FlexCache volumes management

ONTAP 9

NetApp
February 06, 2026

Table of Contents

FlexCache volumes management

Learn about ONTAP FlexCache volumes

NetApp FlexCache technology accelerates data access, reduces WAN latency and lowers WAN bandwidth costs for read-intensive workloads, especially where clients need to access the same data repeatedly. When you create a FlexCache volume, you create a remote cache of an already existing (origin) volume that contains only the actively accessed data (hot data) of the origin volume.

When a FlexCache volume receives a read request of the hot data it contains, it can respond faster than the origin volume because the data does not need to travel as far to reach the client. If a FlexCache volume receives a read request for infrequently read data (cold data), it retrieves the needed data from the origin volume and then stores the data before serving the client request. Subsequent read requests for that data are then served directly from the FlexCache volume. After the first request, the data no longer needs to travel across the network, or be served from a heavily loaded system. For example, suppose you are experiencing bottlenecks within your cluster at a singular access point for frequently requested data. You can use FlexCache volumes within the cluster to provide multiple mount points to the hot data, thereby reducing the bottlenecks and increasing performance. As another example, suppose you need to decrease network traffic to a volume that is accessed from multiple clusters. You can use FlexCache volumes to distribute hot data from the origin volume across the clusters within your network. This reduces WAN traffic by giving users closer access points.

You can also use FlexCache technology to improve performance in cloud and hybrid cloud environments. A FlexCache volume can help you transition workloads to the hybrid cloud by caching data from an on-premises data center to cloud. You can also use FlexCache volumes to remove cloud silos by caching data from one cloud provider to another or between two regions of the same cloud provider.

Beginning with ONTAP 9.10.1, you can enable global file locking across all FlexCache volumes. Global file locking prevents a user from accessing a file that is already opened by another user. Updates to the origin volume are then distributed to all FlexCache volumes simultaneously.

Beginning with ONTAP 9.9.1, FlexCache volumes maintain a list of files not found. This helps reduce network traffic by removing the need to send multiple calls to the origin when clients search for non-existent files.

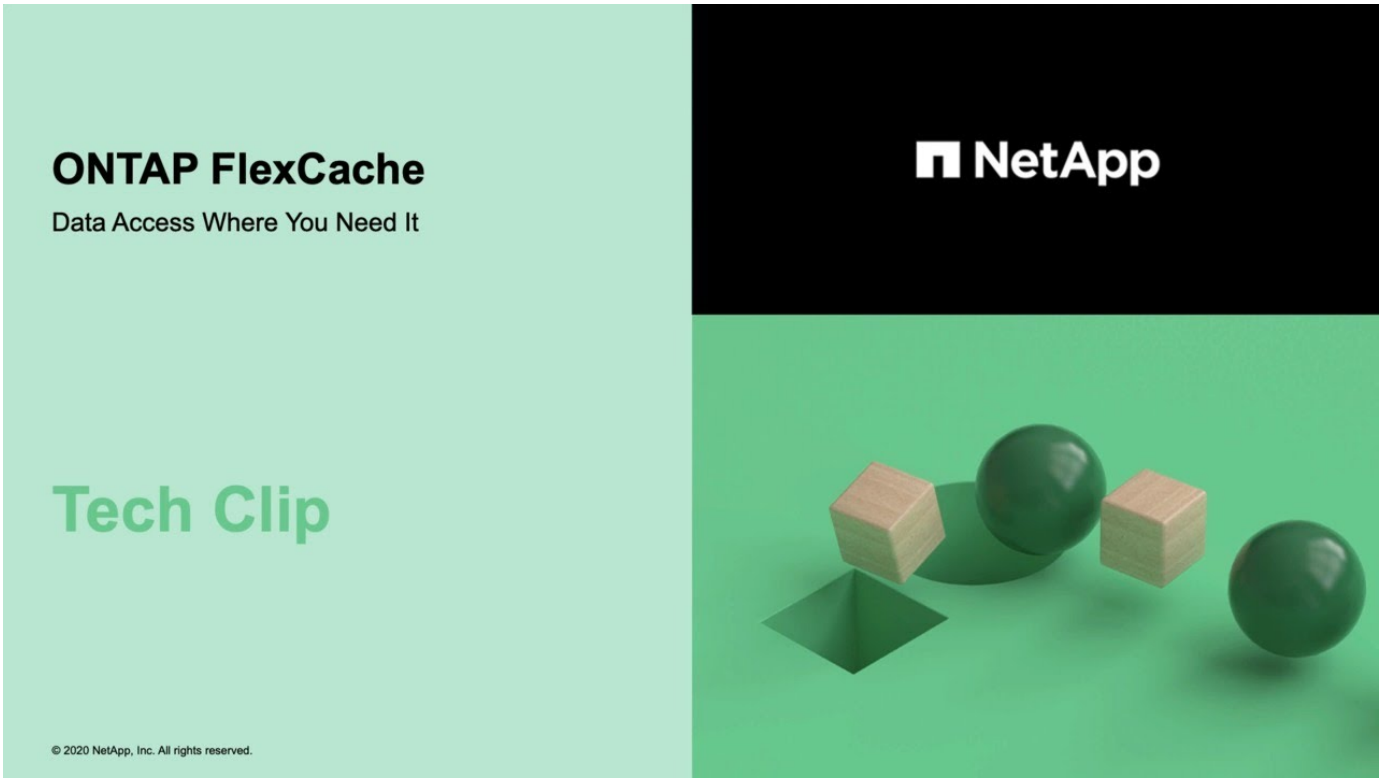
A list of additional features supported for FlexCache volumes and their origin volumes, including a list of supported protocols by ONTAP version, is also available.

You can learn more about the architecture of ONTAP FlexCache technology in TR-4743: FlexCache in ONTAP.

Videos

How FlexCache can reduce WAN latency and read times for global data

Learn about the performance benefits of ONTAP FlexCache!



Supported and unsupported features for ONTAP FlexCache volumes

Beginning with ONTAP 9.5, you can configure FlexCache volumes. FlexVol volumes are

supported as origin volumes and FlexGroup volumes are supported as FlexCache volumes. Beginning with ONTAP 9.7 both FlexVol volumes and FlexGroup volumes are supported as origin volumes. The supported features and protocols for the origin volume and the FlexCache volume vary.

Cache volumes and origin volumes can interoperate as long as both are running on a supported version of ONTAP. Keep in mind that features are supported only when both the cache and the origin are running at least the ONTAP version where support was introduced or a later ONTAP version.

ONTAP version support between FlexCache volumes and origin volumes

The recommended ONTAP version supported between origin volume and the cache volume is no more than four versions earlier or four versions later. For example, if the cache is using ONTAP 9.14.1, the earliest version the origin can be using is ONTAP 9.10.1.

Supported protocols

Protocol	Supported at the origin volume?	Supported at the FlexCache volume?
NFSv3	Yes	Yes
NFSv4	Yes To access cache volumes using NFSv4.x protocol, both the origin and cache clusters must be using ONTAP 9.10.1 or later. The origin cluster and FlexCache cluster can have different ONTAP versions, but both should be ONTAP 9.10.1 and later versions, for example, the origin can have ONTAP 9.10.1, and the cache can have ONTAP 9.11.1.	Yes Supported beginning with ONTAP 9.10.1. To access cache volumes using NFSv4.x protocol, both the origin and cache clusters must be using ONTAP 9.10.1 or later. The origin cluster and FlexCache cluster can have different ONTAP versions, but both should be ONTAP 9.10.1 and later versions, for example, the origin can have ONTAP 9.10.1, and the cache can have ONTAP 9.11.1.
NFSv4.2	Yes	No
SMB	Yes	Yes Supported beginning with ONTAP 9.8.

Supported features

Feature	Supported at the origin volume?	Supported at the FlexCache volume?

Autonomous ransomware protection	Yes Supported for FlexVol origin volumes beginning with ONTAP 9.10.1, and supported for FlexGroup origin volumes beginning with ONTAP 9.13.1. See Autonomous Ransomware Protection use cases and considerations.	No
Antivirus	Yes Supported beginning with ONTAP 9.7.	Not applicable If you configure antivirus scanning at the origin, it is not required on the cache. The origin antivirus scanning detects files infected with viruses before writes are committed, regardless of the write source. For more information about using antivirus scanning with FlexCache, see the FlexCache with ONTAP technical report.
Auditing	Yes Supported beginning with ONTAP 9.7. You can audit NFS file access events in FlexCache relationships using native ONTAP auditing. For more information, see Considerations for auditing FlexCache volumes	Yes Supported beginning with ONTAP 9.7. You can audit NFS file access events in FlexCache relationships using native ONTAP auditing. For more information, see Considerations for auditing FlexCache volumes
Cloud Volumes ONTAP	Yes Supported beginning with ONTAP 9.6	Yes Supported beginning with ONTAP 9.6
Compaction	Yes Supported beginning with ONTAP 9.6	Yes Supported beginning with ONTAP 9.7
Compression	Yes Supported beginning with ONTAP 9.6	Yes Supported beginning with ONTAP 9.6

Deduplication	Yes	Yes Inline deduplication is supported on FlexCache volumes beginning with ONTAP 9.6. Cross-volume deduplication is supported on FlexCache volumes beginning with ONTAP 9.7.
FabricPool	Yes	Yes Supported beginning with ONTAP 9.7 You can create a FlexCache volume as a cache for an origin volume that has FabricPool tiering enabled, but the FlexCache volume itself cannot be tiered.
FlexCache DR	Yes	Yes Supported beginning with ONTAP 9.9.1, with NFSv3 protocol, only. FlexCache volumes must be in separate SVMs or in separate clusters.
FlexGroup volume	Yes Supported beginning with ONTAP 9.7	Yes
FlexVol volume	Yes	No
FPolicy	Yes Supported beginning with ONTAP 9.7	Yes Supported for NFS beginning with ONTAP 9.7. Supported for SMB beginning with ONTAP 9.14.1.
MetroCluster configuration	Yes Supported beginning with ONTAP 9.7	Yes Supported beginning with ONTAP 9.7

Microsoft Offloaded Data Transfer (ODX)	Yes	No
NetApp Aggregate Encryption (NAE)	Yes Supported beginning with ONTAP 9.6	Yes Supported beginning with ONTAP 9.6
NetApp Volume Encryption (NVE)	Yes Supported beginning with ONTAP 9.6	Yes Supported beginning with ONTAP 9.6
ONTAP S3 NAS bucket	Yes Supported beginning with ONTAP 9.12.1	Yes Supported beginning with ONTAP 9.18.1
QoS	Yes	Yes File-level QoS is not supported for FlexCache volumes.
Qtrees	Yes Beginning with ONTAP 9.6, you can create and modify qtrees. Qtrees created on the source can be accessed on the cache.	No
Quotas	Yes Beginning with ONTAP 9.6, quota enforcement on FlexCache origin volumes is supported for users, groups, and qtrees.	No With FlexCache writearound mode (the default mode), writes on the cache are forwarded to the origin volume. Quotas are enforced at the origin. Beginning with ONTAP 9.6, remote quota (rquota) is supported at FlexCache volumes.

SMB Change Notify	Yes	Yes Beginning with ONTAP 9.14.1, SMB Change Notify is supported at the cache.
SnapLock volumes	No	No
SnapMirror asynchronous relationships*	Yes	No
	*FlexCache origins: • You can have a FlexCache volume from an origin FlexVol • You can have a FlexCache volume from an origin FlexGroup • You can have a FlexCache volume from an origin primary volume in SnapMirror relationship. • Beginning with ONTAP 9.8, a SnapMirror secondary volume can be a FlexCache origin volume. The SnapMirror secondary volume must be idle with no active SnapMirror updates; otherwise, FlexCache creation fails.	
SnapMirror synchronous relationships	No	No
SnapRestore	Yes	No
Snapshots	Yes	No
SVM DR configuration	Yes Supported beginning with ONTAP 9.5. The primary SVM of an SVM DR relationship can have the origin volume; however, if you are running an ONTAP release earlier than ONTAP 9.18.1, when the SVM DR relationship is broken, the FlexCache relationship must be re-created with a new origin volume. Beginning with ONTAP 9.18.1, when an origin SVM fails over, caches automatically switch to the origin at the DR site. Manual recovery steps are eliminated. Learn about creating FlexCache volumes.	No You can have FlexCache volumes in primary SVMs, but not in secondary SVMs. Any FlexCache volume in the primary SVM is not replicated as part of the SVM DR relationship.

