file payload but in a different place on the storage media. Embedded or in-file metadata is an integral part of a digital file construct (as defined by file format specification). It is stored with the content and set when the file is first created and generally does not change thereafter.

Each filesystem has its own standards for the metadata tracked on each file. Some adhere in

various degrees to the POSIX standard [7], others, such as object stores, track not much more than a name and a creation timestamp. When a storage intermediary presents an object store as a network filesystem, it will not change the metadata embedded in the files but will most likely alter the filesystem level metadata. The timestamps for example get updated every time a file or folder is copied, moved, written to, or otherwise worked with. Copying or restoring files can change the date created, and a complete storage intermediary migration can reset all the dates to when the migration occurred. It is important to know that file system metadata can change arbitrarily, is easily misinterpreted and therefore should not be used as a check for sameness or file integrity.

III. IMPACT ON PRESERVATION

The goal of digital preservation is to maintain ongoing access to digital objects over time. This involves overcoming hardware and software obsolescence and interpreting what was created and stored with older technologies, without losing the intellectual meaning of the digital objects.

Storage intermediaries perform very useful functions and can significantly reduce the size of data transferred and stored, delivering cost advantages. Techniques such as compression, deduplication and tiering however also transform files, and much of the process is opaque, based on proprietary technology and internal to the storage intermediaries. Files may be broken up and compressed, put away or cached, in parts or entirety, then reassembled. The mapping or restoration is supported by knowledge (metadata) kept by storage intermediaries which is typically not exposed. It is very much like going through a black box, all you can hope for is to get back the same thing you've put in. For this to happen however, the files will have to do the round trip through the black box. Direct access to data at the storage location absolutely does **NOT** mean you have got hold of the files you sent off. Figure 1 shows a list of stored objects in an S3 bucket, written by a storage intermediary product. Without going through the same storage gateway and being reassembled and restored back to files, these encrypted binary blobs become meaningless and are simply not usable as-is.

Figure 1. Examples of "transformed" objects stored in an AWS S3 bucket.

Perhaps the most important implication for digital preservation is the deep dependencies introduced by many storage intermediary products, and their crucial presence for rendering and interacting with bits sequences stored on physical media. Much of the work in digital preservation has focused on the risks of format and media obsolescences, or hardware and software obsolescence in general, without calling out storage intermediaries explicitly. There is no mention of this specific class of hardware and software in Digital Preservation Storage Criteria [12], a list of design attributes for storage that supports the work of digital preservation, while the Usage Guide advocates for "an institution's preservation storage solution…be designed so that there is no single point of failure" [13]. This seems a significant omission as the dependency on storage intermediaries to rebuild files makes them easily the single point of failure that could result in irreparable data loss.

Storage intermediaries directly challenge the notion of redundancy, which relies on maintaining multiple copies of data with geographical, media and vendor diversification to mitigate digital preservation risks. The NDSA level of Digital preservation, among other requirements, calls for maximizing storage diversification and recommends having at least three copies in diverse geographical locations [9].

A storage gateway product may support multiple storage targets including on-premise disk, tape storage, and / or multiple cloud storage such as Wasabi, Microsoft Azure, and AWS S3. This on the surface provides great diversification. However, if

files are stored in compressed, encrypted, and deduplicated blobs, regardless of the location or the media, it effectively reduces multiple copies to a single one. The storage gateway becomes the single point where the organisation of multiple storage targets takes place, and holds the intimate knowledge of the transformations performed on the files, and how they can be rebuilt. An added concern of inadequate diversity is security risk, where almost all systems that share a single vulnerability may be compromised [6]. Some products consist of purpose-built appliances, which are controlled by firmware that provides the low-level control for hardware - obsolescence of these proprietary components are often the trigger for switching vendors/products.

Reliance on storage intermediaries could also turn changing vendors and products into an enormous data migration exercise, often involving restoring data through the existing storage gateway and ingesting the files anew via a different one. This could be time consuming, and ramp up significant egress-ingress costs for terabytes-scale migration between Cloud storage gateways.

A note should be added about checksums, commonly used by the digital preservation community to verify file integrity. Storage intermediaries also use checksums to help ensure file/data integrity. Some expose checksums via an API, others do not make them available as they are considered housekeeping information. When a file is "blobbed", however, checksums are typically calculated at the blob level, not file level. So if you track checksums for files and wish to compare them to the recorded values by storage intermediaries for verification, you should compare the checksum for blobs rather than the entire file.

IV. RECOMMENDATIONS

Storage intermediaries need to be seen explicitly as an integral part of the technical environment that is associated with the digital objects we wish to preserve. A key recommendation to the digital preservation community is to raise awareness and deepen understanding of them, especially how they could become the single point of failure leading to data loss or digital preservation failures.

At the next level it is crucial to understand in detail the dependency introduced by a storage gateway when evaluating storage solutions for digital preservation, knowing what exactly it does to files. The goal would be to choose a product that maintains file structure and file properties as much as possible. Doing so may lose some efficiency but this would be justified by the less intensive access requirements for archival or digital preservation storage, as opposed to storage services that need to support active operational workloads.

