DVX Architecture

Datrium®

WHITEPAPER

Table of Contents

DVX provides autonomous compute and data services for virtualized applications, challenging both the rigid scaling limits of traditional servers and arrays, as well as the server lock-in of hyperconverged platforms.

Introducing DVX Architecture

We introduced the DVX in early 2016 to simplify the scaling and management of private clouds. Datrium's founders believe rich data management features and secure, scalable, and autonomous platforms are essential to achieve the potential of cloud services and dissaggregated hyperconverged infrastructure (DHCI).

Applications must store all data on high-performance, solid-state storage devices (flash SSDs) on the server, as close to the running applications as possible, but with the resiliency, availability, and autonomous data services expected from enterprise-class products.

We also recognize the need to support a mix of server configurations to provide scalable compute resources. Clustering approaches require all nodes within a cluster to be configured identically, or at least nearly so. We believe it's unreasonable to expect homogeneity and lockstep, datacenter-wide server upgrades, so we designed DVX to work with a flexible mix of servers.

In practice, businesses want to keep hardware in service as long as possible, while introducing new technology regularly – often resulting in a mix of both rackmount and blade servers, as well as models from different generations and vendors. Further, customers prefer to allocate hardware resources based on application needs, configuring appropriate amounts of CPU, memory, and I/O resources. They prefer the granularity of configuring host resources per application, rather than deploying entirely different clusters for each app.

DVX provides autonomous compute and data services for virtualized applications (running in VMware and RedHat/KVM VMs, as well as Docker Containers on bare metal Linux hosts). By allowing an elastic mix of Host configurations, DVX challenges both the rigid scaling limits of traditional servers and arrays, as well as the server lock-in of hyperconverged platforms. DVX also provides the option of Blanket Encryption for truly private clouds and autonomous copy data management, including wide-area asynchronous replication.

With DVX, we introduced autonomous data services. DVX splits active data from durable protection copies, so resources for I/O speed and active data can be provisioned and scaled independently of data protection needs. DVX Hosts (Datrium-branded servers sold and supported by Datrium) provide compute resources and server-side, solid-state drives (SSDs) to cache active data, while DVX Data Nodes provide scalable protection and durability via high-capacity storage (drives).

This paper introduces the DVX architecture. DVX innovations combine to create a new model that eliminates the expense, performance bottlenecks, and management headaches of traditional approaches.

Architecture Overview

As shown in Figure 1, the DVX system comprises two components:

- DVX Host Software on each Host (also known as the "hyperdriver") manages all active data for the VMs within that Host. It provides scalable I/O performance, availability, and data management capabilities.
- The DVX Data Node pool provides persistence and resiliency by storing a protection copy of all data in the clusters as well as storing point-in-time snapshots of older data. In regular operation, these copies are write-only, but the Data Nodes also provide streaming read performance for fast flash uploads as well as cluster coordination for simple management.

DVX's split provisioned architecture eliminates traffic between Hosts except in certain transient cases. That's a significant advantage over Hyperconverged Infrastructure (HCI) and Software-Defined Storage (SDS) models.

At the top of Figure 1, there are compute and data-caching servers or Hosts (each of which may have differing amounts of CPU, memory, flash, etc.). Customers can choose from Datrium Select Hosts with Datrium hardware support, or off-the-shelf servers and SSDs from their preferred vendors. These Hosts contain:

- General Host components: This includes the VMware vSphere ESXi, RedHat Virtualization (KVM on Linux), and Docker Containers running on bare metal RedHat or CentOS Linux. vSphere or Linux software controls all have access to physical resources like network, CPU, memory, and flash SSDs. These parts are procured and supported independently.
- DVX Software is automatically installed and maintained in each Host (as a vSphere Installable Bundle (VIB) with VMware and with self-installing executables on Linux). It provides all autonomous data management functions (compression, deduplication, encryption, snapshots, replication, Data Node RAID, space reclamation, rebuilds, etc.) and presents a single NFS datastore to the Hosts.

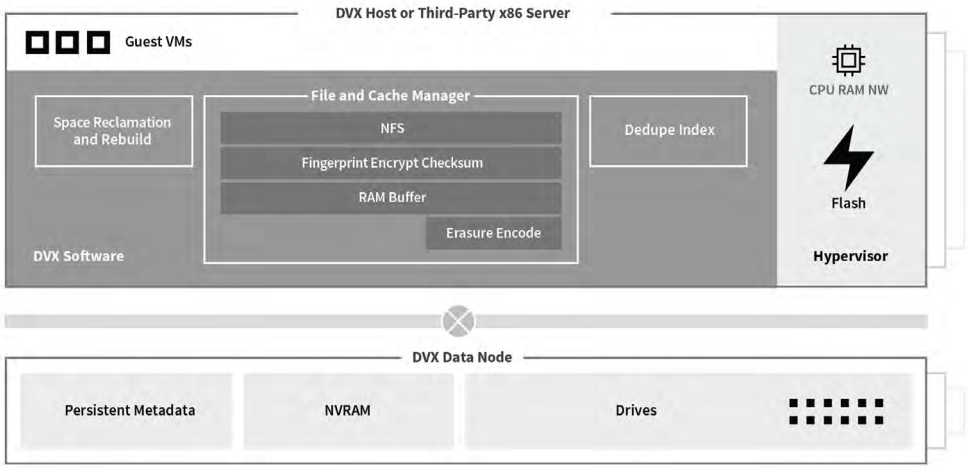

Figure 1 – DVX Components

See the Datrium DVX Data Node (up to 10 supported) at the bottom of Figure 1. NVRAM is mirrored across controllers over the LAN. Each Data Node contains dual active/standby controllers with redundant NVRAM and Ethernet interfaces, as well as disk or flash.