Storage-level Access Guard (SLAG)	No	No
Thin provisioning	Yes	Yes Supported beginning with ONTAP 9.7
Volume cloning	Yes Cloning of an origin volume and the files in the origin volume is supported beginning with ONTAP 9.6.	No
Volume move	Yes	Yes (only for volume constituents) Moving volume constituents of a FlexCache volume is supported with ONTAP 9.6 and later.
Volume rehost	No	No
vStorage API for Array Integration (VAAI)	Yes	No

In ONTAP 9 releases earlier than 9.5, origin FlexVol volumes can only serve data to FlexCache volumes created on systems running Data ONTAP 8.2.x operating in 7-Mode. Beginning with ONTAP 9.5, origin FlexVol volumes can also serve data to FlexCache volumes on ONTAP 9 systems. For information about migrating from 7-Mode FlexCache to ONTAP 9 FlexCache see NetApp Technical Report 4743: FlexCache in ONTAP.

Guidelines for sizing ONTAP FlexCache volumes

You must be aware of the limits for FlexCache volumes before you start provisioning the volumes.

The size limit of a FlexVol volume is applicable to an origin volume. The size of a FlexCache volume can be less than or equal to the origin volume. The best practice for the size of a FlexCache volume is to be at least 10 percent of the size of the origin volume.

You must also be aware of the following additional limits on FlexCache volumes:

Limit	**ONTAP 9.8 and later**	**ONTAP 9.7**	**ONTAP 9.6 - 9.5**
Maximum number of FlexCache volumes that you can create from an origin volume	100	10	10

Recommended maximum number of origin volumes per node	100	100	10
Recommended maximum number of FlexCache volumes per node	100	100	10
Recommended maximum number of FlexGroup constituents in a FlexCache volume per node	800	800	40
Maximum number of constituents per FlexCache volume per node	32	32	32

Related information

- NetApp Interoperability

Create ONTAP FlexCache volumes

You can create a FlexCache volume in the same ONTAP cluster for improving performance when accessing a hot object. If you have data centers in different locations, you can create FlexCache volumes on remote ONTAP clusters for accelerating data access.

About this task

- Beginning with ONTAP 9.18.1, you can enable NAS S3 bucket access on a FlexCache volume by setting the `-is-s3-enabled` option to `true` when you create the volume. This option is disabled by default.
- Beginning with ONTAP 9.18.1, FlexCache supports creating cache volumes for origin volumes with SVMs that belong to an SVM-DR relationship.

  If you are running ONTAP 9.18.1 or later, a storage administrator must peer the cache SVMs with both the primary and secondary origin SVMs that are part of an SVM-DR relationship before creating cache volumes of origin volumes that are part of SVM-DR relationship.

- Beginning with ONTAP 9.14.0, you can create an unencrypted FlexCache volume from an encrypted source.
- Beginning with ONTAP 9.7, both FlexVol volume and FlexGroup volumes are supported as origin volumes.
- Beginning with ONTAP 9.5, FlexCache supports FlexVol volumes as origin volumes and FlexGroup volumes as FlexCache volumes.

Before you begin

- You must be running ONTAP 9.5 or later.
- If you are running ONTAP 9.6 or earlier, you must add a FlexCache license.

  A FlexCache license is not required for ONTAP 9.7 or later. Beginning with ONTAP 9.7, FlexCache functionality is included with ONTAP and no longer requires a license or activation.

If an HA pair is using encrypting SAS or NVMe drives (SED, NSE, FIPS), you must follow the instructions in the topic Returning a FIPS drive or SED to unprotected mode for all drives within the HA pair prior to initializing the system (boot options 4 or 9). Failure to do this may result in future data loss if the drives are repurposed.

Example 1. Steps

System Manager

1. If the FlexCache volume is on a different ONTAP cluster than the origin volume, create a cluster peer relationship:
   a. In the local cluster, click **Protection > Overview**.
   b. Expand **Intercluster Settings**, click **Add Network Interfaces** and add intercluster network interfaces for the cluster.

      Repeat this step on the remote cluster.

   c. In the remote cluster, click **Protection > Overview**. Click ⋮ in the Cluster Peers section and click **Generate Passphrase**.
   d. Copy the generated passphrase and paste it in the local cluster.
   e. In the local cluster, under Cluster Peers, click **Peer Clusters** and peer the local and remote clusters.
2. Create an SVM peer relationship:

   Under Storage VM Peers, click ⋮ and then **Peer Storage VMs** to peer the storage VMs.
3. Select **Storage > Volumes**.
4. Select **Add**.
5. Select **More Options** and then select **Add as cache for a remote volume**.

   ⓘ If you are running ONTAP 9.8 or later and you want to disable QoS or choose a custom QoS policy, click **More Options**, and then under **Storage and Optimization**, select **Performance Service Level**.

CLI

1. If the FlexCache volume to be created is in a different cluster, create a cluster peer relationship:
   a. On the destination cluster, create a peer relationship with the data protection source cluster:

      ```
      cluster peer create -generate-passphrase -offer-expiration
      MM/DD/YYYY HH:MM:SS|1...7days|1...168hours -peer-addrs
      <peer_LIF_IPs> -initial-allowed-vserver-peers <svm_name>,..|*
      -ipspace <ipspace_name>
      ```

      Beginning with ONTAP 9.6, TLS encryption is enabled by default when creating a cluster peer relationship. TLS encryption is supported for the intercluster communication between the origin and FlexCache volumes. You can also disable TLS encryption for the cluster peer relationship, if required.

```
cluster02::> cluster peer create -generate-passphrase -offer
-expiration 2days -initial-allowed-vserver-peers *

                    Passphrase: UCa+6lRVICXeL/gq1WrK7ShR
               Expiration Time: 6/7/2017 08:16:10 EST
  Initial Allowed Vserver Peers: *
            Intercluster LIF IP: 192.140.112.101
              Peer Cluster Name: Clus_7ShR (temporary generated)

Warning: make a note of the passphrase - it cannot be displayed
again.
```

b. On the source cluster, authenticate the source cluster to the destination cluster:

```
cluster peer create -peer-addrs <peer_LIF_IPs> -ipspace <ipspace>
```

```
cluster01::> cluster peer create -peer-addrs
192.140.112.101,192.140.112.102

Notice: Use a generated passphrase or choose a passphrase of 8 or
more characters.
        To ensure the authenticity of the peering relationship,
use a phrase or sequence of characters that would be hard to
guess.

Enter the passphrase:
Confirm the passphrase:

Clusters cluster02 and cluster01 are peered.
```

2. If the FlexCache volume is in a different SVM than that of the origin volume, create an SVM peer relationship with `flexcache` as the application:

a. If the SVM is in a different cluster, create an SVM permission for the peering SVMs:

```
vserver peer permission create -peer-cluster <cluster_name>
-vserver <svm-name> -applications flexcache
```

The following example illustrates how to create an SVM peer permission that applies for all of the local SVMs:

```
cluster1::> vserver peer permission create -peer-cluster cluster2
-vserver "*" -applications flexcache

Warning: This Vserver peer permission applies to all local
Vservers. After that no explict
"vserver peer accept" command required for Vserver peer
relationship creation request
from peer cluster "cluster2" with any of the local Vservers. Do
you want to continue? {y|n}: y
```

b. Create the SVM peer relationship:

```
vserver peer create -vserver <local_SVM> -peer-vserver
<remote_SVM> -peer-cluster <cluster_name> -applications flexcache
```

3. Create a FlexCache volume:

```
volume flexcache create -vserver <cache_svm> -volume
<cache_vol_name> -auto-provision-as flexgroup -size <vol_size>
-origin-vserver <origin_svm> -origin-volume <origin_vol_name> -is-s3
-enabled true|false
```

The following example creates a FlexCache volume and automatically selects existing aggregates for provisioning:

```
cluster1::> volume flexcache create -vserver vs_1 -volume fc1 -auto
-provision-as flexgroup -origin-volume vol_1 -size 160MB -origin
-vserver vs_1
[Job 443] Job succeeded: Successful
```

The following example creates a FlexCache volume and sets the junction path:

```
cluster1::> volume flexcache create -vserver vs34 -volume fc4 -aggr
-list aggr34,aggr43 -origin-volume origin1 -size 400m -junction-path
/fc4
[Job 903] Job succeeded: Successful
```

The following example enables S3 access on a FlexCache volume:

```
cluster1::> volume flexcache create -vserver vs3 -volume
cache_vs3_vol33 -origin-volume vol33 -origin-vserver vs3 -junction
-path /cache_vs3_vol33 -is-s3-enabled true
```

4. Verify the FlexCache relationship from the FlexCache volume and the origin volume.
   a. View the FlexCache relationship in the cluster:

```
volume flexcache show
```

```
cluster1::> volume flexcache show
Vserver Volume      Size       Origin-Vserver Origin-Volume
Origin-Cluster
------- ----------- ---------- -------------- -------------
--------------
vs_1    fc1         160MB      vs_1           vol_1
cluster1
```

   b. View all of the FlexCache relationships in the origin cluster:
   `volume flexcache origin show-caches`

```
cluster::> volume flexcache origin show-caches
Origin-Vserver Origin-Volume   Cache-Vserver    Cache-Volume
Cache-Cluster
-------------- --------------- ---------------  --------------
---------------
vs0            ovol1           vs1              cfg1
clusA
vs0            ovol1           vs2              cfg2
clusB
vs_1           vol_1           vs_1             fc1
cluster1
```

Result

The FlexCache volume is successfully created. Clients can mount the volume by using the junction path of the FlexCache volume.

Related information

Cluster and SVM peering

FlexCache write-back

Learn about ONTAP FlexCache write-back

Introduced in ONTAP 9.15.1, FlexCache write-back is an alternate mode of operation for writing at a cache. Write-back allows the write to be committed to stable storage at the cache and acknowledged to the client without waiting for the data to make it to the origin. The data is asynchronously flushed back to the origin. The result is a globally distributed file system that enables writes to perform at near-local speeds for specific workloads and environments, offering significant performance benefits.

ONTAP 9.12.1 introduced a write-back feature as a public preview. This is referred to as write-back version 1 (wbv1) and shouldn't be thought of as the same as write-back in ONTAP 9.15.1, which is referred to as write-back version 2 (wbv2).

Write-back vs write-around

Since FlexCache was introduced in ONTAP 9.5, it has been a read-writable cache; however, it operated in write-around mode. Writes at the cache were shipped to the origin to be committed to stable storage. After the origin successfully committed the write to stable storage, it acknowledged the write to the cache. The cache would then acknowledge the write to the client. This made every write incur the penalty of traversing the network between the cache and origin. FlexCache write-back changes this.

After upgrading to ONTAP 9.15.1, you can convert a traditional write-around cache to a write-back cache, and, if necessary, back to write-around. This can, however, make reading diagnostic logs harder should a problem arise.

	Write-around	Write-back
ONTAP Version	9.6+	9.15.1+
Use case	Read-heavy workload	Write-heavy workload
Data committed at	Origin	Cache
Client experience	WAN-like	LAN-like
Limits	100 per origin	10 per origin
CAP Theorem	Available and tolerant to partition	Available and consistent

FlexCache write-back terminology

Understand key concepts and terms working with FlexCache write-back.

Term	Definition
Dirty data	Data that has been committed to stable storage at the cache, but has not been flushed to the origin.

Term	Definition
Exclusive Lock Delegation (XLD)	A protocol-level lock authority granted on a per-file basis to a cache. This authority allows the cache to hand out exclusive write locks to clients without contacting the origin.
Shared Lock Delegation (SLD)	A protocol-level lock authority granted on a per-file basis to a cache. This authority allows the cache to hand out shared read locks to clients without contacting the origin.
Write-back	A mode of FlexCache operation where writes to a cache are committed to stable storage at that cache and immediately acknowledged to the client. Data is asynchronously written back to the origin.
Write-around	A mode of FlexCache operation where writes to a cache are forwarded to the origin to be committed to stable storage. Once committed, the origin will acknowledge the write to the cache, and the cache will acknowledge the write to the client.
Dirty Data Record System (DDRS)	A proprietary mechanism that keeps track of the dirty data in a write-back-enabled cache on a per-file basis.
Origin	A FlexGroup or FlexVol that contains the source data for all FlexCache cache volumes. It is the single source of truth, orchestrates locking, and ensures 100% data consistency, currency, and coherency.
Cache	A FlexGroup that is a sparse cache volume of the FlexCache origin.

