DAFS-Enabling Data Center Applications

Jeffrey Carter, Matt DeBergalis, John Gillono, and Arthur Lent, Network Appliance

The Direct Access File System (DAFS) protocol is a new file access method designed to provide application servers with high performance low latency access to shared storage pools over Fibre Channel, Gigabit Ethernet, InfiniBand and other VI-compliant transports in data center environments.

Designed from the ground up to take full advantage of these next generation interconnect technologies, DAFS is a lightweight protocol that enables applications to directly access transport resources. Consequently, a DAFS-enabled application can transfer data from its application buffers to the network transport, bypassing the operating system while still preserving file semantics. In addition, since DAFS is designed specifically for data center environments, it provides data integrity and availability features such as consistent high speed locking, graceful recovery and fail-over of clients and servers, fencing, and enhanced data recovery.

All of this translates into high-performance file I/O, significantly improved CPU utilization, and greatly reduced system overhead due to data copies, user/kernel context switches, thread context switches, interrupts and network protocol processing.

DAFS vs. Traditional File Access Methods

As mentioned above, DAFS greatly reduces the overhead normally associated with file access methods. Figure 1 compares three file access methods: local file system access, network file system access, and DAFS access. In the case of local or network file systems, data is copied from the disk or network subsystem into a buffer cache, and then copied into the application's private buffer. File access over network file systems incurs additional data copies in the networking stack. Some operating systems can bypass the buffer cache copy in certain cases, but all reads over a traditional network file system require at least one data copy.

