

# **The Vision of Multi-Level Caching**

A NEVEX White Paper

## **Abstract**

This white paper outlines the position of NEVEX Virtual Technologies on tightly integrating the NEVEX file-based caching technology with the Windows Server operating system cache. This white paper will demonstrate that managed interoperability between a fast media cache and the Windows main memory (DRAM) cache creates a highly-optimized multi-level caching solution that provides significant system performance gains and allows for increased VM density in virtual environments.

Andrew Flint

NEVEX Virtual Technologies Inc.

August 23, 2011

## **Background**

NEVEX Virtual Technologies was founded in 2009 with a vision of providing virtual storage solutions that further the aims of content-centric networking (CCN)<sup>1</sup> but without requiring a fundamental architectural rewrite for current networking and name services technologies. NEVEX sees a growing requirement for “data anywhere” to catch up with the current “computing anywhere” trend. Current solutions, such as Dropbox and the forthcoming iCloud, solve this need on a personal level, but lack the security and performance requirements for wide scale enterprise use.

The NEVEX vision for an enterprise class solution requires:

1. Secure dispersed data storage that separates the contents of the data from the ownership of the data (metadata). Control of the metadata, including the means and rights to access the data, remains with the data owners. The contents are dispersed across the network but with;
2. A cache layer that re-combines and provides locality of reference for active data. Effectively a “cache cloud” concept, which preloads data as close as possible to the users/applications needing it. The cache layer provides high-performance access to data, regardless of the latency to the true storage locale.

NEVEX’s initial focus is on the cache layer. The first product from NEVEX, CacheWorks, implements this on an Enterprise level. Its expansion to an Internet level may be explored in a further white paper.

## **NEVEX File-Based Caching**

NEVEX CacheWorks implements caching at the server level, utilizing local high-performance flash media as the cache drive. The NEVEX software installs into the Windows operating system itself, using the OS for protocol and driver support. The nature of the integration provides a cache solution that is transparent to both users and applications.

Most available cache solutions implement a block-based caching technique. In addition to being easier to develop, block-based caching provides operating system independence. Block-based caching generally operates at the LUN level, and lacks the file system

---

<sup>1</sup> See [http://en.wikipedia.org/wiki/Content-centric\\_networking](http://en.wikipedia.org/wiki/Content-centric_networking)

awareness that is needed to efficiently determine or enforce what data should be cached, and how it should be cached (read, write-back, write-through, etc.).

In contrast to other cache solutions, NEVEX CacheWorks employs a file-based caching technique utilizing sparse files<sup>2</sup> in the cache. The NEVEX file-based architecture provides for advanced policy management, cache coherency, and NEVEX-managed interoperability with the Windows memory file cache (multi-level caching).

Policy management is discussed below as an integral piece of multi-level caching. Cache coherency is a future feature, providing the next step in the vision of creating a cache cloud surrounding the entire traditional storage fabric. The fundamentals of a multi-node aware cache system will be the focus of a future white paper.

## **Windows Cache Structure**

Windows servers cache data in both the CPU and main system memory. There are two or three levels of data cache inside the CPU itself (L1, L2, and potentially L3), and a further level, the file cache (aka: page cache) in main system memory:

### CPU Cache

L1 Data Cache (smallest and fastest, per-core)

L2 Data Cache (larger and slower than L1, also per-core)

L3 Data Cache (largest and slowest, shared across all cores)

### Memory File Cache

Utilizes DRAM not currently allocated to applications

The data cache levels are differentiated by size and performance, with performance dictated by the distance of the cache from the active processor. Size also relates to performance, as larger caches have better hit rates but longer latency. This is the fundamental reason for multi-level caching, where the smallest and fastest caches are backed by larger and slower ones.

The L1 (Level 1) cache is generally less than 128 KB and built directly on the CPU, providing the fastest transfer speed. The L2 cache usually ranges in size from 256 KB to 2 MB, and may be situated off the CPU chip, but in close proximity. The L3 cache is also off-chip, usually ranging from 1 to 64 MB and shared across all CPU cores. Not all CPUs support an L3 cache. All levels of CPU cache use Static Random Access Memory (SRAM), which is significantly faster and more expensive than Dynamic Random Access Memory (DRAM) used in the memory file cache.