Consistent, current, and coherent

FlexCache is NetApp's solution to having the right data, everywhere, every time. FlexCache is 100% consistent, current, and coherent 100% of the time:

- **Consistent:** The data is the same wherever it is accessed.
- **Current:** The data is always up-to-date.
- **Coherent:** The data is correct/uncorrupted.

ONTAP FlexCache write-back guidelines

FlexCache write-back involves many complex interactions between the origin and caches. For optimal performance, you should ensure your environment follows these guidelines. These guidelines are based on the latest major ONTAP version (ONTAP 9.17.1.) available at the time of content creation.

As a best practice, test your production workload in a non-production environment. This is even more important if you are implementing FlexCache write-back outside of these guidelines.

The following guidelines are well-tested internally at NetApp. It is ***strongly*** recommended you stay within them. If you do not, unexpected behavior could occur.

- Significant enhancements for FlexCache write-back were introduced in ONTAP 9.17.1P1. It is ***strongly*** advised you run the current recommended release after 9.17.1P1 at both the origin and cache clusters. If

you are unable to run 9.17.1 codeline, the latest P release of 9.16.1 is the next suggested release. ONTAP 9.15.1 does not have all the necessary fixes and improvements for FlexCache write-back, and is not recommended for production workloads.

- In its current iteration, FlexCache write-back caches should be configured with a single constituent for the entire FlexCache volume. Multi-constituent FlexCaches can result in unwanted evictions of data from the cache.
- Testing has been executed for files smaller than 100GB and WAN round-trip times between the cache and origin not exceeding 200ms. Any workloads outside of these limits might result in unexpected performance characteristics.
- Writing to SMB alternate data streams causes the main file to be evicted from the cache. All dirty data for the main file needs to be flushed to the origin before any other operations can take place on that file. The alternate data stream is also forwarded to the origin.
- Renaming a file causes the file to be evicted from the cache. All dirty data for the file needs to be flushed to the origin before any other operations can take place on that file.
- At this time, the only attributes that can be changed or set on a file on the write-back-enabled FlexCache volume are:
  - Timestamps
  - Mode bits
  - NT ACLs
  - Owner
  - Group
  - Size

    Any other attributes that are changed or set are forwarded to origin which might result in evicting the file from the cache. If you require other attributes to be changed or set at the cache, ask your account team to open a PVR.
- Snapshots taken at the origin cause recalling all outstanding dirty data from every write-back-enabled cache associated with that origin volume. This might require multiple retries of the operation if there is significant write-back activity in progress, as evicts of those dirty files might take some time.
- SMB Opportunistic Locks (Oplocks) for writes are not supported on write-back-enabled FlexCache volumes.
- The origin must remain under 80% full. Cache volumes are not granted exclusive lock delegations if there isn’t at least 20% space remaining in the origin volume. Calls to a write-back-enabled cache are forwarded to the origin in this situation. This helps prevent running out of space at the origin, which would result in leaving dirty data orphaned at a write-back-enabled cache.
- Low bandwidth and/or lossy intercluster networks can have a significant negative effect on FlexCache write-back performance. While there isn’t a specific bandwidth requirement, as it is highly dependent on your workload, it is ***strongly*** recommended you ensure the health of the intercluster link between the cache(s) and origin.

ONTAP FlexCache write-back architecture

FlexCache was designed with strong consistency in mind, including both modes of write operation: write-back and write-around. Both the traditional write-around mode of operation and the new write-back mode of operation introduced in ONTAP 9.15.1 guarantee that the data accessed will always be 100% consistent, current, and coherent.

The following concepts detail how FlexCache write-back operates.

Delegations

Lock delegations and data delegations helps FlexCache keep both write-back and write-around caches data consistent, coherent, and current. The origin orchestrates both delegations.

Lock delegations

A lock delegation is a protocol-level lock authority the origin grants on a per-file basis to a cache to issue protocol locks to clients as needed. These include exclusive lock delegations (XLD) and shared lock delegations (SLD).

XLD and write-back

To ensure ONTAP never has to reconcile a conflicting write, an XLD is granted to a cache where a client requests to write to a file. Importantly, only one XLD can exist for any file at any time, meaning there never will be more than one writer to a file at a time.

When the request to write to a file comes into a write-back enabled cache, the following steps take place:

1. The cache checks if it already has an XLD for the requested file. If so, it will grant the write lock to the client as long as another client isn't writing to the file at the cache. If the cache doesn't have an XLD for the requested file, it will request one from the origin. This is a proprietary call that traverses the intercluster network.
2. Upon receiving the XLD request from the cache, the origin will check if there is an outstanding XLD for the file at another cache. If so, it will recall that file's XLD, which triggers a flush of any dirty data from that cache back to the origin.
3. Once the dirty data from that cache is flushed back and committed to stable storage at the origin, the origin will grant the XLD for the file to the requesting cache.
4. Once the file's XLD is received, the cache grants the lock to the client, and the write commences.

A high-level sequence diagram covering some of these steps is covered in the Write-back sequence diagram.

From a client perspective, all locking will work as if it were writing to a standard FlexVol or FlexGroup with a potential small delay when the write lock is requested.

In it's current iteration, if a write-back enabled cache holds the XLD for a file, ONTAP will block **any** access to that file at other caches, including `READ` operations.

There is a limit of 170 XLDs per origin constituent.

Data delegations

A data delegation is a per-file guarantee given to a cache by the origin that the data cached for that file is up-to-date. As long as the cache has a data delegation for a file, it can serve the cached data for that file to the client without having to contact the origin. If the cache doesn't have a data delegation for the file, it must contact the origin to receive the data requested by the client.

In write-back mode, a file's data delegation is revoked if an XLD is taken for that file at another cache or the origin. This effectively fences off the file from clients at all other caches and the origin, even for reads. This is a trade off that must be made to ensure old data is never accessed.

Reads at a write-back-enabled cache generally operate like reads at a write-around cache. In both write-

around and write-back-enabled caches, there could be an initial `READ` performance hit when the requested file has an exclusive write lock at a write-back-enabled cache other than where the read is issued. The XLD has to be revoked, and the dirty data must be committed to the origin before the read at the other cache can be serviced.

Tracking dirty data

Write-back from cache to origin happens asynchronously. This means that dirty data isn't immediately written back to the origin. ONTAP employs a dirty data record system to keep track of dirty data per file. Each dirty data record (DDR) represents approximately 20MB of dirty data for a particular file. When a file is actively being written, ONTAP will start flushing dirty data back after two DDRs have been filled and the third DDR is being written. This results in approximately 40MB of dirty data remaining in a cache during writes. For stateful protocols (NFSv4.x, SMB), the remaining 40MB of data will be flushed back to the origin when the file is closed. For stateless protocols (NFSv3), the 40MB of data will be flushed back when either access to the file is requested at a different cache or after the file is idle for two or more minutes, up to a maximum of five minutes. For more information on timer-triggered or space-triggered dirty data flushing, see Cache scrubbers.

In addition to the DDRs and scrubbers, some front-end NAS operations also trigger the flushing of all dirty data for a file:

- `SETATTR`
  - `SETATTR`s that modify only mtime, atime, and/or ctime can be processed at the cache, avoiding the penalty of the WAN.
- `CLOSE`
- `OPEN` at another cache
- `READ` at another cache
- `READDIR` at another cache
- `READDIRPLUS` at another cache
- `WRITE` at another cache

Disconnected mode

When an XLD for a file is held at a write-around cache and that cache gets disconnected from the origin, reads for that file are still allowed at the other caches and origin. This behavior differs when an XLD is held by a write-back-enabled cache. In this case, if the cache is disconnected, reads to the file will hang everywhere. This helps ensure 100% consistency, currency, and coherence are maintained. The reads are allowed in write-around mode because the origin is guaranteed to have all of the data available that has been write-acknowledged to the client. In write-back mode during a disconnect, the origin can not guarantee that all of the data written to and acknowledged by the write-back-enabled cache made it to the origin before the disconnect occurred.

In the event a cache with an XLD for a file is disconnected for an extended period of time, a system administrator can manually revoke the XLD at the origin. This will allow IO to the file to resume at the surviving caches and the origin.

Manually revoking the XLD will result in the loss of any dirty data for the file at the disconnected cache. Manually revoking an XLD should only be done in the event of a catastrophic disruption between the cache and origin.

Cache scrubbers

There are scrubbers in ONTAP that run in response to specific events, such as a timer expiring or space thresholds being breached. The scrubbers take an exclusive lock on the file being scrubbed, effectively freezing IO to that file until the scrub completes.

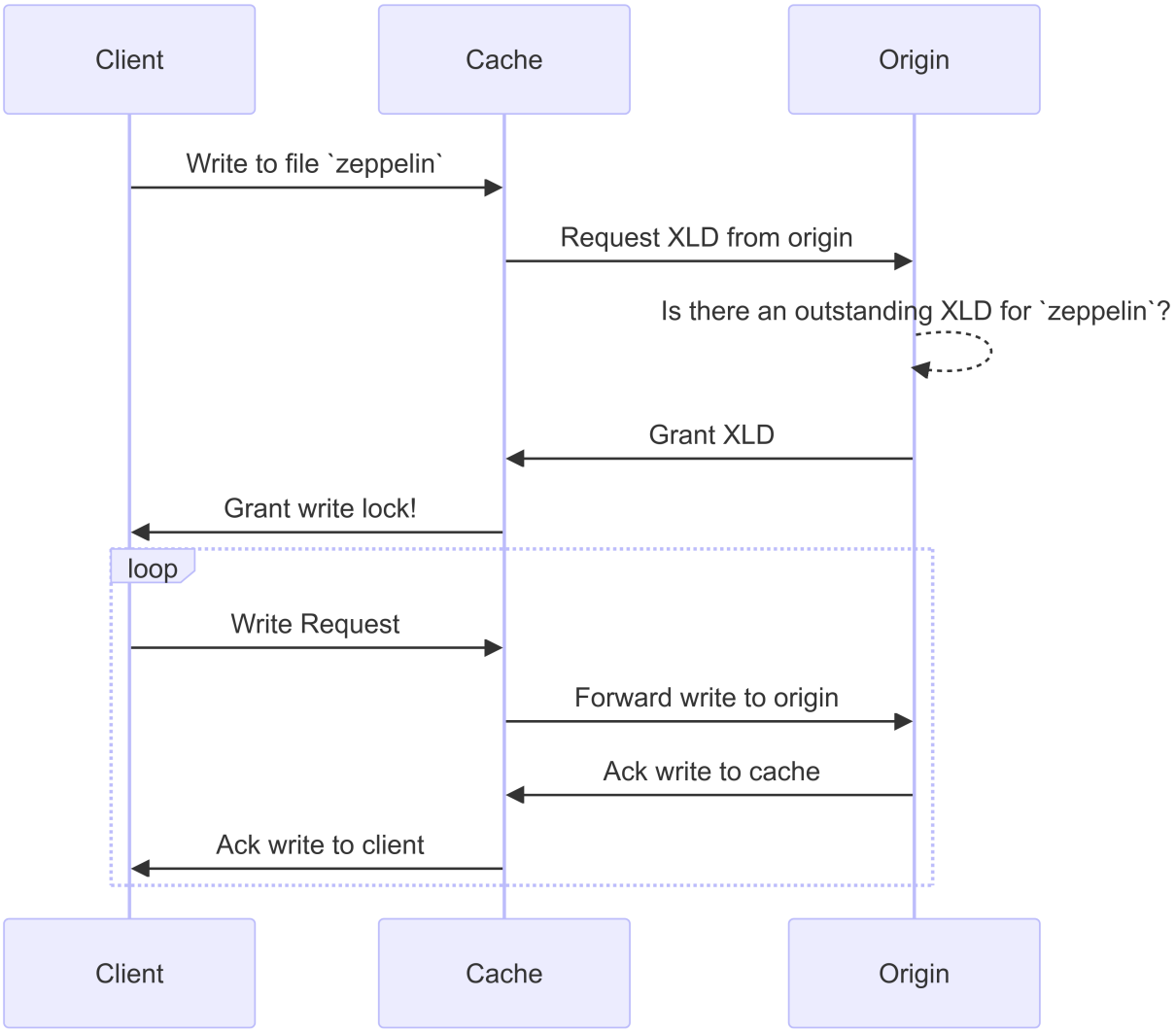
Scrubbers include:

- **mtime-based scrubber on the cache:** This scrubber starts every five minutes and scrubs any file sitting unmodified for two minutes. If any dirty data for the file is still in the cache, IO to that file is quiesced and write-back is triggered. IO will resume after the write-back is complete.
- **mtime-based scrubber on origin:** Much like the mtime-based scrubber at the cache, this also runs every five minutes. However, it scrubs any file sitting unmodified for 15 minutes, recalling the inode's delegation. This scrubber doesn't initiate any write-back.
- **RW limit-based scrubber on origin:** ONTAP monitors how many RW lock delegations are handed out per origin constituent. If this number surpasses 170, ONTAP starts scrubbing write lock delegations on a least-recently-used (LRU) basis.
- **Space-based scrubber on the cache:** If a FlexCache volume reaches 90% full, the cache is scrubbed, evicting on an LRU basis.
- **Space-based scrubber on the origin:** If a FlexCache origin volume reaches 90% full, the cache is scrubbed, evicting on an LRU basis.