Logically, a DVX system presents a single storage pool, with 1-32 datastore namespaces.

The Data Nodes in DVX present a "pool" of drives to protect the active data in flash SSDs on the Hosts. DVX Host Software stores protected copies of the active data to this pool in very wide, erasure-coded stripes. In addition to providing more protection capacity, each Data Node also linearly increases the write bandwidth, and it dramatically decreases the amount of time it takes to rebuild a drive after a failure.

The Datrium DVX Specifications datasheet provides the hardware and software requirements for third-party servers as well as the environmental and physical details of the DVX Data Node.

Encryption and Security

We believe that if it's necessary to encrypt any data, then all data should be encrypted, everywhere practical, as soon as possible. That's true for a very simple reason: the administrators responsible for setting encryption policies are rarely the users that create data – it's difficult or impossible for them to predict which data requires encryption.

To ensure maximum security, we also believe data should be automatically encrypted everywhere: whether in use (cached on the Host), in flight (traversing the network), or at rest (cold, protected copies stored on the Data Nodes).

Datrium Blanket Encryption (see Figure 2) provides the industry's first end-to-end autonomous encryption solution that doesn't sacrifice storage efficiency. Previous solutions from other vendors forced customers to choose between:

1. Software-based encryption, either in the guest or at the vSphere Hypervisor level. Unfortunately, non-DVX software-based encryption completely eliminates any compression/dedupe benefits (because all data is effectively randomized).
2. Array-side encryption using either software running in an array controller or self-encrypting drives. These techniques only protect data at rest and leave cached data, and data traversing the network, vulnerable to exposure.

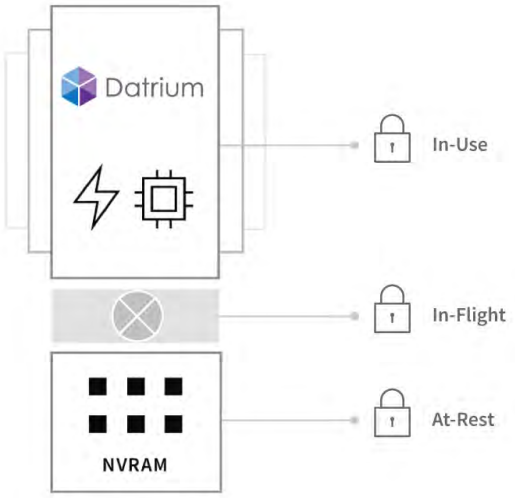

Figure 2 – Blanket Encryption

Only Datrium Blanket Encryption allows you to encrypt data everywhere, without losing the storage efficiency of dedupe and compression.

We can do that because DVX software runs at both ends of every network hop. Software running only on the Hosts, or only on a storage array, simply can't achieve similar blanket encryption.

As shown in Figure 2, all data is fingerprinted (for subsequent deduplication) and then immediately compressed and optionally encrypted before further processing or storage anywhere in the private cloud. Once DVX receives new data, encryption is performed as soon as possible to ensure maximum security.

Write Path

When a VM or container issues an individual write request (a block of data typically on the order of 4- to 64 KB), several things happen fast as DVX quickly and safely replicates a copy of the data off the Host and returns control to the VM.

1 As shown in Figure 3, the Hypervisor software redirects each VM's I/O requests to the NFS datastore presented by DVX.

NFS is treated as an API for reads and writes – the protocol always terminates within the client Host. Writes are always checksummed and compressed before sending them. All data to and from the Data Node is always compressed for network efficiency.

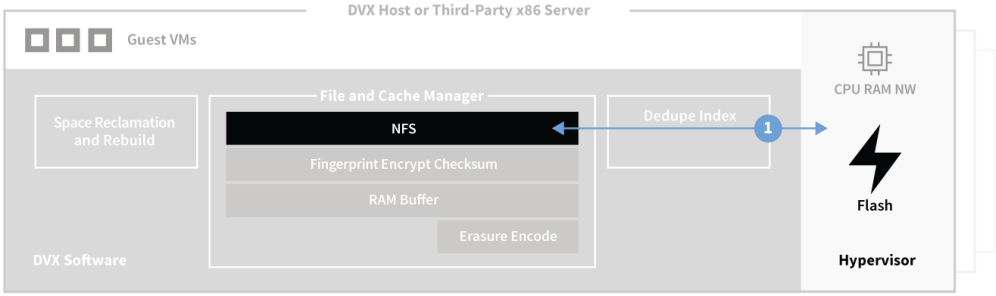

Figure 3 – NFS Redirect from Hypervisor

The DVX Host Software computes a cryptographic fingerprint of the raw data before encryption. Next, the Host Software writes a compressed (and optionally encrypted) version of this block with ECC codes added directly to a local RAM buffer.

2 Simultaneously, the Hyperdriver software synchronously writes the compressed/encrypted block over Ethernet to a non-volatile log on a Data Node that's mirrored to NVRAM in another controller (as shown in Figure 4). That's a low-latency (sub-millisecond) operation because it's writing to memory rather than disks or SSDs.

Promoters of HCI or software-only storage products may object that NVRAM is "proprietary." DVX, however, allocates just a portion of each controller's RAM and makes it non-volatile with a battery-equivalent power source. That results in much lower latency for writes than an SSD.