The size of the memory file cache is effectively equal to the total size of physical system

---

<sup>2</sup> See [http://en.wikipedia.org/wiki/Sparse\\_file](http://en.wikipedia.org/wiki/Sparse_file)

memory less the amount of memory allocated to applications. As application memory requirements grow, files are evicted from the cache to compensate. The memory file cache is managed by the Windows operating system software, in contrast to the CPU caches which are generally managed entirely in hardware.

In addition to data caching, the CPU also has an instruction cache to speed up executable instruction fetch, and a Translation Lookaside Buffer (TLB) to speed up virtual-to-physical address translation. The instruction cache and TLB operate independently of the data cache levels.

### **NEVEX Multi-Level Caching**

NEVEX CacheWorks uses NAND flash as a further cache level within the Windows operating system. NAND flash is approximately 10 times slower than DRAM (though still up to 100 times faster than disk), and can provide an order of magnitude or greater cache size.

The NEVEX cache follows the same principles as the CPU and file caches; a larger and slower cache backing up smaller and faster caches:

#### **CPU**

1. CPU Caches (SRAM)
2. Memory File Cache (DRAM)
3. NEVEX Cache (NAND Flash)

#### **Primary Storage**

As previously stated, the CPU caches are managed in the processor hardware itself and, as such, are outside of the scope of NEVEX's cache management software. NEVEX has, however, achieved a tight integration between the DRAM memory and the NAND Flash caches. The result of which optimizes the overall usage of the DRAM in the system, and better manages what data is or is not placed in the memory cache. Depending on the I/O workload, this can allow applications to perform faster with NEVEX caching than the application running on the NAND Flash directly.

NEVEX intends to extend this integration by providing advanced policy management, including which types of data are cached (files, directories, applications, etc.), under what caching modes (read only, write-through<sup>3</sup>, write-back<sup>4</sup>, write-around<sup>5</sup>, etc.), and which cache level to use (DRAM or Flash). NEVEX will also provide inclusive and exclusive rules governing which types of data are permitted in either cache.

---

<sup>3</sup> Every write to the cache causes a synchronous write to primary storage

<sup>4</sup> Writes first to the cache and is mirrored to primary storage when I/O is available

<sup>5</sup> Cache is bypassed, writes directly to primary storage

Before examining these in more detail, let's look at the default behavior of the integrated cache.

### Default Policy

Usage patterns demonstrate that data is either used once or more than twice, but rarely *exactly* twice. Further, a write of data only results in a read of that data in less than a quarter of instances. Using these assumptions, NEVEX can reduce the pressure on precious DRAM resources by placing active data first in the flash cache, and only promoting to the memory cache on a subsequent access. In a non-NEVEX environment, with a full memory cache, unique reads and writes continually evict other data, resulting in a smaller amount of the DRAM cache being used effectively.

If a requested piece of data is not present in either cache (first access) then it is returned to the application from its primary storage location and promoted to the flash cache but not the DRAM cache<sup>6</sup>. If the data is re-requested (resulting in a DRAM cache miss, and a flash cache hit) then it is returned from the flash cache, and promoted to the DRAM cache. Subsequent requests and eviction from DRAM is handled normally by the Windows operating system itself.

With NEVEX, by default, a write operation will also only populate the flash cache (equivalent to a first read). Subsequent re-reads will promote data to DRAM as described above. Subsequent re-writes will update the data in the flash cache, but will not, however, result in promotion from the flash to DRAM cache.

Also by default, the DRAM and flash caches operate in a strictly inclusive manner. This means that a copy of any data promoted from flash to DRAM is also kept in flash (until evicted by NEVEX cache maintenance policies). If data is re-read after being evicted from DRAM, it is served from flash (and re-promoted to DRAM) rather than its primary storage locale, extending the file cache from the size of available DRAM to the size of the flash device.

The combination of flash to DRAM promotion and an inclusive cache level, creates a more efficient use of available DRAM, and provides the Windows server with an order of magnitude larger effective total file cache.

### Policy Management

NEVEX CacheWork's default policy for the two-level DRAM and flash cache combination provides a significant performance benefit. Further benefit, however, becomes apparent through the planned utilization of the extensive management controls NEVEX provides over the cache operation. Tuning both included and excluded data combined with the provided control over the interoperability between the two cache

---

<sup>6</sup> Unless the data can be promoted to the memory cache *without* using incremental DRAM. Windows allocates DRAM by ranges, meaning there could be allocated but "empty" memory space available.

levels allows performance to be tuned to a specific set of requirements or to a specific environment.