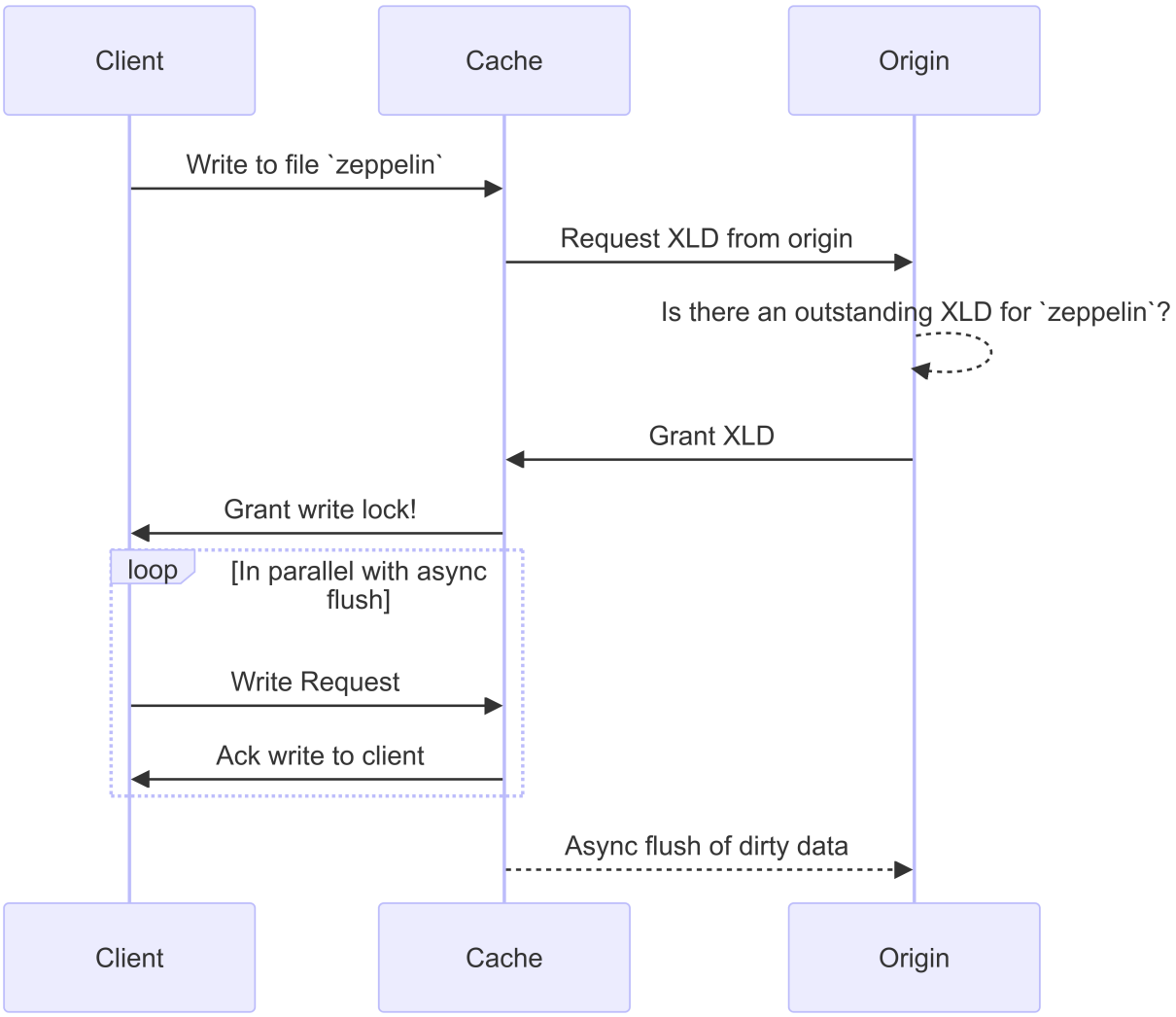
Sequence diagrams

These sequence diagrams depict the difference in write acknowledgements between write-around and write-back mode.

Write-around


Client
Cache
Origin
Write to file `zeppelin`
Request XLD from origin
Is there an outstanding XLD for `zeppelin`?
Grant XLD
Grant write lock!
loop
Write Request
Forward write to origin
Ack write to cache
Ack write to client
Client
Cache
Origin

Write-back



ONTAP FlexCache write-back use cases

These are write profiles best suited for a write-back-enabled FlexCache. You should test your workload to see if write-back or write-around provides the best performance.

Write-back is not a replacement for write-around. Although write-back is designed with write-heavy workloads, write-around is still the better choice for many workloads.

Target workloads

File size

File size is less important than the number of writes issued between the `OPEN` and `CLOSE` calls for a file. Small files inherently have fewer `WRITE` calls, making them less ideal for write-back. Large files might have more writes between `OPEN` and `CLOSE` calls, but this isn't guaranteed.

Refer to the FlexCache write-back guidelines page for the most current recommendations regarding max file size.

Write size

When writing from a client, other modifying NAS calls are involved other than write calls. These include, but are

not limited to:

- `CREATE`
- `OPEN`
- `CLOSE`
- `SETATTR`
- `SET_INFO`

`SETATTR` and `SET_INFO` calls that set `mtime`, `atime`, `ctime`, `owner`, `group`, or `size` are processed at the cache. The rest of these calls must be processed at the origin and trigger a write-back of any dirty data accumulated at the write-back-enabled cache for the file being operated on. IO to the file will be quiesced until the write-back is complete.

Knowing that these calls must traverse the WAN helps you to identify workloads suited for write-back. Generally, the more writes that can be done between `OPEN` and `CLOSE` calls without one of the other calls listed above being issued, the better the performance gain write-back provides.

Read-after-write

Read-after-write workloads have historically performed poorly at FlexCache. This is due to the write-around mode of operation before 9.15.1. The `WRITE` call to the file has to be committed at the origin, and the subsequent `READ` call would have to pull the data back to the cache. This results in both operations incurring the penalty of the WAN. Therefore, read-after-write workloads are discouraged for FlexCache in write-around mode. With the introduction of write-back in 9.15.1, data is now committed at the cache, and can immediately be read from the cache, eliminating the WAN penalty. If your workload includes read-after-write at FlexCache volumes, you should configure the cache to operate in write-back mode.

If read-after-write is a critical part of your workload, you should configure your cache to operate in write-back mode.

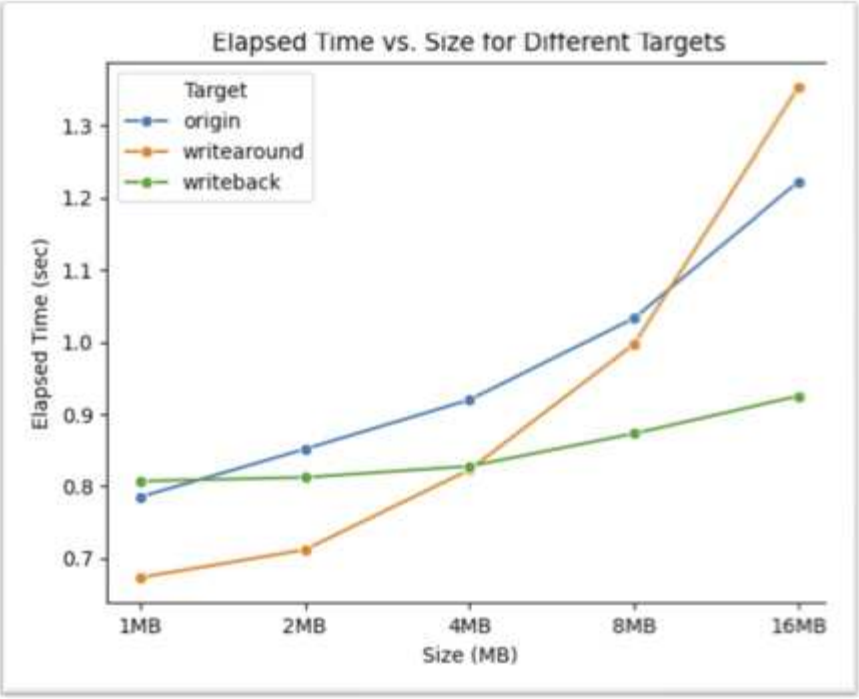
Write-after-write

When a file accumulates dirty data in a cache, the cache asynchronously writes the data back to the origin. This naturally leads to times when the client closes the file with dirty data still waiting to be flushed back to origin. If another open or write comes in for the file that was just closed and still has dirty data, the write will be suspended until all the dirty data has been flushed to origin.

Latency considerations

When FlexCache operates in write-back mode, it becomes more beneficial to NAS clients as latency increases. There is a point, however, at which the overhead of write-back outweighs the advantages gained in low-latency environments. In some NetApp tests, write-back benefits started around a minimum latency between cache and origin of 8ms. This latency varies with workload, so be sure to test to know your workload's point-of-return.

The following graph shows the point-of-return for write-back in NetApp lab tests. The `x` axis is the file-size, and the `y` axis is the elapsed time. The test used NFSv3, mounting with an `rsize` and `wsize` of 256KB, and 64ms of WAN latency. This test was performed using a small ONTAP Select instance for both the cache and origin, and a single threaded-write operation. Your results might vary.



Write-back should not be used for intracluster caching. Intracluster caching occurs when the origin and cache are in the same cluster.

ONTAP FlexCache write-back prerequisites

Before you deploy FlexCache in write-back mode, ensure you have met these performance, software, licensing, and system configuration requirements.

CPU and Memory

It is ***strongly recommended*** that each origin cluster node have at least 128GB of RAM and 20 CPUs to absorb the write-back messages initiated by write-back enabled caches. This is the equivalent of an A400 or greater. If the origin cluster serves as the origin to multiple write-back enabled FlexCaches, it will require more CPU and RAM.

Using an undersized origin for your workload can have profound impacts on performance at the write-back-enabled cache or the origin.

ONTAP version

- The origin ***must*** be running ONTAP 9.15.1 or later.
- Any caching cluster that needs to operate in write-back mode ***must*** be running ONTAP 9.15.1 or later.
- Any caching cluster that does not need to operate in write-back mode can run any generally supported ONTAP version.

Licensing

FlexCache, including the write-back mode of operation, is included with your ONTAP purchase. No extra license is required.

Peering

- The origin and cache clusters must be cluster peered
- The server virtual machines (SVMs) on the origin and cache cluster must be vserver peered with the FlexCache option.

> You do not need to peer a cache cluster to another cache cluster. There is also no need to peer a cache SVM to another cache SVM.

ONTAP FlexCache write-back interoperability

Understand these interoperability considerations when deploying FlexCache in write-back mode.

ONTAP version

To use the write-back mode of operation, both the cache and origin **must** be running ONTAP 9.15.1 or later.

> Clusters where a write-back-enabled cache is unnecessary can run earlier versions of ONTAP, but that cluster can only operate in write-around mode.

You can have a mix of ONTAP versions in your environment.

Table 1. Mixed cluster versions example 1

Cluster	ONTAP version	Write-back supported?
Origin	ONTAP 9.15.1	N/A †
Cluster 1	ONTAP 9.15.1	Yes
Cluster 2	ONTAP 9.14.1	No

Table 2. Mixed cluster versions example 2

Cluster	ONTAP version	Write-back supported?
Origin	ONTAP 9.14.1	N/A †
Cluster 1	ONTAP 9.15.1	No
Cluster 2	ONTAP 9.15.1	No

† *Origins aren't a cache, so neither write-back nor write-around support is applicable.*

> In Mixed cluster versions example 2, neither cluster can enable write-back mode because the origin is not running ONTAP 9.15.1 or later, which is a strict requirement.