Not all storage gateway products transform files in the same way. The BlackPearl Converged Storage System allows you to opt out of blobbing for files smaller than 1TB, and choose between using object IDs or retaining the original file names. One can even specify if compression should be used and whether the timestamp for files is updated when the file is read at access time [11]. The AWS Storage Gateway File Gateway maintains 1:1 mapping between a file and an object, including file permissions. The S3 key name of the object is identical to the full path of the file that is written to the mount point in AWS Storage Gateway [4].

Where possible, also consider not using a storage gateway but interacting directly with storage services. This removes the dependencies on storage intermediaries entirely. AWS S3 for example can be accessed using a S3 browser, a command-line interface (CLI), or AWS console or programmatically via an API. The trade-off is between efficiency in data transfer and storage, which improves performance and delivers cost saving, and file integrity and independence.

Presenting tape or Cloud storage as mapped network drives for end user access is best avoided because of the "phantom" files issue described in section II.B and the risk of inadvertently adding filesystem artefacts to digital preservation storage. Additional workflow should be in place to filter out any hidden or irrelevant files before ingesting files to archival or preservation storage.

Dependencies on storage intermediaries need to be factored in to inform a range of digital preservation decisions:

1. Redundancy and diversification: be aware that this could be significantly reduced when multiple copies are managed and written by a single storage intermediary.
2. Exit strategy: do not accept direct access to data at storage destination as a valid exit strategy, without understanding if and how

the data may be restructured by a storage intermediary.

3. Integrity verification: Take into account the trust issue and egress charges, as discussed by D. Rosenthal [5], relating to verifying the integrity of data stored in a Cloud service. When you do decide to utilise checksums exposed by storage intermediaries or services, be aware these may not always be at file-level.
4. Vendor/data migration: plan properly with adequest time and resources. Do not under-estimate the time required to restore and re-ingest large amounts of data, and the costs, especially when data is stored in the Cloud.
5. Self-contained objects: use file packaging or container formats to store metadata needed to verify data alongside it rather than relying on special capabilities of a storage intermediary. This allows for independent data integrity verification. Formats such as BagIt [8] or Oxford Common File Layout [1,2] are good models to start from.
6. Filesystem metadata: this is useful for reference or troubleshooting but do not regard it as fixed properties and rely on it for digital preservation purposes because it can change. Timestamps e.g. cannot be used to determine the age of files.

Our final recommendation is a call for storage intermediaries vendors to standardise and make transparent the way in which data is transformed to gain efficiency, to store data in a unified layout, and allow access to it in a standard fashion. Ideally we are looking for interoperability between storage intermediaries, which keeps data intact at each storage location and allows data to be moved to a different provider without having to restore and re-ingest all files.

II. REFERENCES

[1] A. Hankinson, et al., "The Oxford Common File Layout: A Common Approach to Digital Preservation". *Selected Papers from Open Repositories 2018.* https://doi.org/10.3390/publications7020039

[2] A. Hankinson, et al., "Oxford Common File Layout Specification". https://ocfl.io/draft/spec/

[3] Amazon Web Services, "File Gateway", 2021. https://aws.amazon.com/storagegateway/file/?nc=sn&loc=2&dn=2

[4] Amazon Web Services, "File Gateway for Hybrid Cloud Storage Architectures Overview and Best Practices for the File Gateway Configuration of the AWS Storage Gateway Service", March 2019. https://d0.awsstatic.com/whitepapers/aws-storage-gateway-file-gateway-for-hybrid-architectures.pdf

[5] D. Rosenthal, "Cloud for Preservation", DSHR's blog, February 2019. https://blog.dshr.org/2019/02/cloud-for-preservation.html

[6] D. Rosenthal, et al., "Requirements for Digital Preservation Systems: A Bottom-Up Approach". D-Lib Magazine, vol 11, #11, Nov 2005. http://www.dlib.org/dlib/november05/rosenthal/11rosenthal.html

[7] IEEE SA, IEEE/Open Group 1003.1-2017 - IEEE Standard for Information Technology--Portable Operating System Interface (POSIX(TM)) Base Specifications, Issue 7. https://standards.ieee.org/standard/1003_1-2017.html

[8] J. Kunze, et al. "The BagIt File Packaging Format (V1.0)". Internet Engineering Task Force, RFC 8493. https://datatracker.ietf.org/doc/html/rfc8493

[9] NDSA & DLF, Levels of Digital Preservation, https://ndsa.org/publications/levels-of-digital-preservation/

[10] Spectra Logic, "Spectra BlackPearl Converged Storage System". https://spectralogic.com/products/blackpearl/

[11] Spectra Logic, "Spectra BlackPearlConverged Storage System User Guide", September 2017. https://support.spectralogic.com/python/documents/Spectra%20BlackPearl%20User%20Guide.pdf

[12] S. Schaefer, N. McGovern, A. Goethals, E. Zierau, and G. Truman, "Digital Preservation Storage Criteria, Version 3", 14-Jan-2021. https://osf.io/pwcvz/.

[13] S. Schaefer, N. McGovern, A. Goethals, E. Zierau, and G. Truman, User Guide for the Preservation Storage Criteria Version 3", 14-Jan-2021. https://osf.io/uqkpb/