Once the data is safely stored in externally mirrored NVRAM, the Data Node sends an acknowledgment back to the Host, allowing the guest VM or container to continue issuing I/O requests.

Note that DVX only replicates compressed data over the wire to the Data Node (a substantial savings in network bandwidth relative to traditional techniques).

At this point, the compressed block is safely stored in both RAM locally on the ESXi host and in a power-safe log on the Data Node for safekeeping. If the guest issued a synchronous write request to its vDisk or persistent volume, it's no longer blocked and can continue issuing requests.

In regular operation, the non-volatile log on the Data Node is write-only. The Host Software reads from its local RAM buffer for subsequent processing (dedupe and RAID) rather than from the Data Node. The non-volatile log on the Data Node serves purely as an insurance copy to allow replay in the event of component failures or a power outage.

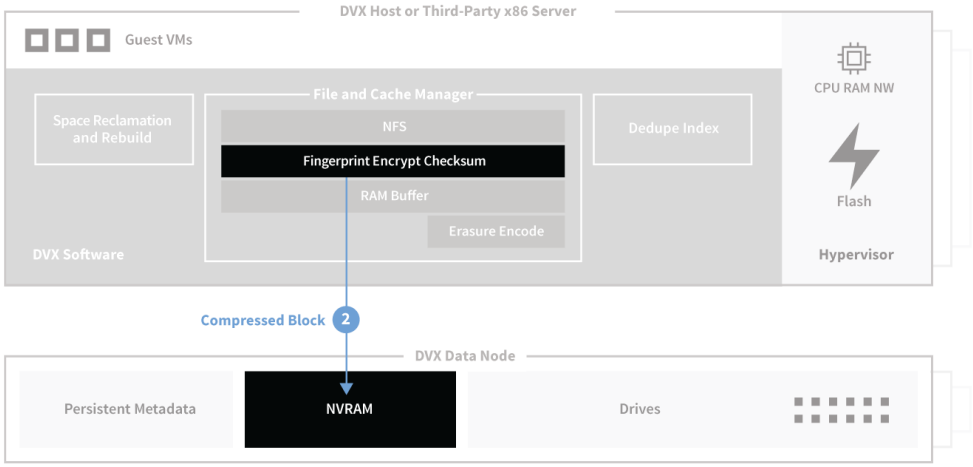

Figure 4 – Low-Latency Write of Compressed Blocks

DVX uses scalable Host resources to drain the log. Unlike traditional arrays that depend on the limited resources within an array controller, DVX Host Software (on each Host) removes the non-volatile log entries written by that Host (after storing a durable copy on the Data Node as described below). All RAID computation, lookup, and I/O required before draining NVRAM are performed with scalable Host resources. The finite resources in a SAN array controller can become a bottleneck, causing I/O to be blocked while NVRAM is drained. DVX isn't limited to the resources in a single controller.

Further, because DVX manages both the client-side writing on each Host and the server-side receiving on the Data Node, it can balance the overall load dynamically on the non-volatile log. Each Host receives a dynamically adjusted NVLOG quota allocation (based on observed I/O activity). DVX continuously monitors NVLOG usage by each Host. If an intense write load threatens to overflow the log, Hosts re-prioritize log draining to eliminate the problem.

As a failsafe in the event of down or misbehaving Hosts, DVX software on the Data Node can also spool NVLOG contents to the drive pool for longer-term retention and overflow protection.

Packing, Dedupe, and RAID

DVX software performs most of its work at this stage. All of this work happens asynchronously, out of the write path, with no blocking of guest I/O.

Packing for I/O Speed

First, the DVX software packs multiple compressed (and optionally encrypted) blocks in RAM into larger groupings that are deduplicated and stored in local Flash SSDs in the Host. Packing is done primarily for three reasons:

1. Disk drives perform poorly with small, random I/O. Packing allows us to send only large, sequential writes to the drives in the Data Nodes (a wide erasure-coded stripe at a time).
2. Flash SSDs also have problems with small random writes (both performance and endurance problems). Packing allows DVX to perform only large writes to the local Flash SSDs.
3. A DVX Data Node is designed for streaming writes and flash uploads, not random block I/O. The Data Node drive pool design challenge is to accept writes fast and upload to Host flash quickly. Packing not only allows DVX to write in large sequential blocks, but it also organizes the data for quick retrieval when re-populating Host flash.

DVX's ability to dedupe globally and to perform extremely fast flash uploads is important for two reasons:

1. It provides optimal performance after Host failures. Applications restarted on surviving servers quickly return to optimal, local-I/O performance, with only the first miss within a region reading from a Data Node. In practice, far more data is returned on the first miss than was requested, effectively "reading ahead." All subsequent reads within the region will be serviced from local flash.
2. It also allows vDisks and Persistent Volumes (PVs) to be easily and quickly cloned between physical Hosts, making writable copies of the same data available to process in parallel on separate, distinct hardware. That's an increasingly important workflow for many industries.

.