Client interoperability

Any client generally supported by ONTAP can access a FlexCache volume regardless of whether it is operating in write-around or write-back mode. For an up-to-date list of supported clients, refer to NetApp's interoperability matrix.

Although the client version doesn't matter specifically, the client must be new enough to support NFSv3,

NFSv4.0, NFSv4.1, SMB2.x, or SMB3.x. SMB1 and NFSv2 are deprecated protocols and are not supported.

Write-back and write-around

As seen in Mixed cluster versions example 1, FlexCache operating in write-back mode can co-exist with caches operating in write-around mode. It is advised to compare write-around against write-back with your specific workload.

If the performance for a workload is the same between write-back and write-around, use write-around.

ONTAP feature interoperability

For the most up-to-date list of FlexCache feature interoperability, refer to the supported and unsupported features for FlexCache volumes.

Enable and manage ONTAP FlexCache write-back

Beginning with ONTAP 9.15.1, you can enable FlexCache write-back mode on FlexCache volumes to provide better performance for edge computing environments and caches with write-heavy workloads. You can also determine whether write-back is enabled on a FlexCache volume or disable write-back on the volume when necessary.

When write-back is enabled on the cache volume, write requests are sent to the local cache rather than to the origin volume.

Before you begin

You must be in advanced privilege mode.

Create a new FlexCache volume with write-back enabled

Steps

You can create a new FlexCache volume with write-back enabled by using ONTAP System Manager or the ONTAP CLI.

System Manager

1. If the FlexCache volume is on a different cluster than the origin volume, create a cluster peer relationship:
   a. On the local cluster, click **Protection > Overview**.
   b. Expand **Intercluster Settings**, click **Add Network Interfaces**, and add intercluster interfaces to the cluster.

      Repeat this on the remote cluster.

   c. On the remote cluster, click **Protection > Overview**. Click ⋮ in the Cluster Peers section and click **Generate Passphrase**.
   d. Copy the generated passphrase and paste it in the local cluster.
   e. On the local cluster, under Cluster Peers, click **Peer Clusters** and peer the local and remote clusters.
2. If the FlexCache volume is on a different cluster than the origin volume, create an SVM peer relationship:

   Under **Storage VM Peers**, click ⋮ and then **Peer Storage VMs** to peer the storage VMs.

   If the FlexCache volume is on the same cluster, you cannot create an SVM peer relationship using System Manager.

3. Select **Storage > Volumes**.
4. Select **Add**.
5. Select **More Options** and then select **Add as cache for a remote volume**.
6. Select **Enable FlexCache write-back**.

CLI

1. If the FlexCache volume to be created is in a different cluster, create a cluster peer relationship:
   a. On the destination cluster, create a peer relationship with the data protection source cluster:

      ```
      cluster peer create -generate-passphrase -offer-expiration
      MM/DD/YYYY HH:MM:SS|1...7days|1...168hours -peer-addrs
      <peer_LIF_IPs> -initial-allowed-vserver-peers <svm_name>,..|*
      -ipspace <ipspace_name>
      ```

      Beginning with ONTAP 9.6, TLS encryption is enabled by default when creating a cluster peer relationship. TLS encryption is supported for the intercluster communication between the origin and FlexCache volumes. You can also disable TLS encryption for the cluster peer relationship, if required.

```
cluster02::> cluster peer create -generate-passphrase -offer
-expiration 2days -initial-allowed-vserver-peers *

                    Passphrase: UCa+6lRVICXeL/gq1WrK7ShR
               Expiration Time: 6/7/2017 08:16:10 EST
  Initial Allowed Vserver Peers: *
            Intercluster LIF IP: 192.140.112.101
              Peer Cluster Name: Clus_7ShR (temporary generated)

Warning: make a note of the passphrase - it cannot be displayed
again.
```

b. On the source cluster, authenticate the source cluster to the destination cluster:

```
cluster peer create -peer-addrs <peer_LIF_IPs> -ipspace <ipspace>
```

```
cluster01::> cluster peer create -peer-addrs
192.140.112.101,192.140.112.102

Notice: Use a generated passphrase or choose a passphrase of 8 or
more characters.
        To ensure the authenticity of the peering relationship,
use a phrase or sequence of characters that would be hard to
guess.

Enter the passphrase:
Confirm the passphrase:

Clusters cluster02 and cluster01 are peered.
```

2. If the FlexCache volume is in a different SVM than that of the origin volume, create an SVM peer relationship with `flexcache` as the application:

a. If the SVM is in a different cluster, create an SVM permission for the peering SVMs:

```
vserver peer permission create -peer-cluster <cluster_name>
-vserver <svm-name> -applications flexcache
```

The following example illustrates how to create an SVM peer permission that applies for all of the local SVMs:

```
cluster1::> vserver peer permission create -peer-cluster cluster2
-vserver "*" -applications flexcache

Warning: This Vserver peer permission applies to all local
Vservers. After that no explict
"vserver peer accept" command required for Vserver peer
relationship creation request
from peer cluster "cluster2" with any of the local Vservers. Do
you want to continue? {y|n}: y
```

b. Create the SVM peer relationship:

```
vserver peer create -vserver <local_SVM> -peer-vserver
<remote_SVM> -peer-cluster <cluster_name> -applications flexcache
```

3. Create a FlexCache volume with write-back enabled:

```
volume flexcache create -vserver <cache_vserver_name> -volume
<cache_flexgroup_name> -aggr-list <list_of_aggregates> -origin
-volume <origin flexgroup> -origin-vserver <origin_vserver name>
-junction-path <junction_path> -is-writeback-enabled true
```

Enable FlexCache write-back on an existing FlexCache volume

You can enable FlexCache write-back on an existing FlexCache volume using ONTAP System Manager or the ONTAP CLI.

System Manager

1. Select **Storage > Volumes** and select an existing FlexCache volume.
2. On the volume's Overview page, click **Edit** in the upper right corner.
3. In the **Edit Volume** window, select **Enable FlexCache write-back**.

CLI

1. Enable write-back on an existing FlexCache volume:

```
volume flexcache config modify -volume <cache_flexgroup_name> -is
-writeback-enabled true
```

Check if FlexCache write-back is enabled

Steps

You can use System Manager or the ONTAP CLI to determine whether FlexCache write-back is enabled.

System Manager

1. Select **Storage > Volumes** and select a volume.
2. In the volume **Overview**, locate **FlexCache details** and check if FlexCache write-back is set to **Enabled** on the FlexCache volume.

CLI

1. Check if FlexCache write-back is enabled:

```
volume flexcache config show -volume <cache_flexgroup_name> -fields
is-writeback-enabled
```

Disable write-back on a FlexCache volume

Before you can delete a FlexCache volume you need to disable FlexCache write-back.

Steps

You can use System Manager or the ONTAP CLI to disable FlexCache write-back.

System Manager

1. Select **Storage > Volumes** and select an existing FlexCache volume that has FlexCache write-back enabled.
2. On the volume's Overview page, click **Edit** in the upper right corner.
3. In the **Edit Volume** window, deselect **Enable FlexCache write-back**.

CLI

1. Disable write-back:

```
volume flexcache config modify -volume <cache_vol_name> -is
-writeback-enabled false
```

Frequently asked questions about ONTAP FlexCache write-back

This FAQ can help if you are looking for a quick answer to a question.

I want to use write-back. What version of ONTAP do I need to run?

Both the cache and the origin must be running ONTAP 9.15.1 or later. It is ***strongly*** recommended that you run the latest P release. Engineering is constantly improving the performance and functionality of write-back-enabled caches.

Can clients accessing the origin have an effect on clients accessing the write-back-enabled cache?

Yes. The origin has equal right to the data as any of the caches. If an operation is executed on a file that requires the file to be evicted from the cache, or a lock/data delegation to be revoked, the client at the cache might see a delay accessing the file.

Can I apply QoS to write-back-enabled FlexCaches?

Yes. Every cache and the origin can have independent QoS policies applied. This will have no direct effect on any write-back initiated intercluster traffic. Indirectly, you can slow down intercluster write-back traffic by QoS limiting the front-end traffic at the write-back-enabled cache.

Is multi-protocol NAS supported at write-back-enabled FlexCaches?

Yes. Multi-protocol is fully supported at write-back-enabled FlexCaches. Currently, NFSv4.2 and S3 are not supported by FlexCache operating in write-around or write-back mode.

Are SMB alternate data streams supported at write-back-enabled FlexCaches?

SMB alternate data streams (ADS) are supported, but not accelerated by write-back. The write to the ADS is forwarded to the origin, incurring the penalty of the WAN latency. The write also evicts the main file the ADS is a part of from the cache.

Can I switch a cache between write-around and write-back mode after it is created?

Yes. All you have to do is toggle the `is-writeback-enabled` flag in the `flexcache modify` command.

Are there bandwidth considerations I should be aware of for the intercluster link between the cache(s) and origin?

Yes. FlexCache write-back is highly dependent on the intercluster link between the cache(s) and origin. Low bandwidth and/or lossy networks can have a significant negative effect on performance. There isn't a specific bandwidth requirement, as it is highly dependent on your workload.

FlexCache duality

FAQ about FlexCache duality

This FAQ answers common questions about FlexCache duality introduced in ONTAP 9.18.1.

Frequently asked questions

What is "duality?"

Duality enables unified access to the same data using both file (NAS) and object (S3) protocols. Introduced in ONTAP 9.12.1 without FlexCache support, duality was extended in ONTAP 9.18.1 to include FlexCache volumes, allowing S3 protocol access to NAS files cached in a FlexCache volume.

What S3 operations are supported on a FlexCache S3 bucket?

S3 operations supported on standard S3 NAS buckets are supported on FlexCache S3 NAS buckets, with the exception of the `COPY` operation. For an up-to-date list of unsupported operations for a standard S3 NAS bucket, visit the interoperability documentation.

Can I use FlexCache in write-back mode with FlexCache duality?

No. If a FlexCache S3 NAS bucket is created on a FlexCache volume, the FlexCache volume **must** be in write-around mode. If you attempt to create a FlexCache S3 NAS bucket on a FlexCache volume in write-back mode, the operation will fail.

I can't upgrade one of my clusters to ONTAP 9.18.1 because of hardware limitations. Will duality still work in my cluster if only the cache cluster is running ONTAP 9.18.1?

No. Both the cache cluster and origin cluster must have a minimum effective cluster version of 9.18.1. If you attempt to create a FlexCache S3 NAS bucket on a cache cluster peered with an origin running an ONTAP version earlier than 9.18.1, the operation will fail.

I have a MetroCluster configuration. Can I use FlexCache duality?

No. FlexCache duality is not supported in MetroCluster configurations.

Can I audit S3 access to files in a FlexCache S3 NAS bucket?

S3 auditing is provided by the NAS auditing functionality FlexCache volumes use. For more information about NAS auditing of FlexCache volumes, see Learn more about FlexCache auditing.

What should I expect if the cache cluster becomes disconnected from the origin cluster?

S3 requests to a FlexCache S3 NAS bucket will fail with a `503 Service Unavailable` error if the cache cluster is disconnected from the origin cluster.

Can I use multipart S3 operations with FlexCache duality?

For multipart S3 operations to work, the underlying FlexCache volume must have the granular-data field set to 'advanced'. This field is set to whatever value is set for the origin volume.

Does FlexCache duality support HTTP and HTTPS access?

Yes. By default, HTTPS is required. You can configure the S3 service to allow HTTP access if needed.

Enable S3 access to NAS FlexCache volumes

Beginning in ONTAP 9.18.1, you can enable S3 access to NAS FlexCache volumes, also referred to as "duality." This allows clients to access data stored in a FlexCache volume using the S3 protocol, in addition to traditional NAS protocols like NFS and SMB. You can use the following information to set up FlexCache duality.

Prerequisites

Before you begin, you must ensure you complete the following prerequisites:

- Make sure the S3 protocol and desired NAS protocols (NFS, SMB, or both) are licensed and configured on the SVM.
- Verify that DNS and any other required services are configured.
- Cluster and SVM Peered
- FlexCache Volume create
- Data-lif created

For more thorough documentation on FlexCache duality, see ONTAP S3 multiprotocol support.

Step 1: Create and sign certificates

To enable S3 access to a FlexCache volume, you need to install certificates for the SVM that hosts the FlexCache volume. This example uses self-signed certificates, but in a production environment, you should use certificates signed by a trusted Certificate Authority (CA).

