Replication Redux

The "Guiding Principle of Replication" at BASIS is to use the fewest resources necessary to continuously back up the data as requested by the user and remain as current as those resources will allow. Clearly, some resource use is required, but the impact on user operations should be as small as possible and be imperceptible to the user group. In pursuit of this goal, BASIS added these improvements to shrink replication resource usage:

- Reduced the number of open files on both the target and the source
- Optimized the copying of small files and large string files from the source to the target

These changes greatly improved the efficiency of a replication job with large numbers of files to copy from the source to the target, such as when there is no initial rsync or when creating or changing a large number of string files. This article takes a closer look at these improvements.

Caching File Opens on the Target

The first improvement was to tighten control over the number of files open on the replication target. A replication target needs a file open in order to make such changes as writing or removing records. However, replication operations are a stream and encompass changes to potentially many files. It is much too expensive to open and close a file every time there is a file operation. After all, the user might be in the process of adding a million records to the same file, and opening and closing the file a million times (once for each record) is much too slow.

To resolve the issue, we keep a cache of recently used files on the replication target; the first operation on the file will open it and subsequent operations will find it already open and ready to modify. If the file has not had any operations for a couple of minutes, we go ahead and close it in order to avoid having too many files open. Unfortunately, this does not help if users were modifying many files at the same time. If users modified a thousand files within a couple of minutes, then the replication target could open all thousand files at the same time, which would lead to running out of file handles. BASIS resolved this by adding a restriction on the total number of open files in the cache. In addition to closing files when they have not been used for a couple of minutes, the total number of open files is limited to prevent overwhelming the replication target with open files even if there are near simultaneous changes to many different files on the source.

Limiting Open Files on the Source

BASIS also made changes to minimize the number and duration of open files on the replication source. Unlike the target, the replication source only needs to open files in order to check for OS-level changes and to copy files to the target. We record simple data file modifications directly to the replication log and send them on to the target without needing to open the file. However, when we detect OS-level changes to a file, we need to open the file to check the contents against the target and copy it when necessary. Originally, we would open all of the source files in order to check for changes and if we required a new copy of a file we would keep that file open until

By Chris Hardekopf
Software Engineer

the copy completed. Unfortunately, this could lead to problems. It could cause each file to remain open for an extended period of time, using up open file handles and preventing it from being deleted.

We changed how the copy works so that instead of keeping the file open until it was copied, we immediately close the file and add the name to a queue. When we are ready to actually copy the file, we attempt to open it again just for the duration of the copy. This means that only a few files are open at a time and the user is able to delete the file while it is waiting for the job to copy it since it will only be open for the minimum time necessary for the check and thereafter for the actual copy.

Copy Whole Files

Copying files from the replication source to the target is normally performed a block at a time for string files, or a record at a time for data files. This lets us efficiently copy large files as well as allow file modifications during the copy process. However, if the file is small enough, we now copy the whole file in a single operation. It is much more efficient to simply load the entire file into memory and send it to the target. This process avoids extra communication between the source and target about the file, minimizing the amount of time the file is open on the replication source. However, copying large files in this manner would take too much time and use too much memory and network bandwidth in loading the entire file into memory and sending it to the target.

Add Checksums to Minimize Traffic

Occasionally, a user will put a large string file into a replication job, requiring the replication source to copy the file to the target. If the file does not exist on the target, the job must copy the entire file. However, sometimes the large file already exists on the target and users are only modifying the source file in relatively small ways. For example, when users append to log files and only change the end of the file.

In order to handle such cases better, BASIS changed large string file replication to get checksums for blocks of the file that already exist on the target and only copy the parts of the file that have different checksums or that do not already exist on the target. Now, instead of just copying the file from scratch as it used to, the replication target iterates through the target file gathering a list of checksums that it can send back to the replication source. The replication source then iterates through the source file it is copying, comparing the checksums in order to determine which parts of the file have actually changed. The replication job only needs to send the changed parts of the file over the network to the replication target. This process requires that the replication target read the target file, and the replication source still needs to read the entire source file, but it has the potential to greatly reduce network traffic and speed up the file copy for minor changes (such as appending) to large string files.

Summary – Faster, More Robust, More Efficient

At BASIS, our belief in the concept of continuous improvement made these recent changes to replication an easy target, utilizing the feedback from the community and our own research helped us make the process as efficient and unobtrusive as possible. We look forward to finding more ways to improve the performance and reduce resource usage into the future. ■

For more information, refer to:

- *Replication Introduction* in the online documentation
- *Anatomy of a Replication Job* in The *BASIS International Advantage*