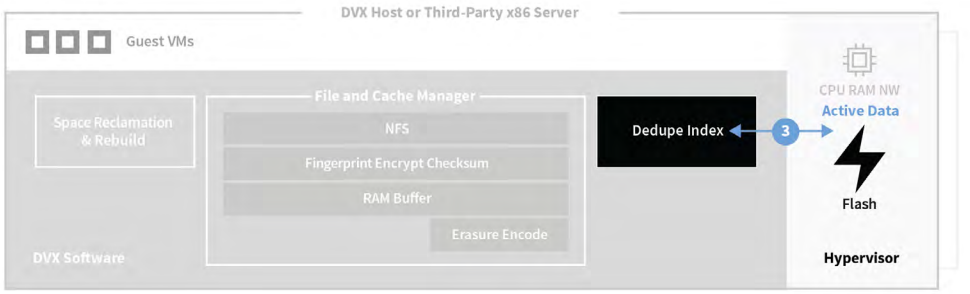

Figure 5 – Active Data Stored in Local Flash

When writing asynchronously to local flash and drives in the Data Node pool, the DVX software goes to great lengths to pack these compressed blocks for efficient retrieval. The software packs based on both spatial and temporal constraints, grouping blocks that are updated together within a virtual disk as close to each other as possible (because they will likely be retrieved together). This compressed packing allows DVX to accelerate writes and solves the VM I/O blender problem.

An individual LUN on a traditional SAN or scale-out array may contain dozens or even hundreds of VMs and virtual disks, all being updated simultaneously. Such arrays aren't aware of separate vDisks – they necessarily treat unrelated changes within a LUN from different VMs identically. A LUN-based array simply can't store related changes together within a vDisk. DVX understands VMs and vDisks natively.

Because LUN-based arrays are unable to guarantee that related changes within a given vDisk will be stored together, thrashing or other inefficiencies may occur as pieces of a vDisk are read from all over the array.

Active Data Written to Flash

3 As shown in Figure 5, each Host deduplicates the variably-sized groupings inline and stores them in local flash. As described above, DVX creates a cryptographic-strength fingerprint very early in the write pipeline (before encryption) to uniquely identify each grouping of data. These fingerprints are maintained in an index.

If the fingerprint for a newly created grouping isn't already in the index, DVX writes the group of compressed (and optionally encrypted) blocks to Host flash. DVX also stores the fingerprint in the index, along with mapping information to locate the block subsequently.

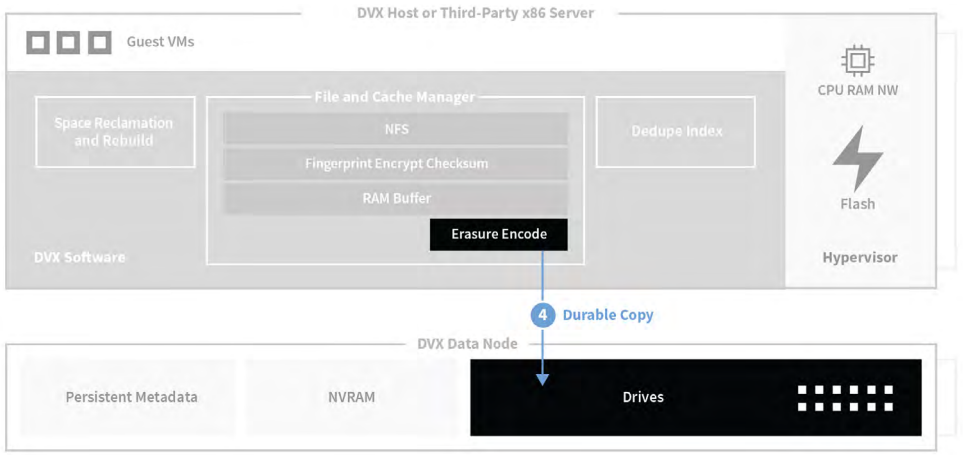

Figure 6 – Durable Copy Synchronously Written to Data Node

If the fingerprint was already in the local index, it just updates the mapping information with a reference to the existing group in flash, discarding the duplicate group currently in memory (note that this saves a flash write).

Durable Write and NVLOG Truncate

4 DVX Host Software also replicates a durable copy of the compressed and packed blocks to a Data Node as a wide, erasure-coded stripe across the pool of drives (as shown in Figure 6). The erasure-coded stripe is on the order of 10 MB, which the Host software writes in smaller chunks to individual drives. RAID parity calculations, ECC, and drive pool addressing I/O are all performed by the Host software on each physical Host.

Once the durable copy is safely written to the drive pool, the log entry in NVRAM in the data pool and the locally cached copy in Host RAM is removed.

DVX performs all durable write processing proportionally within each Host. That's a significant advantage over traditional storage, which limits the resources available for processing. Because a durable copy is stored safely on the Data Nodes, DVX never wastes space in Host flash for RAID or erasure-coding, nor does it store redundant copies in two-way or three-way mirrors (as most HCI and SDS solutions require). Further, unlike traditional approaches, all data in Host flash is compressed and deduplicated inline as the data is written.

Read Path

DVX can read data from Host RAM, Host flash, Data Node drives, or NVRAM in the Data Nodes. DVX optimizes reads to retrieve data from the fastest media. (Data is only read from NVRAM on a Data Node in the event of a hardware failure.)

Host RAM is the most trivial case: a guest VM or container reads a block almost immediately after it's written. Such blocks are still in the RAM buffer and are returned immediately. In practice, the guest VMs themselves usually avoid issuing an I/O request altogether in this case as the blocks are likely still in the guest's I/O buffer.

Local Flash Read

In almost all other cases, read requests will be satisfied by the local SSDs. We recommend sizing the SSDs on each Host to hold all active data for VMs hosted by that server. As stated previously, that's economically feasible because DVX uses inexpensive commodity SSDs with no RAID overhead and modern dedupe/compression techniques. (Note that each Host only needs enough flash to hold all data for VMs on that Host, not for all VMs in aggregate.)