1. Create an SVM root CA:

```
security certificate create -vserver <svm> -type root-ca -common-name
<arbitrary_name>
```

2. Generate a Certificate Signing Request:

```
security certificate generate-csr -common-name <dns_name_of_data_lif>
-dns-name <dns_name_of_data_lif> -ipaddr <data_lif_ip>
```

Example output:

```
-----BEGIN CERTIFICATE REQUEST-----
MIICzjCCAbYCAQAwHzEdMBsGA1UEAxMUY2FjaGUxZy1kYXRhLm5hcy5sYWIwggEi
MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCusJk075O8Uh329cHI6x+BaRS2
w5wrqvzoYlidXtYmdCH3m1DDprBiAyfIwBC0/iU3Xd5NpB7nc1wK1CI2VEkrXGUg
...
vMIGN351+FgzLQ4X5lKfoMXCV70NqIakxzEmkTIUDKv7n9EVZ4b5DTTlrL03X/nK
+Bim2y2y180PaFB3NauZHTnIIzIc8zCp2IEqmFWyMDcdBjP9KS0+jNm4QhuXiM8F
D7gm3g/O70qa5OxbAEal5o4NbOl95U0T0rwqTaSzFG0XQnK2PmA1OIwS5ET35p3Z
dLU=
-----END CERTIFICATE REQUEST-----
```

Private key example:

```
-----BEGIN PRIVATE KEY-----
MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQCusJk075O8Uh32
9cHI6x+BaRS2w5wrqvzoYlidXtYmdCH3m1DDprBiAyfIwBC0/iU3Xd5NpB7nc1wK
1CI2VEkrXGUgwBtx1K4IlrCTB829Q1aLGAQXVyWnzhQc4tS5PW/DsQ8t7olZ9zEI
...
rXGEdDaqp7jQGNXUGlbxO3zcBil1/A9Hc6oalNECgYBKwe3PeZamiwhIHLy9ph7w
dJfFCshsPalMuAp2OuKIAnNa9l6fT9y5kf9tIbskT+t5Dth8bmV9pwe8UZaK5eC4
Svxm19jHT5QqloDaZVUmMXFKyKoqPDdfvcDk2Eb5gMfIIb0a3TPC/jqqpDn9BzuH
TO02fuRvRR/G/HUz2yRd+A==
-----END PRIVATE KEY-----
```

Keep a copy of your certificate request and private key for future reference.

3. Sign the certificate:

The `root-ca` is the one you created in Create an SVM root CA.

```
certificate sign -ca <svm_root_ca> -ca-serial <svm_root_ca_sn> -expire
-days 364 -format PEM -vserver <svm>
```

4. Paste the Certificate Signing Request (CSR) generated in Generate a Certificate Signing Request.

   Example:

```
-----BEGIN CERTIFICATE REQUEST-----
MIICzjCCAbYCAQAwHzEdMBsGA1UEAxMUY2FjaGUxZy1kYXRhLm5hcy5sYWIwggEi
MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCusJk075O8Uh329cHI6x+BaRS2
w5wrqvzoYlidXtYmdCH3m1DDprBiAyfIwBC0/iU3Xd5NpB7nc1wK1CI2VEkrXGUg
...
vMIGN351+FgzLQ4X5lKfoMXCV70NqIakxzEmkTIUDKv7n9EVZ4b5DTTlrL03X/nK
+Bim2y2y180PaFB3NauZHTnIIzIc8zCp2IEqmFWyMDcdBjP9KS0+jNm4QhuXiM8F
D7gm3g/O70qa5OxbAEal5o4NbOl95U0T0rwqTaSzFG0XQnK2PmA1OIwS5ET35p3Z
dLU=
-----END CERTIFICATE REQUEST-----
```

   This prints a signed certificate to the console, similar to the following example.

   Signed Certificate example:

```
-----BEGIN CERTIFICATE-----
MIIDdzCCAl+gAwIBAgIIGHolbgv5DPowDQYJKoZIhvcNAQELBQAwLjEfMB0GA1UE
AxMWY2FjaGUtMTY0Zy1zdm0tcm9vdC1jYTELMAkGA1UEBhMCVVMwHhcNMjUxMTIx
MjIxNTU4WhcNMjYxMTIwMjIxNTU4WjAfMR0wGwYDVQQDExRjYWNoZTFnLWRhdGEu
...
qS7zhj3ikWE3Gp9s+QijKWXx/0HDd1UuGqy0QZNqNm/M0mqVnokJNk5F4fBFxMiR
1o63BxL8xGIRdtTCjjb2Gq2Wj7EClUw6CykEkxAcVk+XrRtArGkNtcYdtHfUsKVE
wswvv0rNydrNnWhJLhSl8TW5Tex+OMyTXgk9/3K8kB0mAMrtxxYjt8tm+gztkivf
J0eo1uDJhaNxqwEZRzFyGaa4k1+56oFzRfTc
-----END CERTIFICATE-----
```

5. Copy the certificate for the next step.
6. Install the server certificate on the SVM:

```
certificate install -type server -vserver <svm> -cert-name flexcache-
duality
```

7. Paste the signed certificate from Sign the certificate.

   Example:

```
Please enter Certificate: Press <Enter> [twice] when done
-----BEGIN CERTIFICATE-----
MIIDdzCCAl+gAwIBAgIIGHolbgv5DPowDQYJKoZIhvcNAQELBQAwLjEfMB0GA1UE
AxMWY2FjaGUtMTY0Zy1zdm0tcm9vdC1jYTELMAkGA1UEBhMCVVMwHhcNMjUxMTIx
MjIxNTU4WhcNMjYxMTIwMjIxNTU4WjAfMR0wGwYDVQQDExRjYWNoZTFnLWRhdGEu
bmFzLmxhYjCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAK6wmTTvk7xS
...
qS7zhj3ikWE3Gp9s+QijKWXx/0HDd1UuGqy0QZNqNm/M0mqVnokJNk5F4fBFxMiR
1o63BxL8xGIRdtTCjjb2Gq2Wj7EClUw6CykEkxAcVk+XrRtArGkNtcYdtHfUsKVE
wswvv0rNydrNnWhJLhSl8TW5Tex+OMyTXgk9/3K8kB0mAMrtxxYjt8tm+gztkivf
J0eo1uDJhaNxqwEZRzFyGaa4k1+56oFzRfTc
-----END CERTIFICATE-----
```

8. Paste the private key generated in Generate a Certificate Signing Request.

   Example:

```
Please enter Private Key: Press <Enter> [twice] when done
-----BEGIN PRIVATE KEY-----
MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQCusJk075O8Uh32
9cHI6x+BaRS2w5wrqvzoYlidXtYmdCH3m1DDprBiAyfIwBC0/iU3Xd5NpB7nc1wK
1CI2VEkrXGUgwBtx1K4IlrCTB829Q1aLGAQXVyWnzhQc4tS5PW/DsQ8t7olZ9zEI
W/gaEIajgpXIwGNWZ+weKQK+yoolxC+gy4IUE7WvnEUiezaIdoqzyPhYq5GC4XWf
0johpQugOPe0/w2nVFRWJoFQp3ZP3NZAXc8H0qkRB6SjaM243XV2jnuEzX2joXvT
wHHH+IBAQ2JDs7s1TY0I20e49J2Fx2+HvUxDx4BHao7CCHA1+MnmEl+9E38wTaEk
NLsU724ZAgMBAAECggEABHUy06wxcIk5hO3S9Ik1FDZV3JWzsu5gGdLSQOHRd5W+
...
rXGEdDaqp7jQGNXUGlbxO3zcBil1/A9Hc6oalNECgYBKwe3PeZamiwhIHLy9ph7w
dJfFCshsPalMuAp2OuKIAnNa9l6fT9y5kf9tIbskT+t5Dth8bmV9pwe8UZaK5eC4
Svxm19jHT5QqloDaZVUmMXFKyKoqPDdfvcDk2Eb5gMfIIb0a3TPC/jqqpDn9BzuH
TO02fuRvRR/G/HUz2yRd+A==
-----END PRIVATE KEY-----
```

9. Enter certificates of certification authorities (CA) which form the certificate chain of the server certificate.

   This starts with the issuing CA certificate of the server certificate and can range up to the root CA certificate.

```
Do you want to continue entering root and/or intermediate certificates
{y|n}: n

You should keep a copy of the private key and the CA-signed digital
certificate for future reference.

The installed certificate's CA and serial number for reference:
CA: cache-164g-svm-root-ca
serial: 187A256E0BF90CFA
```

10. Get the public key for the SVM root CA:

```
security certificate show -vserver <svm> -common-name <root_ca_cn> -ca
<root_ca_cn> -type root-ca -instance

-----BEGIN CERTIFICATE-----
MIIDgTCCAmmgAwIBAgIIGHokTnbsHKEwDQYJKoZIhvcNAQELBQAwLjEfMB0GA1UE
AxMWY2FjaGUtMTY0Zy1zdm0tcm9vdC1jYTELMAkGA1UEBhMCVVMwHhcNMjUxMTIx
MjE1NTIzWhcNMjYxMTIxMjE1NTIzWjAuMR8wHQYDVQQDExZjYWNoZS0xNjRnLXN2
bS1yb290LWNhMQswCQYDVQQGEwJVUzCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
...
DoOL7vZFFt44xd+rp0DwafhSnLH5HNhdIAfa2JvZW+eJ7rgevH9wmOzyc1vaihl3
Ewtb6cz1a/mtESSYRNBmGkIGM/SFCy5v1ROZXCzF96XPbYQN4cW0AYI3AHYBZP0A
HlNzDR8iml4k9IuKf6BHLFA+VwLTJJZKrdf5Jvjgh0trGAbQGI/Hp2Bjuiopkui+
n4aa5Rz0JFQopqQddAYnMuvcq10CyNn7S0vF/XLd3fJaprH8kQ==
-----END CERTIFICATE-----
```

This is needed to configure the client to trust the certificates signed by the SVM root-ca. The public key is printed to the console. Copy and save the public key. The values in this command are the same ones you entered in Create an SVM root CA.

Step 2: Configure the S3 server

1. Enable S3 protocol access:

```
vserver show -vserver <svm> -fields allowed-protocols
```

S3 is allowed at the SVM level by default.

2. Clone an existing policy:

```
network interface service-policy clone -vserver <svm> -policy default-
data-files -target-vserver <svm> -target-policy <any_name>
```

3. Add S3 to the cloned policy:

```
network interface service-policy add-service -vserver <svm> -policy
<any_name> -service data-s3-server
```

4. Add the new policy to the data lif:

```
network interface modify -vserver <svm> -lif <data_lif> -service-policy
duality
```

> Modifying the service policy of an existing LIF can be disruptive. It requires the LIF to be taken down and brought back up with a listener for the new service. TCP **should** recover from this quickly, but be aware of potential impact.

5. Create the S3 object store server on the SVM:

```
vserver object-store-server create -vserver <svm> -object-store-server
<dns_name_of_data_lif> -certificate-name flexcache-duality
```

6. Enable S3 capability on FlexCache volume:

   The `flexcache config` option `-is-s3-enabled` must be set to `true` before you can create a bucket. You must also set the option `-is-writeback-enabled` to `false`.

   The following command modifies an existing FlexCache:

```
flexcache config modify -vserver <svm> -volume <fcache_vol> -is
-writeback-enabled false -is-s3-enabled true
```

7. Create an S3 bucket:

```
vserver object-store-server bucket create -vserver <svm> -bucket
<bucket_name> -type nas -nas-path <flexcache_junction_path>
```

8. Create a bucket policy:

```
vserver object-store-server bucket policy add-statement -vserver <svm>
-bucket <bucket_name> -effect allow
```

9. Create an S3 user:

```
vserver object-store-server user create -user <user> -comment ""
```

Example output:

```
    Vserver: <svm>>
       User: <user>>
 Access Key: WCOT7...Y7D6U
 Secret Key: 6143s...pd__P
    Warning: The secret key won't be displayed again. Save this key for
future use.
```

10. Regenerate keys for the root user:

```
vserver object-store-server user regenerate-keys -vserver <svm> -user
root
```

Example output:

```
    Vserver: <svm>>
       User: root
 Access Key: US791...2F1RB
 Secret Key: tgYmn...8_3o2
    Warning: The secret key won't be displayed again. Save this key for
future use.
```

Step 3: Set up the client

There are many S3 clients available. A good place to start is with the AWS CLI. For more information, see Installing the AWS CLI.