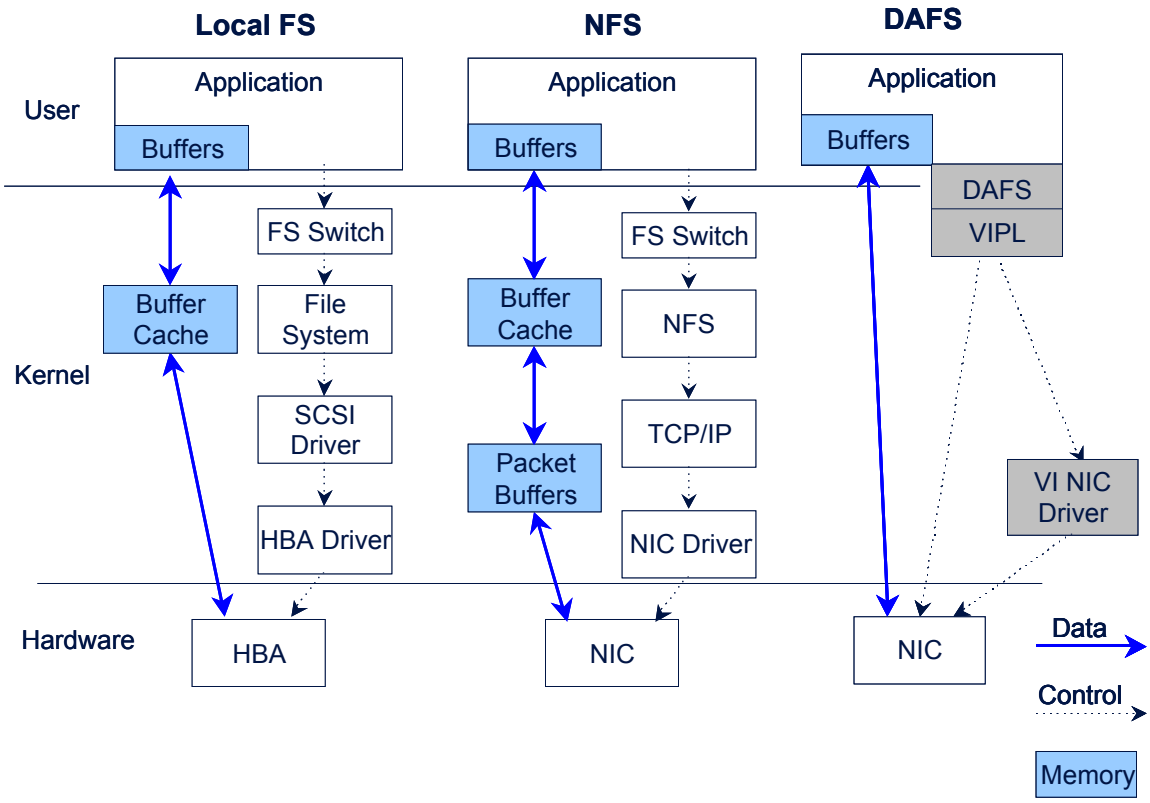

Figure 1 – Data Transfer Overhead

DAFS has a fundamental advantage over other file access methods when reading data. By using the remote memory addressing capability of transports like VI (Virtual Interface Architecture) and InfiniBand, an application using the DAFS API can read a file without requiring any copies on the client side. Using the "direct" DAFS operations, a client's read or write request causes the DAFS server to issue remote DMA requests back to the client, so data can be transferred to and from a client application's buffers without any CPU overhead at all on the client side. The DAFS write path is also efficient; to avoid extra data copies on write requests, a traditional local or remote file system must lock down the application's I/O buffers before each request. A DAFS client allows an application to register its buffers with the NIC once, which avoids the per-operation registration overhead.

DAFS Client Implementations

Applications can take advantage of these capabilities in several ways. The first is through a user library that implements the DAFS protocol and is loaded as a DLL or shared library. Alternatively, an application may access a DAFS server transparently through a loadable kernel module. These two client implementations are shown in Figure 2.

The first method shown is a user library that implements the DAFS protocol. A user space implementation of DAFS offers the greatest potential for increased I/O performance, by taking advantage of its transport's ability to directly access the networking hardware from user space and providing an appropriate API for asynchronous zero-copy I/O. The DAFS library calls into the kernel to set up an initial end-to-end connection, which can then be safely accessed from user space. The NIC (Network Interface Card) implements the necessary protection to prevent any other processes from accessing or interfering with the connection, so applications can transfer data directly to the NIC, avoiding the overhead of data copies and system call context switches while still enjoying the security benefits of traditional kernel-based networking implementations. The combination of direct initiation of I/O and the elimination of data copies reduces operation latency and increases bulk data transfer throughput.

To maximize performance, the DAFS library exports an API that gives the user application explicit control over NIC DMA access to its address space. The library API also exposes the asynchronous nature of the server interaction, allowing an intelligent application to manage overlapped data transfers.

The disadvantage of the user library approach is its lack of compatibility with the usual system call interface to the host OS file systems, requiring applications to be modified to take advantage of these capabilities. The benefit is optimal performance. This approach, therefore, is intended for high-performance applications that are sensitive to either throughput or latency, or applications that can make use of the extended DAFS services made available through the user API.