DVX stores all active data in local flash. When a guest issues a read request at a given offset within a vDisk or PV, the DVX Host Software consults its internal maps and fingerprint index to locate the appropriate block in flash, returning the data without ever issuing an I/O request to the network.

Traditional arrays always require an over-the-network I/O unless the guest VM still has the data in its extremely small I/O buffer. Even with all-flash arrays and large, write-through caches on Hosts, queuing and network latency can have a critical impact on performance. By eliminating network reads, we significantly reduce queuing delays.

There are, however, four far less common scenarios when a block might not be found in local flash:

1. Guest VMs or Containers may have been recently "vMotioned" from another Host.
2. VMs or containers might boot on Hosts with new or empty flash devices (after a server hardware failure, for example).
3. A faulty SSD might return incomplete or invalid data, requiring a read from the durable copy.
4. A given Host might not have large enough flash SSDs to store all active data.

Remote Flash Read After vMotion

Immediately after vMotion, a VM's blocks still reside in the origin Host's flash. As the destination Host has never seen this VM read or write anything locally, the requested block may not be stored in the destination Host's flash.

We know that reading from flash, even with an intervening network hop, is orders of magnitude faster than reading from a disk drive. So, DVX preferentially reads from remote flash devices before reading from the drive pool serviced by the Data Nodes.

When a guest application or operating system reads a block on the destination Host, DVX consults its local index and finds that the block is missing. Because DVX is a distributed system, it knows that the block is still stored in the origin Host's flash and issues a request to retrieve it from that Host.

Dedupe may obviate even this case: if identical data is already in-use by another VM on the destination Host, the index lookup will succeed, and DVX will read the data from local flash.

Once DVX reads the information from the remote Host, it also stores it in flash locally, so subsequent requests won't need to traverse the network. When the entries in remote flash are no longer referenced by any guest, space reclamation will free up flash in the origin Host.

Note that, in practice, the entire region of blocks surrounding the requested block is retrieved from remote flash. This prefetching provides further performance gains – subsequent reads are likely to be in local flash already.

Further, if an administrator places a vSphere host into "maintenance mode," all of the VMs on that host will be vMotioned elsewhere, but DVX will continue to make the flash contents available. DVX Host Software continues to make flash available, even during maintenance mode. Remember that flash is persistent storage; even if the Host undergoing maintenance is rebooted or power-cycled, the flash contents will remain valid and available.

Data Node Read With Empty Flash

The most uncommon situation of all is when none of the blocks is stored in flash on any host. One situation where this might occur is when a guest boots after replacing failed hosts or flash SSDs.

DVX optimizes the storage format in the Data Node drive pool for precisely this situation. Guests tend to access vDisks and PV in logically consecutive regions. Data that's written together within a vDisk or PVs will usually be subsequently read together as well. Therefore, DVX stores large clumps of the most current version of vDisks and PVs together rather than spreading data in discontiguous pieces all over the disk pool. Accessed regions are uploaded as a contiguous stream upon the request of any individual block within the region, providing a significant degree of read-ahead. In effect, entire vDisks or PVs are quickly uploaded to flash. DVX thus satisfies all but the initial read request within a region from local flash rather than from the Data Node disk pool over the network.

Legacy storage systems treat all blocks from any vDisk or PV the same. Because they're built for random block I/O, they're incapable of keeping blocks within individual vDisks together (much less keeping recent changes within a vDisk together).

"Cold cache" boot storms with traditional arrays and simple caching (even with all-flash or hybrid arrays) can become a performance nightmare. DVX avoids these problems by optimizing the layout of data within the Data Nodes for quick and efficient uploads to the Host flash. DVX uses Host flash to store all active data, not as a random block write-through or write-back cache.

Faulty Flash Devices

As data is written, DVX protects it with error-correcting-codes (ECC) and cryptographic fingerprints. As data is read, the system continuously verifies that the retrieved data matches the data integrity codes.

If the Host Software discovers an uncorrectable error within local flash as data is read (e.g., with a partial or complete SSD failure) the software will retrieve correct data from the durable copy on the Data Node drive pool, store it in a new location locally, and mark the previous location as bad (DVX will fail the entire device after too many uncorrectable errors).

That means while Host flash failures will impact performance, DVX still guarantees data integrity. Your data is safe, even if your Host hardware fails. By comparison, Host failures affect both performance and data resiliency/durability with hyperconverged/SDS approaches.

Managing Performance

Despite world-class performance, DVX prioritizes data integrity and availability above performance. While accelerating reads with server-side flash is important, nothing is more important than returning correct data upon every request.

Note that such assurances are only possible because DVX provides end-to-end coverage. A traditional array can only protect the integrity of data after it traverses a network (dutifully protecting data that may have been corrupted by intervening network or Host-side problems). DVX protects your data before storing it locally or sending it over the network.

Undersized Host Flash

The final possibility that would mandate reading from a Data Node is when there isn't sufficient flash capacity on a given Host to store all data in use by that Host. DVX was designed to make that as simple and inexpensive to address as other Host resources like CPU and memory.

The DVX user interface informs the user if a given Host has insufficient flash headroom. If so, the solution is simple: move guests to other Hosts, add more, or larger flash devices. Note that while insufficient flash will affect performance, it doesn't affect data durability – your data is always safe.