Manage FlexCache volumes

Learn about auditing ONTAP FlexCache volumes

Beginning with ONTAP 9.7, you can audit NFS file access events in FlexCache relationships using native ONTAP auditing and file policy management with FPolicy.

Beginning with ONTAP 9.14.1, FPolicy is supported for FlexCache volumes with NFS or SMB. Previously, FPolicy was not supported for FlexCache volumes with SMB.

Native auditing and FPolicy are configured and managed with the same CLI commands used for FlexVol volumes. However, there is some different behavior with FlexCache volumes.

- **Native auditing**
  - You can't use a FlexCache volume as the destination for audit logs.
  - If you want to audit read and writes on FlexCache volumes, you must configure auditing on both the cache SVM as well as on the origin SVM.

    This is because file system operations are audited where they are processed. That is, reads are audited on the cache SVM and writes are audited on the origin SVM.

  - To track the origin of write operations, the SVM UUID and MSID are appended in the audit log to identify the FlexCache volume from which the write originated.
- **FPolicy**
  - Although writes to a FlexCache volume are committed on the origin volume, FPolicy configurations monitor the writes on the cache volume. This is unlike native auditing, in which the writes are audited on the origin volume.
  - While ONTAP does not require the same FPolicy configuration on cache and origin SVMs, it is recommended that you deploy two similar configurations. You can do so by creating a new FPolicy policy for the cache, configured like that of the origin SVM but with the scope of the new policy limited to the cache SVM.
  - The size of extensions in an FPolicy configuration is limited to 20KB (20480 bytes). When the size of extensions used in an FPolicy configuration on a FlexCache volume exceeds 20KB, the EMS message `nblade.fpolicy.extn.failed` is triggered.

Synchronize properties of an ONTAP FlexCache volume from an origin volume

Some of the volume properties of the FlexCache volume must always be synchronized with those of the origin volume. If the volume properties of a FlexCache volume fail to synchronize automatically after the properties are modified at the origin volume, you can manually synchronize the properties.

About this task

The following volume properties of a FlexCache volume must always be synchronized with those of the origin volume:

- Security style (`-security-style`)
- Volume name (`-volume-name`)
- Maximum directory size (`-maxdir-size`)
- Minimum read ahead (`-min-readahead`)

Step

1. From the FlexCache volume, synchronize the volume properties:

```
volume flexcache sync-properties -vserver svm_name -volume flexcache_volume
```

```
cluster1::> volume flexcache sync-properties -vserver vs1 -volume fc1
```

Update the configuration of ONTAP FlexCache relationships

After events such as volume move, aggregate relocation, or storage failover, the volume configuration information on the origin volume and FlexCache volume is updated automatically. In case the automatic updates fail, an EMS message is generated and then you must manually update the configuration for the FlexCache relationship.

If the origin volume and the FlexCache volume are in the disconnected mode, you might need to perform some additional operations to update a FlexCache relationship manually.

About this task

If you want to update the configurations of a FlexCache volume, you must run the command from the origin volume. If you want to update the configurations of an origin volume, you must run the command from the FlexCache volume.

Step

1. Update the configuration of the FlexCache relationship:

   ```
   volume flexcache config-refresh -peer-vserver peer_svm -peer-volume
   peer_volume_to_update -peer-endpoint-type [origin | cache]
   ```

Enable file access time updates on the ONTAP FlexCache volume

Beginning with ONTAP 9.11.1, you can enable the `-atime-update` field on the FlexCache volume to permit file access time updates. You can also set an access time update period with the `-atime-update-period` attribute. The `-atime-update-period` attribute controls how often access time updates can take place and when they can propagate to the origin volume.

Overview

ONTAP provides a volume-level field called `-atime-update`, to manage access time updates on files and directories that are read using READ, READLINK, and READDIR. Atime is used for data lifecycle decisions for files and directories that are infrequently accessed. The infrequently accessed files are eventually migrated to archive storage and are often later moved to tape.

The atime-update field is disabled by default on existing and newly created FlexCache volumes. If you are using FlexCache volumes with ONTAP releases earlier than 9.11.1, you should leave the atime-update field disabled so caches aren't unnecessarily evicted when a read operation is performed on the origin volume. With large FlexCache caches, however, administrators use special tools to manage data and help to ensure that hot data remains in the cache and cold data is purged. This is not possible when atime-update is disabled. However, beginning with ONTAP 9.11.1, you can enable `-atime-update` and `-atime-update-period`, and use the tools required to manage the cached data.

Before you begin

- All FlexCache volumes must be running ONTAP 9.11.1 or later.
- You must use the `advanced` privilege mode.

About this task

Setting `-atime-update-period` to 86400 seconds allows no more than one access time update per 24-hour period, regardless of the number of read-like operations performed on a file.

Setting the `-atime-update-period` to 0 sends messages to the origin for each read access. The origin then informs each FlexCache volume that the atime is outdated, which impacts performance.

Steps

1. Set the privilege mode to `advanced`:

   ```
   set -privilege advanced
   ```

2. Enable file access time updates and set the update frequency:

   ```
   volume modify -volume vol_name -vserver <SVM name> -atime-update true -atime
   -update-period <seconds>
   ```

   The following example enables `-atime-update` and sets `-atime-update-period` to 86400 seconds, or 24 hours:

   ```
   c1: volume modify -volume origin1 vs1_c1 -atime-update true -atime
   -update-period 86400
   ```

3. Verify that `-atime-update` is enabled:

   ```
   volume show -volume vol_name -fields atime-update,atime-update-period
   ```

   ```
   c1::*> volume show -volume cache1_origin1 -fields atime-update,atime-
   update-period
   vserver volume         atime-update atime-update-period
   ------- -------------- ------------ -------------------
   vs2_c1  cache1_origin1 true         86400
   ```

4. After `-atime-update` is enabled, you can specify if the files on a FlexCache volume can be scrubbed automatically and a scrubbing interval:

   ```
   volume flexcache config modify -vserver <SVM name> -volume <volume_name> -is
   -atime-scrub-enabled <true|false> -atime-scrub-period <integer>
   ```

Learn more about `-is-atime-scrub-enabled` parameter in the ONTAP command reference.

Enable global file locking on ONTAP FlexCache volumes

Beginning with ONTAP 9.10.1, global file locking can be applied to prevent reads across all related cached files.

With global file locking enabled, modifications to the origin volume are suspended until all FlexCache volumes are online. You should only enable global file locking when you have control over the reliability of the connections between cache and origin due to suspension and possible timeouts of modifications when FlexCache volumes are offline.

Before you begin

- Global file locking requires the clusters containing the origin and all associated caches to be running ONTAP 9.9.1 or later. Global file locking can be enabled on new or existing FlexCache volumes. The command can be run on one volume and applies to all associated FlexCache volumes.
- You must be in the advanced privilege level to enable global file locking.
- If you revert to a version of ONTAP earlier than 9.9.1, global file locking must first be disabled on the origin and associated caches. To disable, from the origin volume, run: `volume flexcache prepare-to-downgrade -disable-feature-set 9.10.0`
- The process to enable global file locking depends on whether the origin has existing caches:
  - Enable global file locking on new FlexCache volumes
  - Enable global file locking on existing FlexCache volumes

Enable global file locking on new FlexCache volumes

Steps

1. Create the FlexCache volume with `-is-global-file-locking` set to true:

   `volume flexcache create volume` *`volume_name`* `-is-global-file-locking-enabled true`

   The default value of `-is-global-file-locking` is “false”. When any subsequent `volume flexcache create` commands are run on a volume, they must be passed with `-is-global-file-locking enabled` set to “true”.

Enable global file locking on existing FlexCache volumes

Steps

1. Global file locking must be set from the origin volume.
2. The origin cannot have any other existing relationships (for example, SnapMirror). Any existing relationships must be dissociated. All caches and volumes must be connected at the time of running the command. To check the connection status, run:

   `volume flexcache connection-status show`

   The status for all the listed volumes should display as `connected`. For more information, see View the status of a FlexCache relationship or Synchronize properties of a FlexCache volume from an origin.

3. Enable global file locking on the caches:

```
volume flexcache origin config show/modify -volume volume_name -is-global-file
-locking-enabled true
```

Related information

- ONTAP command reference

Prepopulate ONTAP FlexCache volumes

You can prepopulate a FlexCache volume to reduce the time it takes to access cached data.

Before you begin

- You must be a cluster administrator at the advanced privilege level
- The paths you pass for prepopulation must exist or the prepopulate operation fails.

About this task

- Prepopulate reads files only and crawls through directories
- The `-isRecursion` flag applies to the entire list of directories passed to prepopulate

Steps

1. Prepopulate a FlexCache volume:

   ```
   volume flexcache prepopulate -cache-vserver vserver_name -cache-volume -path
   -list path_list -isRecursion true|false
   ```

   - The `-path-list` parameter indicates the relative directory path you want to prepopulate starting from the origin root directory. For example, if the origin root directory is named /origin and it contains directories /origin/dir1 and /origin/dir2, you can specify the path list as follows: `-path-list dir1, dir2` or `-path-list /dir1, /dir2`.
   - The default value of the `-isRecursion` parameter is True.

   This example prepopulates a single directory path:

   ```
   cluster1::*> flexcache prepopulate start -cache-vserver vs2 -cache
   -volume fg_cachevol_1 -path-list /dir1
     (volume flexcache prepopulate start)
   [JobId 207]: FlexCache prepopulate job queued.
   ```

   This example prepopulates files from several directories:

   ```
   cluster1::*> flexcache prepopulate start -cache-vserver vs2 -cache
   -volume fg_cachevol_1 -path-list /dir1,/dir2,/dir3,/dir4
     (volume flexcache prepopulate start)
   [JobId 208]: FlexCache prepopulate job queued.
   ```

   This example prepopulates a single file:

```
cluster1::*> flexcache prepopulate start -cache-vserver vs2 -cache
-volume fg_cachevol_1 -path-list /dir1/file1.txt
  (volume flexcache prepopulate start)
[JobId 209]: FlexCache prepopulate job queued.
```

This example prepopulates all files from the origin:

```
cluster1::*> flexcache prepopulate start -cache-vserver vs2 -cache
-volume fg_cachevol_1 -path-list / -isRecursion true
  (volume flexcache prepopulate start)
[JobId 210]: FlexCache prepopulate job queued.
```

This example includes an invalid path for prepopulation:

```
cluster1::*> flexcache prepopulate start -cache-volume
vol_cache2_vs3_c2_vol_origin1_vs1_c1 -cache-vserver vs3_c2 -path-list
/dir1, dir5, dir6
  (volume flexcache prepopulate start)

Error: command failed: Path(s) "dir5, dir6" does not exist in origin
volume
       "vol_origin1_vs1_c1" in Vserver "vs1_c1".
```

2. Display the number of files read:

   `job show -id job_ID -ins`

Related information

- job show

Delete ONTAP FlexCache relationships

You can delete a FlexCache relationship and the FlexCache volume if you no longer require the FlexCache volume.

Steps

1. From the cluster that has the FlexCache volume, take the FlexCache volume offline:

   `volume offline -vserver svm_name -volume volume_name`

2. Delete the FlexCache volume:

   `volume flexcache delete -vserver svm_name -volume volume_name`

   The FlexCache relationship details are removed from the origin volume and the FlexCache volume.

FlexCache for hotspot remediation

Remediating hotspotting in high-performance compute workloads with ONTAP FlexCache volumes

A common problem with many high-performance compute workloads, such as animation rendering or EDA, is hotspotting. Hotspotting is a situation that occurs when a specific part of the cluster or network experiences a significantly higher load compared to other areas, leading to performance bottlenecks and reduced overall efficiency due to excessive data traffic concentrated in that location. For example, a file, or multiple files, is in high demand for the job running which results in a bottleneck at the CPU used to service requests (via a volume affinity) to that file. FlexCache can help alleviate this bottleneck, but it must be set up properly.

This documentation explains how to set up FlexCache to remediate hotspotting.

> Beginning July 2024, content from technical reports previously published as PDFs has been integrated with ONTAP product documentation. This ONTAP hotspot remediation technical report content is net new as of the date of its publication and no earlier format was ever produced.

Key concepts

When planning hotspot remediation, it's important to understand these essential concepts.

