

STORAGE SWITZERLAND

PROFICIENT OBJECT-BASED REPLICATION KEEPS UP WITH HIGH PERFORMANCE STORAGE SYSTEMS



Eric Slack, Senior Analyst

Replication is a data service that has become a standard feature in most storage systems, often used to move data off-site as part of a disaster recovery strategy. But demands for increasing storage growth are starting to outpace the abilities of traditional replication technologies and the WAN connectivity they use to transmit data. Similar to the benefits that object-based architectures brought to file systems, object-based replication has the proficiency to keep up with these high growth and high performance storage systems.

Traditional file systems manage data in discrete files which are organized in a hierarchical 'tree' structure, similar to the file - folder - directory layout presented in the user interface. This hierarchy requires a significant amount of metadata to maintain its organization and structure, metadata which is typically stored within each file. It's this metadata management and the process of opening each file in order to examine it that can create efficiency and performance problems for traditional file systems especially during metadata-intensive operations like replication. Upper limits on file system i-nodes effectively limit overall scalability as well.

Object-based file systems

An object-based file system (OBFS) creates a collection of sub-file data containers called an "object store" which

references physical locations on disk. These objects hold different types of data, like user data and the various kinds of metadata associated with file access control, object location and associated objects. Objects are identified by unique Object ID numbers (OID), which are kept in a master table and used to reconstruct files when accessed. The object architecture organizes data as a flat collection of unique OIDs, not a complex hierarchy of files, folders, directories, etc. Without the extra overhead associated with accessing and handling all this metadata, OBFSs can scale without many of the size restrictions that traditional hierarchical file systems have. They're also much more efficient, resulting in higher performance and less CPU and storage overhead.

Asynchronous data replication is the standard method for moving large amounts of data, especially in data protection and disaster recovery applications. This process allows lower bandwidth connections to reduce costs and enables scheduling of data transfers to fit business needs. This incremental, or 'change-based' method keeps a source and target data set synchronized by identifying the subset of the source volume that's been modified since the last update and just transferring that. There are a number of methods for implementing this incremental replication scheme, but not all are suited for high performance file systems.

File-based replication

Replication methods differ in the precision or ‘granularity’ with which they identify these changes between source and target data sets, and in their efficiency. As the name implies, file-based replication compares data at the file level to determine if any changes have occurred. This ‘coarse’ level of granularity means that a small change in a file can’t be distinguished, and instead causes the entire file to be identified for replication, resulting in wasted resources as much more data is handled than was actually modified. An example of file-based granularity is NDMP, or Network Data Management Protocol. It involves a process of walking the file system hierarchy to examine files for evidence of modification. Handling this file system metadata adds to the overhead and can impact overall performance of the replication process itself. However, replication at the file level does offer the ability to only transfer certain files or directories.

Block-based replication

Block- or volume-based replication essentially parses a data set into subfile blocks and keeps track of the change activity for each block. This way, when a file is modified slightly, only the blocks that are affected get marked for replication. This finer level of granularity in determining changes can result in better efficiency, compared with file-based replication, as significantly less data is handled and sent over the network to the target location. On the downside, block replication has no awareness of data structure outside of the volume itself, it simply creates a mirror image on the replication target, somewhat like a bitmap image. This means that an entire 16TB volume, for example, must be replicated and maintained on the target system, even if only 2TB of it was critical for DR. This requirement to maintain an exact, block-for-block copy at the target site means block-based replication lacks the flexibility to support other services. Keeping multiple

snapshots, for example, would not be possible since it requires the target data set to maintain more blocks than the source.

With block-based replication, unlike file replication, individual files or directories can’t be handled differently or specifically included or excluded in the replication schedule. Also, block replication has no awareness of the content of each block, which means that ‘empty’ blocks, like white space in a photo, will consume processing and storage capacity the same as those which actually contain data.

Object-based replication (OBR) like [BlueArc’s JetMirror technology](#), leverages the same ‘flat architecture’ of SiliconFS, its underlying object-based file system (OBFS), to improve efficiency and performance in the data transfer process. Compared with file-based replication, it doesn’t use the file-directory hierarchy in data handling, eliminating a significant amount of metadata processing. Object-based replication also shares the advantage of sub-file granularity that block replication enjoys, but has the content awareness to skip empty blocks, reducing the data handled and transferred. In a sense, it offers the ‘best of both worlds’, while avoiding the shortcomings of either file-based or block-based replication.

Object-based file systems contain data objects, which hold the data itself or the ‘payload’, and metadata objects which support security, directory structure and associations with other objects. They also contain the master index or ‘map’ which enables objects to be reconstructed into files. When the replication job begins, OBR doesn’t spend cycles accessing file metadata or directory structures in preparation for a data transfer, it simply sends the objects to be replicated and the master map. Then, the target system can reassemble the files, the file system structure, security, etc from the objects sent.

However, where the OBR process really shines is with incremental replication, or the regular updates that the source file system sends to the target system to keep it in sync. File-based replication spends a significant amount of resources walking the directory hierarchy looking for changed files. An OBFS can identify changed data much more quickly by looking directly at metadata objects and creates a list that tells the OBR process which objects to transfer. The result is a much faster replication process that handles and transfers far less data. For example, in tests of incremental replication on systems of small files, the JetMirror object-based replication process was *26 times faster* than file-based replication.

Traditional replication technologies don't have the proficiency to keep up with data growth in many enterprise storage environments. File-based replication lacks the granularity to effectively handle small file changes and can be overhead intensive. Block- or volume-based replication lacks the file structure or content awareness to efficiently maintain data synchronization between source and target data sets.

Object based file systems are improving the effectiveness and scalability of large data storage infrastructures and a similar architecture can improve replication in these environments as well. Object-based replication offers the file structure understanding of file-based replication and the sub-file granularity of block-based replication in a single proficient architecture. In an object-based file system, like [BlueArc's SiliconFS](#), it can reduce the storage, bandwidth and processing overhead associated with large scale data movement, while it shortens replication times.

About Storage Switzerland

Storage Switzerland is an analyst firm focused on the virtualization and storage marketplaces. For more information please visit our web site: <http://www.storage-switzerland.com>

Copyright © 2011 Storage Switzerland, Inc. - All rights reserved