To reiterate, DVX makes flash capacity extremely inexpensive by eliminating RAID/mirroring overhead, compressing and deduplicating all data, and leveraging commodity server SSD pricing. Administrators should size the flash capacity within every Host to sufficiently store all data.

DVX degrades to familiar caching behavior if there isn't sufficient flash capacity. We feel strongly that it should never be necessary, however. Administrators should provide sufficient flash capacity on every Host to store all active data for the guests running on that Host.

Storage administration mostly boils down to provisioning, capacity management, and I/O load balancing. With DVX, administrators only deal with the business-meaningful abstractions of VMs, containers, virtual disks, PVs, Hosts, and a datastore. That simplifies administration tremendously.

Administrators can dynamically balance workloads by vMotioning VMs to other Hosts with more headroom (hot-adding Hosts or flash resources, as required). Further, administrators can also dynamically enable Insane Mode on an individual Host with available CPU headroom to handle short-term load spikes (avoiding vMotion altogether).

DVX makes balancing I/O performance similar to balancing a CPU or memory load. Total aggregate load is balanced live and online by moving guests between Hosts. DVX also allows administrators to temporarily (or even indefinitely) increase the CPU reserve allocated to I/O processing on a given Host with Insane Mode. That provides a nearly instantaneous, dynamic way to handle I/O load spikes.

SAN storage administrators must be intimately familiar and experienced with arcane topics like LUNs, queue depths, RAID levels, cache sizing, and step sizes. SANs are also totally dependent on resources within the performance of the active array controller to destage data from NVRAM and manage drive RAID; this is so subtle that some hybrid array vendors have started using cloud-based AI to help make the right recommendations. A SAN array must also compute compression, dedupe, and other data services within the controller. Balancing workloads in a SAN is far more complex than with DVX.

Reclamation and Failures

As guests overwrite locations within a virtual disk or PV, and as guests/vDisks/PVs are created and destroyed, some compressed data blocks will become redundant or obsolete (and unreferenced). To reclaim the space consumed by unreferenced blocks, and deduplicate the Data Node drive pool globally, DVX runs a low priority, distributed Space Reclamation process periodically to find and delete any unreferenced or redundant blocks of data. Space Reclamation also regroups live data for faster upload.

Also, as with any hardware storage system, a drive on a Data Node might fail. Because DVX writes all data with widely-striped, dual parity erasure-coding and hot-sparing, it's able to rebuild the missing data contained on a failed drive and store it in available spare locations. This rebuild process is triggered automatically upon hardware failures.

As always, DVX performs all space reclamation and rebuild tasks using scalable Host resources and Data Node resources. Where a traditional array controller might be severely taxed during "degraded mode" and rebuilds, distributing the work across multiple (and expandable) Hosts is both faster and more scalable. Additional Hosts add workload as well as resources to address it. Additional Data Nodes add protection capacity as well as write bandwidth and recovery/rebuild speed.

An SSD or a Host in its entirety might also fail. DVX ensures that a durable copy of data is available at all times on a Data Node. Host or SSD failures will not cause data loss. That's true for single failures as well as multiple or cascaded Host failures: a durable copy of data always remains available within the Data Node drive pool.

DVX Data Nodes are also able to withstand any individual hardware component failure. If an individual power supply fails, the remaining redundant power supply module provides enough power for the entire system. If a controller (or any sub-component) fails, the standby controller will take over I/O processing.

DVX performs globally applicable tasks like space reclamation and rebuild in a distributed fashion, using resources available throughout the system. The Data Nodes act as coordinators during these system-wide activities.

Guest I/O changes timestamps and capacity usage metadata on each vDisk and PV. The hypervisor may also perform other operations (like adding or deleting guests) that cause file and metadata changes within the NFS datastore presented by DVX.

5 Figure 7 shows how DVX maintains metadata as I/O flows throughout the system.

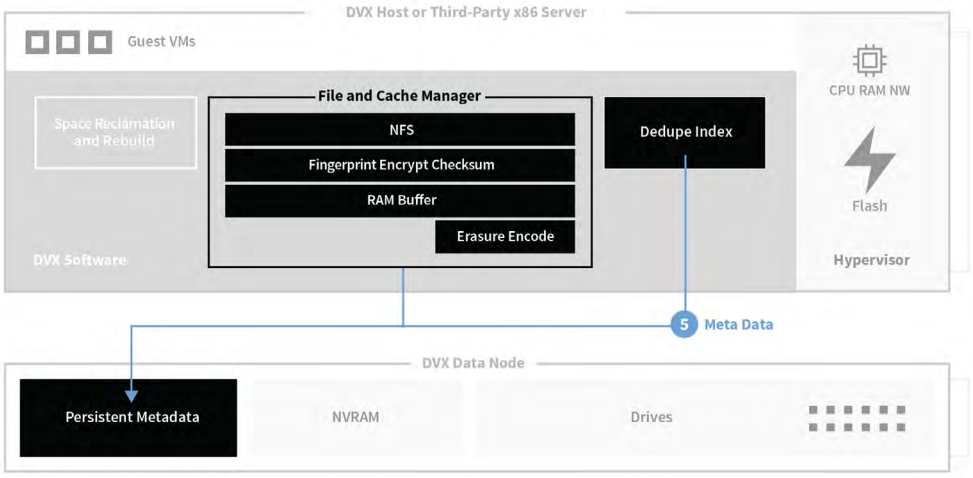

Figure 7 – Overall Data -Flow within DVX, Including Metadata

Filesystem metadata/namespace operations are handled synchronously by the Data Node (though some non-essential metadata is updated asynchronously for performance reasons). These operations are also logged, but unlike guest I/O, they don't consume much bandwidth nor present a significant load to the system in regular operation.