CacheWorks currently provides the ability to specify which data is accelerated through the NEVEX flash cache, and which is excluded. This includes types of files, directories, applications, etc. Further, in some cases, such as Microsoft SQL Server, it is possible to specify caching policies down to the *sub-application* level.

For example, consider a scenario where an SQL Server instance has both a constant use transactional database, as well as a limited use historicals archive. It is possible to specify that the performance sensitive and I/O intensive transaction database is accelerated, and the limited use historicals database is excluded. In so doing, the non time critical large data set is prevented from flooding the cache, and instead, the time critical queries in the transactional database are given priority.

NEVEX intends to extend the functionality of the policy management system to support the two-level cache interaction. This will add the capability to define which data is included or excluded from either cache, as well as distinct caching modes per type of data, per cache level.

Below is a simplified view of the policy settings, but it illustrates the level of control provided over the cache system:

	<b>Flash Cache</b>	<b>DRAM Cache</b>
<b>SQL Server DB 1</b>	Read, Write-Back	Exclude
<b>SQL Server DB 2</b>	Read, Write-Through	Exclude
<b>Exchange Server</b>	Read, Write-Around	Read, Write-Around
<b>App 3</b>	Read, Write-Back	Read, Write-Back
<b>App 4</b>	R/W-B, Evict Policy A*	Exclude
<b>Data Set A**</b>	R/W-T, Evict Policy B	Exclude
<b>Data Set B</b>	Cache Now, Never Evict	Read, Write-Back
<b>Media Files</b>	Exclude	Exclude

\*Eviction policies based on Least Recently Used (LRU), Least Frequently Used (LFU), Size, etc.

\*\*Data Sets based on Filenames (pattern match), Directory/Path, File Type (extension), Min/Max File Size, Process ID/User, etc.

Granular policy control over the cache system allows for the reservation of the DRAM cache, the fastest level, for extremely important data, the use of the fast flash level for important data, and the bypassing of both caches for non-essential data. This non-essential data could include data that is known to be read or written only once. This kind of policy significantly improves performance and reduces latency for applications, users, and data with the greatest need.

## **Improved VM Density**

NEVEX CacheWorks is installed on the Windows Server operating system, regardless of whether the OS is running on a physical server (installed on bare metal) or inside a virtual machine (Windows Server as the Guest OS or Hyper-V Host OS). In either case the optimized interoperability created between the flash cache and the Windows DRAM cache provides significant system performance gains. In addition, there is a further benefit for virtual environments: increased VM density.

Virtual Machine (VM) density is the measure of the number of virtual machines that can be supported per physical server. The greater the VM density, the less infrastructure required to support a given number of virtual applications, and therefore, less cost per application. The core density of modern processors provides for very high VM density per server, making the limiting factors the available memory and storage I/O.

Available memory directly relates to the number of VMs that can be supported, but the I/O needs are slightly more complicated. Storage I/O has a different profile in virtualized environments. The varying levels of I/O demands from all the VMs are merged through the virtual layer onto a single physical server, creating a constantly high I/O level versus the bursty profile of a physical server. This is often referred to as the “I/O Blender” effect, which causes a greater storage I/O strain.

The multi-level cache management provided by NEVEX reduces both the memory and I/O issues that limit VM density. The larger cache size provided by the flash device, versus the smaller Windows file cache in DRAM, provides for a 5X or greater improvement in IOPS (workload dependant) effectively removing the I/O blender as a limiter. Further, the flash cache can be configured effectively as a replacement for the file cache (bypassing the DRAM cache) for most or all data, allowing more physical memory can be applied to the VMs.

## **Conclusion**

In sum, NEVEX’s file-based caching solution, designed on and for the Windows Server operating system, makes it uniquely suited for integration with the Windows caching methods. NEVEX’s tight integration of its file-based caching technology with the Windows DRAM cache, has created an industry unique a multi-level caching solution.

This white paper demonstrates that the multi-level cache creates an order-of-magnitude larger effective file cache while preserving DRAM for the most active data, providing significantly improved system performance. The future policy-based control over the combined cache solution will allow for further performance gains through cache optimization, and increased VM density in virtual environments.