KB00114 Terminal Concentrator File Caching

This article provides details about the Terminal Concentrator (TC) file caching capabilities. The purpose of TC's file caching is to significantly reduce the number of file i/o read requests issued to the 4690 controller. The benefits include reduced controller workload and faster application loading under TC.

There are two file caching facilities built into TC:

- File caching - local disk caching of load modules needed to load a TC terminal instance. With this type of caching, a copy of the 4690 file is kept on the TC system hard drive. Example files are terminal sales applications and include .286, .386, .SRL and .DLL modules. Also included are terminal ram disk pre-load images.
- Record caching - after the application image has been loaded and during the application 'loading options' phase, TC implements an internal memory-based record cache. For non-keyed files, opened read-only by the sales application, records that are read from the controller are saved in this cache. Prior to reading the record from the controller, the cache is searched and, if found, pulled from the cache instead of from the controller. This cache greatly reduces 4690-controller file I/O during loading options.
  The information in this article applies to:

- All versions of Terminal Concentrator

MORE INFORMATION

File Caching Details

- Cache files are kept in the TSCACHE directory (i.e. \tc_nt\tscache).
- Before using the cache version of the file, the file's date, time, and size are verified to match that of the file on the 4690 controller.
- There are two CDI file options related to file caching: 'FileCacheFiles' and 'FileCacheAdditionalFiles.' There is almost never a reason to use these options.

Record Caching Details

- Record caching is only enabled during application initialization. An application is considered to be initialized until it issues its first 'wait' on I/O processor input.
- Caching is only enabled for non-keyed files opened in read-only mode.
- The records that end up benefiting most from this caching are typically application options and message files. Usually these files are read very inefficiently by the applications. They are read record by record instead of sector by sector (lots of 32 byte reads instead of fewer 512 byte reads).
- A feature of TC record caching which benefits even a single head is that all reads are expanded to sector reads.
- Imagine 80 heads loading simultaneously and the benefit that this feature provides. As a practical matter, we found that TC wouldn't work without it. At least not with existing controllers and existing networks.
- The caching algorithm includes an age-out or stale record timer. A record is aged-out after 30 seconds. The algorithm is run every 10 seconds.

- A fixed amount of memory is used for record /sector caching. The size is currently 300 records and has been since the beginning. Besides aging out, Least Recently Used (LRU) records can be purged to make room for a new record.

Diagnostic Details

- There are no separate diagnostic or reporting logs for caching other than normal TC in-memory and optional debug trace logging. Most log entries related to our caching can be found by searching our logs for the word 'cache.' The tool we use most for analyzing our logs is an editor called KEDIT which has a hugely powerful command named 'all.' 'All' hides/filters all lines in the file except for lines containing the text you are searching for and lets you easily expand between showing all lines and showing the filtered lines.