As described previously, DVX also maintains a fingerprint index and mapping information asynchronously, both locally in each Host and globally on the Data Nodes. This information is also a form of metadata.

Conceptually, the filesystem metadata operations aren't dramatically different from a traditional NFS fileserver, so we won't describe them further. The important thing to understand is that the filesystem namespace is managed on the Data Nodes and shared by all Hosts accessing that DVX datastore. *Every Host sees the exact same state of the filesystem (datastore) as a whole*.

Data Protection

DVX provides world-class primary storage for virtualized applications, and it also provides extremely advanced secondary data protection capabilities.

DVX provides efficient, granular, low-impact, and resilient mechanisms to manage point-in-time copies of your data. It allows you to direct the movement of snapshots (data copies) between different DVX systems (over local or wide area networks). DVX helps you achieve the agility you need for backup, disaster recovery, dev/test cloning, and application/content distribution.

Protection Groups

DVX data protection is based on the concept of protection groups. Protection groups apply policies to groups of "live," running applications.

The policies comprise a schedule of when and how frequently snapshots are created, whether or not the snapshots are replicated to another DVX system, precisely which DVX system snapshots are replicated to, and how long the snapshots are retained both locally and at the remote site.

Protection groups apply these policies to one or more VMs or datastore -files (a datastore file may be an individual virtual disk, PV, ISO image, installation/patch file, or powered-off template/OVA/etc). Administrators can modify the collection of VMs and datastore files in a protection group any time.

When creating or editing a protection group, administrators can select individual VMs and datastore files, or they use patterns ("file globs") that are dynamically matched each time snapshots are performed. Dynamic pattern matching is particularly useful when administrators want newly created but similarly named VMs/files to inherit the same protection policies as previous objects (a VM pattern like *db* would match any VM containing the letters "db" in its name).

All VMs and datastore files within a protection group are snapped at the same instant in time: all changes before the instant the snapshot is taken are included in the snapshot, all changes afterward are not. Note that there is no pause in I/O for the running guest VMs – a snapshot is effectively instantaneous across the entire collection of VMs and datastore files. That is, all contents are snapped at the same instant, not sequentially (this is particularly useful with applications that span multiple VMs like common "three-tier" apps).

Unlike traditional LUN-oriented snapshots, protection groups span arbitrary groupings of VMs/files, and it's never necessary to move or copy a VM just to change its protection policy.

Snapstore and Snapshots

DVX maintains two distinct "namespaces" (filesystem metadata about VMs and datastore files).

The "datastore" contains metadata for the current, live version of all VMs and files. That's what the hypervisors "see." The contents of these files always contain the most recently written data.

On the other hand, the "snapstore" contains previous point-in-time snapshots of the live datastore as it existed previously. Every time a protection group causes a snapshot to be taken, new files are created in the snapstore. These read-only files in the snapstore contain the contents of the corresponding files in the live datastore at the instant the snapshot was taken.

DVX uses a "redirect on write" (ROW) technique to store incoming data. New data is always written to new locations (vs. copy-on-write techniques that can introduce delays as changes are copied).

Because only changes are stored in a snapshot, and because DVX only stores compressed and deduplicated data, snapshots consume relatively little space. Of course, as more locations are overwritten with unique data, the more capacity a snapshot consumes.

If, for example, a given VM or file's protection policy causes snapshots to be taken every hour and retained for two days, then the snapstore will contain up to 48 different versions of this VM's files.

The snapstore is only accessible via the DVX GUI. Because the hypervisor only "sees" the live datastore, it's not possible to use the datastore browser on the hypervisor to browse the contents of the snapstore.

The DVX user interface provides two mechanisms to use the snapstore contents: restoring/reverting the contents of VMs and files in the live datastore, and creating net new VMs/files (cloning in the live datastore).

Restoring/reverting replaces the contents of live VMs or datastore files with the state from when the snapshot was taken. It instantly "rolls back" VMs or datastore files to a previous point in time. Because a running VM or container would become confused if the state of a vDisk/PV changed while information is still retained in I/O buffers, only VMs and containers that are currently powered off can be restored.

Cloning is the process of taking a point-in-time snapshot and creating a net new VM or datastore file that's immediately populated with the state contained in the snapshot. Clones are particularly useful for creating dev or test instances from production data. It's an instantaneous way to create one or more copies of existing data.

Elastic Replication

Automatrix Elastic Replication moves snapshots extremely efficiently from a snapstore in one DVX to another DVX.

Elastic Replication is host-based replication, so it's not bound by the performance of a single active controller. As with all DVX operations, it's a scalable, server-powered operation. All reads to support replication are from local flash. In practice, replication creates only a modest additional load on a DVX cluster.

Because each protection group allows you to specify a different destination DVX system (with its own retention policy), it's easy to configure arbitrary topologies: back-to-back replication between two sites, round-robin (site A data replicated to site B, site B data replicated to site C), as well as one-to-many and many-to-one topologies.

Elastic Replication is an incredibly efficient, resilient, and simple to operate data management capability. Only compressed changes are sent over-the-wire, asynchronously, to the remote DVX system. Restores and clones can be performed at either the local or remote site.