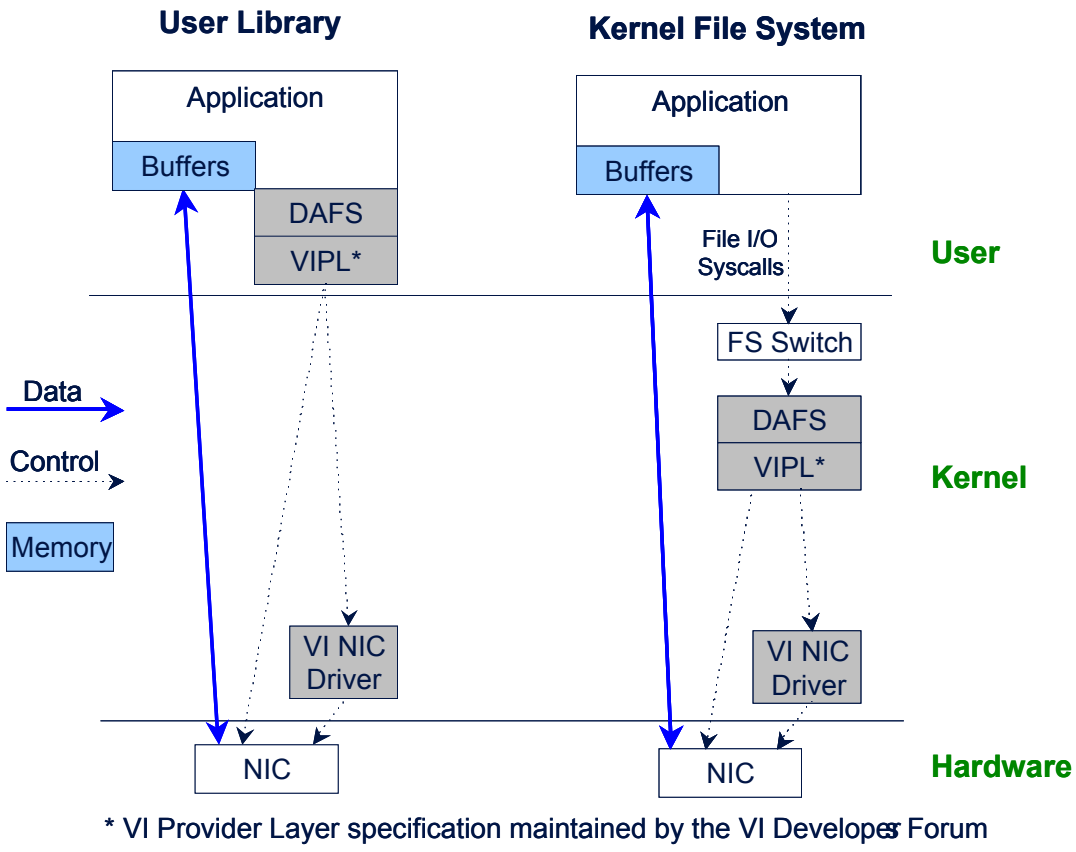

Figure 2 – DAFS Client Implementations

The second client implementation shown in Figure 2 is a loadable kernel module. In the Unix world, this takes the form of a Virtual File System (VFS). In the Win32 world it is called an Installable File System (IFS). The VFS/IFS fits in the traditional place in the operating system architecture where new file systems are added. It is a peer both to the other "remote/redirector" file system implementations (e.g. NFS and CIFS), and to the local file systems (e.g. ufs, FFS, NTFS, FAT, VxFS). Under this approach, when an application uses the standard kernel interface to read from or write to a file, the VFS/IFS layer passes the request on to the individual file system responsible for that particular file. If the file is on a DAFS file system, then the kernel passes the I/O request to the DAFS VFS/IFS, which issues DAFS requests to a server. Like other remote file systems, each call to the VFS/IFS layer may map to one or more over-the-wire protocol requests to a remote file server.

The advantage of this type of DAFS client implementation is that applications can use it transparently, just like other remote file system implementations (NFS, AFS, DFS for Unix, CIFS for WinNT/Win2k). Applications can enjoy many of the advantages of DAFS without having to be rewritten to the DAFS API. Performance is limited by the kernel

transitions involved in any operation to a VFS/IFS, but the kernel DAFS implementation still uses remote DMA and other VI capabilities, so it consumes significantly fewer CPU cycles compared to other remote file systems under similar loads for a given client platform. Even in the worst case, over-the-wire data transfer rates (both single stream and aggregate) will be comparable to local file system access to Fibre Channel connected (direct attached) block storage arrays. Finally, the application still benefits from the non-performance advantages of DAFS, including improved locking and fail-over semantics.

Summary

Applications can take advantage of DAFS in several different ways. Applications that run directly on top of DBMS systems automatically benefit once the database program is DAFS-enabled. Applications that are sensitive to either throughput or latency achieve maximum performance by using a user DAFS library. Alternatively, for applications where transparent access to DAFS is more appropriate, a Loadable Kernel Module delivers all the data management and ease of use advantages of file mode access, with similar or better performance than local access to Fibre Channel connected (direct attached) block storage arrays.

The DAFS Collaborative was formed in June 2000 by Network Appliance, Intel, and other leading systems and storage networking vendors, with the goal of making the Direct Access File System protocol available to the industry. The group is soliciting industry review and feedback before submitting the new file system to an appropriate standards body.

The DAFS Collaborative encourages broad industry participation. Additional participants are welcome and can join online.

For additional information, contact

Werner Glinka
Executive Director
DAFS Collaborative
650 851 5909
fax 851 5987
werner.glinka@dafscollaborative.org

or visit **www.dafscollaborative.org**