Common Questions

Primary storage is understandably a very conservative space. This section attempts to address some of the more common questions and concerns that we've heard from customers, analysts, and competitors.

Is DVX Host Flash a Cache?
Host-side flash in a DVX system can be called a cache because the Hyperdriver uses it for access speed, not durability.

Like traditional caches, DVX Host flash is effectively a "stateless" cache because it can fail in its entirety without losing any data. As noted above, DVX also exhibits familiar caching behavior if flash is undersized.

But DVX Host flash usage differs from traditional RAM or flash caches in several important ways:

1. Because server flash is cheap and abundant, especially without RAID overhead and with compression and dedupe, it holds all data from all VMs, not just a small percentage. If all data is cached, is it still a cache?
2. The flash contents are persistent (remaining valid even after a power loss).
3. The "cache" is not only distributed among multiple servers rather than centralized, but it also services all active I/O locally within each Host (during regular operation). DVX maintains this locality dynamically, even as VMs are vMotioned from one Host to another.
4. DVX quickly repopulates the "cache" on a surviving node after a failure elsewhere, fetching large chunks of surrounding data and not just individual pieces requested.

The result is that most hard-earned knowledge about caching behavior is simply not required with DVX, including concerns about backup or virus scanning causing a "cache blowout," awful when a cache was 10% of protected capacity, aren't relevant when the cache can be as large or even larger than total capacity consumed by each Host.

What is Write Speed and Bandwidth?
Some people become concerned when they realize that data is first written over the network to the NVRAM log and later written again to the Data Node drive pool. At first glance, this "double write" seems excessive.

In practice, however, DVX is quite efficient in terms of network bandwidth. Because log writes and durable writes are both compressed, total write bandwidth demand is similar to a SAN. Note that DVX write bandwidth scales: each Data Node provides NVRAM, disk, and network resources to increase the total write bandwidth linearly.

Further, it's important to realize that any enterprise-class storage system issues multiple I/O requests to the back-end for each individual write request received from the clients (for RAID and mirroring). The slow part is writing safely to durable storage, not traversing the network. DVX allows you to scale this back-end work across multiple Hosts rather than using a single array controller.

What about Host Resource Consumption?
Some HCI storage systems are coy about how many host resources are required. Datrium is up-front about the number DVX requires: up to 20% of cores are reserved by default and up to 40% when Insane Mode is enabled. These values are limits, not permanently allocated reserves.

The practical reality is that almost all hosts have idle CPU cycles going to waste. Several worldwide studies have concluded that the majority of host CPU cycles aren't being used, often with 60% or more CPU headroom remaining throughout the workday. RAM is typically far more constrained than CPU, and DVX is quite efficient with RAM.

Also, while enterprises configure their servers based on their application needs, a VM that is blocked for I/O won't be able to use any CPU cycles. We believe that by allocating up to 20% of scalable host resources to I/O needs, applications perform better overall.

What are the Data Node Bandwidth Constraints?
The last concern is an obvious one: a single Data Node is limited to the bandwidth provided by the 10- or 25-Gigabit Ethernet network interfaces it provides.

Note that each D12X4 or D12X10 Data Node contains just 12 hard disk drives. The bandwidth of two 10-Gigabit Ethernet links in each controller is appropriate for the aggregate write bandwidth of 12 7200 RPM disk drives. Within the constraints of the drives involved, each DVX Data Node provides a balanced system for storing durable data.

Further, the majority of I/O is handled locally within each Host, without ever traversing the network. Eliminating nearly 100% of network-read I/O makes DVX network bandwidth utilization efficient indeed.

Lastly, DVX is a scalable system: read bandwidth scales with the number of Hosts, and write bandwidth scales with the number of Data Nodes. DVX currently supports up to 128 Hosts and up to 120 drives in the Data Pool (up to 10 disk-based Data Nodes, or up to 5 flash-based Data Nodes).

Conclusion

Datrium DVX is a disaggregated HCI (DHCI) solution that enables you to scale I/O performance elastically, independent of your capacity needs. Other DHCI systems put all storage I/O resources in the data pool; DVX uses compute and flash on Hosts to eliminate this bottleneck.

DVX controls the end-to-end data path, and it services I/O from guest applications locally, from flash SSDs internal to the local Host. That's done without sacrificing durability. DVX provides enterprise-class data protection by storing a protected, durable copy of all live data, as well as previous point-in-time snapshots, in a drive pool serviced by one or more Data Nodes.

Because virtually all reads are handled locally, off the network, and writes are compressed before writing to non-volatile memory, I/O performance scales far beyond the “wire speed” of a 10-Gigabit Ethernet network. That's accomplished without sacrificing durability, resilience, server vendor independence, or data management features provided by traditional enterprise arrays.

By separating active data from its durable copy, DVX makes high-performance storage economically feasible, while still providing the resilience and serviceability required by any modern enterprise.

DVX is the leading DHCI solution that allows you to mix and match server configurations while still providing local, off-network, solid-state I/O performance and enterprise-grade resilience and durability. DVX delivers this power with the significant cost advantages of inline dedupe and compression and commodity, server-side flash SSDs.

References

http://blog.fosketts.net/2012/05/24/io-blender-part-2-virtualization

https://www.vmware.com/products/esxi-and-esx/overview

https://www.datrium.com/products/dvx

System Availability: Datrium DHCI vs. Traditional HCI whitepaper

Datrium DRaaS whitepaper

Always-On Erasure Coding whitepaper

Datrium Ransomware Protection whitepaper