- **High-density FlexCache (HDF)**: A FlexCache that is condensed to span as few nodes as the cache capacity requirements allow
- **HDF Array (HDFA)**: A group of HDFs that are caches of the same origin, distributed across the cluster
- **Inter-SVM HDFA**: One HDF from the HDFA per server virtual machine (SVM)
- **Intra-SVM HDFA**: All HDFs in the HDFA in one SVM
- **East-west traffic**: Cluster backend traffic generated from indirect data access

What's next

- Understand how to architect with high-density FlexCache to help remediate hotspotting
- Decide on FlexCache array density
- Determine the density of your HDFs and decide whether you will be accessing the HDFs using NFS with inter-SVM HDFAs and intra-SVM HDFAs
- Configure HDFA and the data LIFs to realize the benefits of using intracluster caching with ONTAP configuration
- Learn how to configure clients to distribute ONTAP NAS connections with client configuration

Architecting an ONTAP FlexCache hotspot remediation solution

To remediate hotspotting, explore the underlying causes of bottlenecks, why auto-provisioned FlexCache isn't sufficient, and the technical details necessary to effectively architect a FlexCache solution. By understanding and implementing high-density

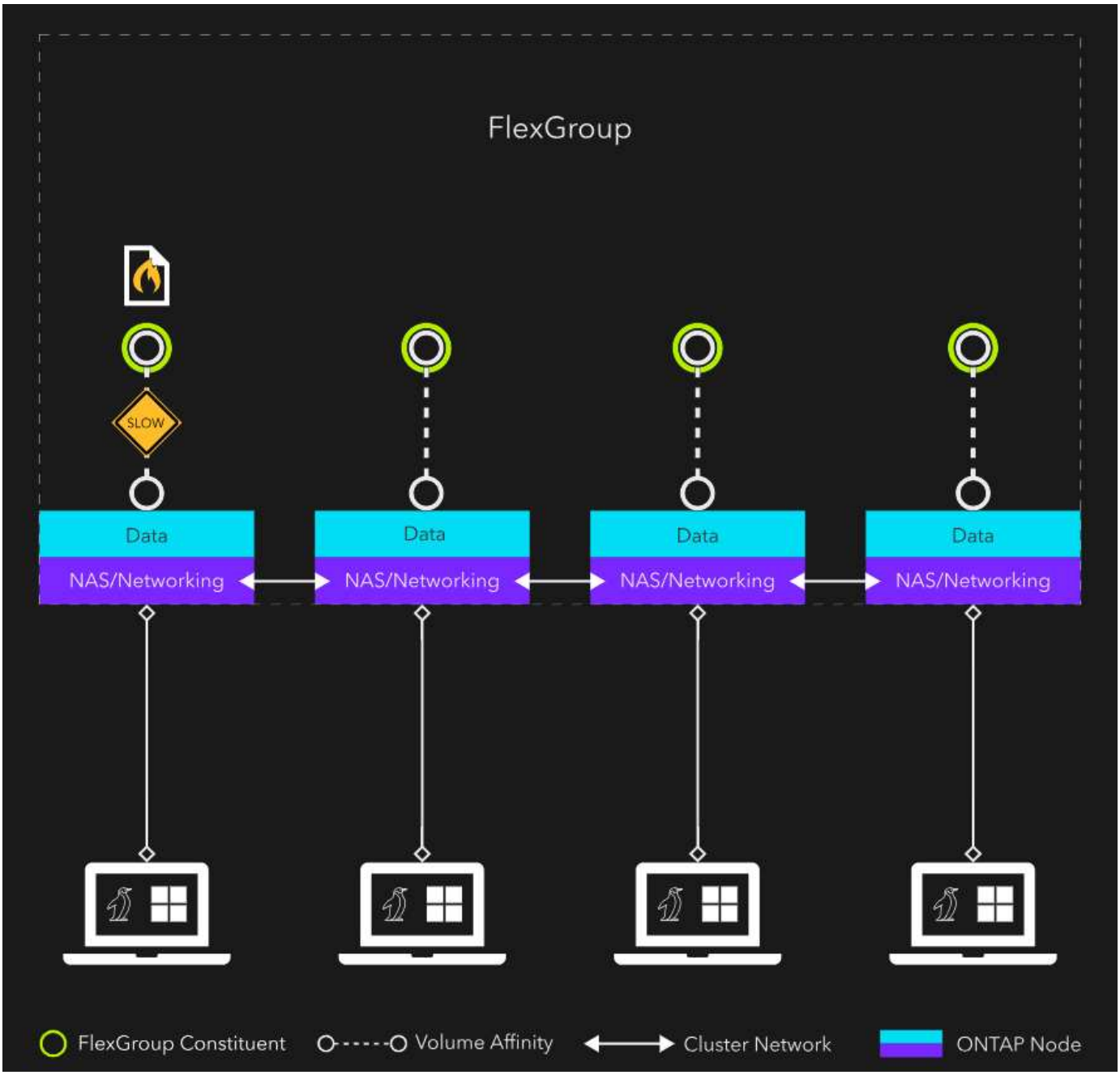
FlexCache arrays (HDFAs), you can optimize performance and eliminate bottlenecks in your high-demand workloads.

Understanding the bottleneck

The following image shows a typical single-file hotspotting scenario. The volume is a FlexGroup with a single constituent per node, and the file resides on node 1.

If you distribute all of the NAS clients' network connections across different nodes in the cluster, you still bottleneck on the CPU that services the volume affinity where the hot file resides. You also introduce cluster network traffic (east-west traffic) to the calls coming from clients connected to nodes other than where the file resides. The east-west traffic overhead is typically small, but for high-performance compute workloads every little bit counts.

Figure 1: FlexGroup single-file hotspot scenario

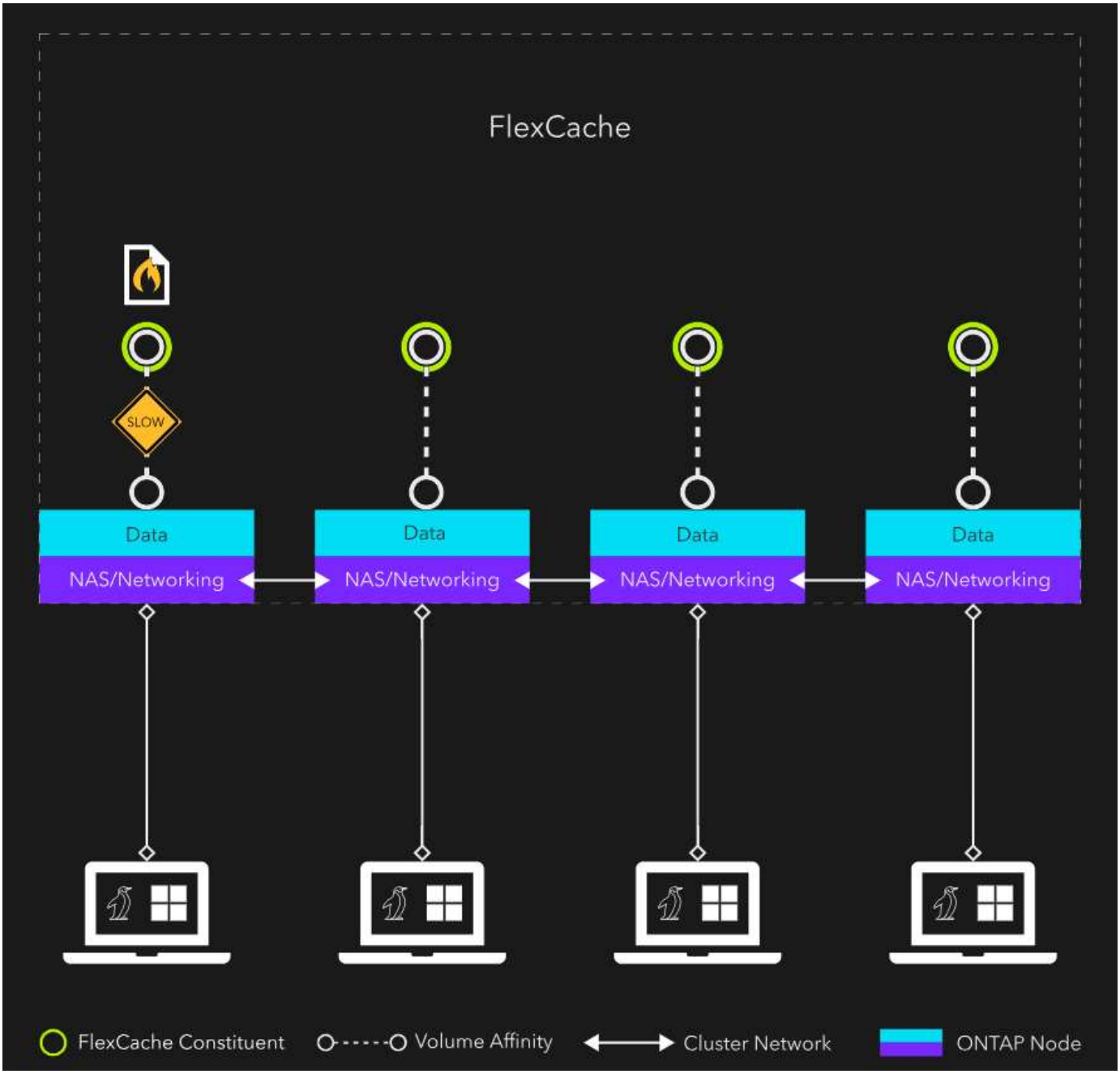


Why an auto-provisioned FlexCache isn't the answer

To remedy hotspotting, eliminate the CPU bottleneck and preferably the east-west traffic too. FlexCache can help if set up properly.

In the following example, FlexCache is auto-provisioned with System Manager, NetApp Console, or default CLI arguments. Figure 1 and figure 2 at first appear the same: both are four-node, single-constituent NAS containers. The only difference is that figure 1's NAS container is a FlexGroup, and figure 2's NAS container is a FlexCache. Each figure profiles the same bottleneck: node 1's CPU for volume affinity servicing access to the hot file, and east-west traffic contributing to latency. An auto-provisioned FlexCache hasn't eliminated the bottleneck.

Figure 2: Auto-provisioned FlexCache scenario



Anatomy of a FlexCache

To effectively architect a FlexCache for hotspot remediation, you need to understand some technical details about FlexCache.

FlexCache is always a sparse FlexGroup. A FlexGroup is made up of multiple FlexVols. These FlexVols are called FlexGroup constituents. In a default FlexGroup layout, there are one or more constituents per node in the cluster. The constituents are "sewn together" under an abstraction layer and presented to the client as a single large NAS container. When a file is written to a FlexGroup, ingest heuristics determine which constituent the file will be stored on. It might be a constituent containing the client's NAS connection or it might be a different node. The location is irrelevant because everything operates under the abstraction layer and is invisible to the client.

Let's apply this understanding of FlexGroup to FlexCache. Because FlexCache is built on a FlexGroup, by default you have a single FlexCache that has constituents on all the nodes in the cluster, as depicted in figure 1. In most cases, this is a great thing. You are utilizing all the resources in your cluster.

For remediating hot files, however, this isn't ideal because of the two bottlenecks: CPU for a single file and east-west traffic. If you create a FlexCache with constituents on every node for a hot file, that file will still reside on only one of the constituents. This means there's one CPU to service all access to the hot file. You also want to limit the amount of east-west traffic required to reach the hot file.

The solution is an array of high-density FlexCaches.

Anatomy of a high-density FlexCache

A high-density FlexCache (HDF) will have constituents on as few nodes as the capacity requirements for the cached data allow. The goal is to get your cache to live on a single node. If capacity requirements make that impossible, you can have constituents on only a few nodes instead.

For example, a 24-node cluster could have three high-density FlexCaches:

- One that spans nodes 1 through 8
- A second that spans nodes 9 through 16
- A third that spans nodes 17 through 24

These three HDFs would make up one high-density FlexCache array (HDFA). If the files are evenly distributed within each HDF, you will have a one-in-eight chance that the file requested by the client resides local to the front-end NAS connection. If you were to have 12 HDFs that span only two nodes each, you have a 50% chance of the file being local. If you can collapse the HDF down to a single node, and create 24 of them, you are guaranteed that the file is local.

This configuration will eliminate all east-west traffic and, most importantly, will provide 24 CPUs/volume affinities for accessing the hot file.

What's next?

Decide on FlexCache array density

Related information

Documentation on FlexGroup and TRs

Determine ONTAP FlexCache density

Your first hotspot remediation design decision is to figure out FlexCache density. The following examples are four-node clusters. Assume that the file count is evenly distributed among all the constituents in each HDF. Assume also an even distribution of frontend NAS connections across all nodes.

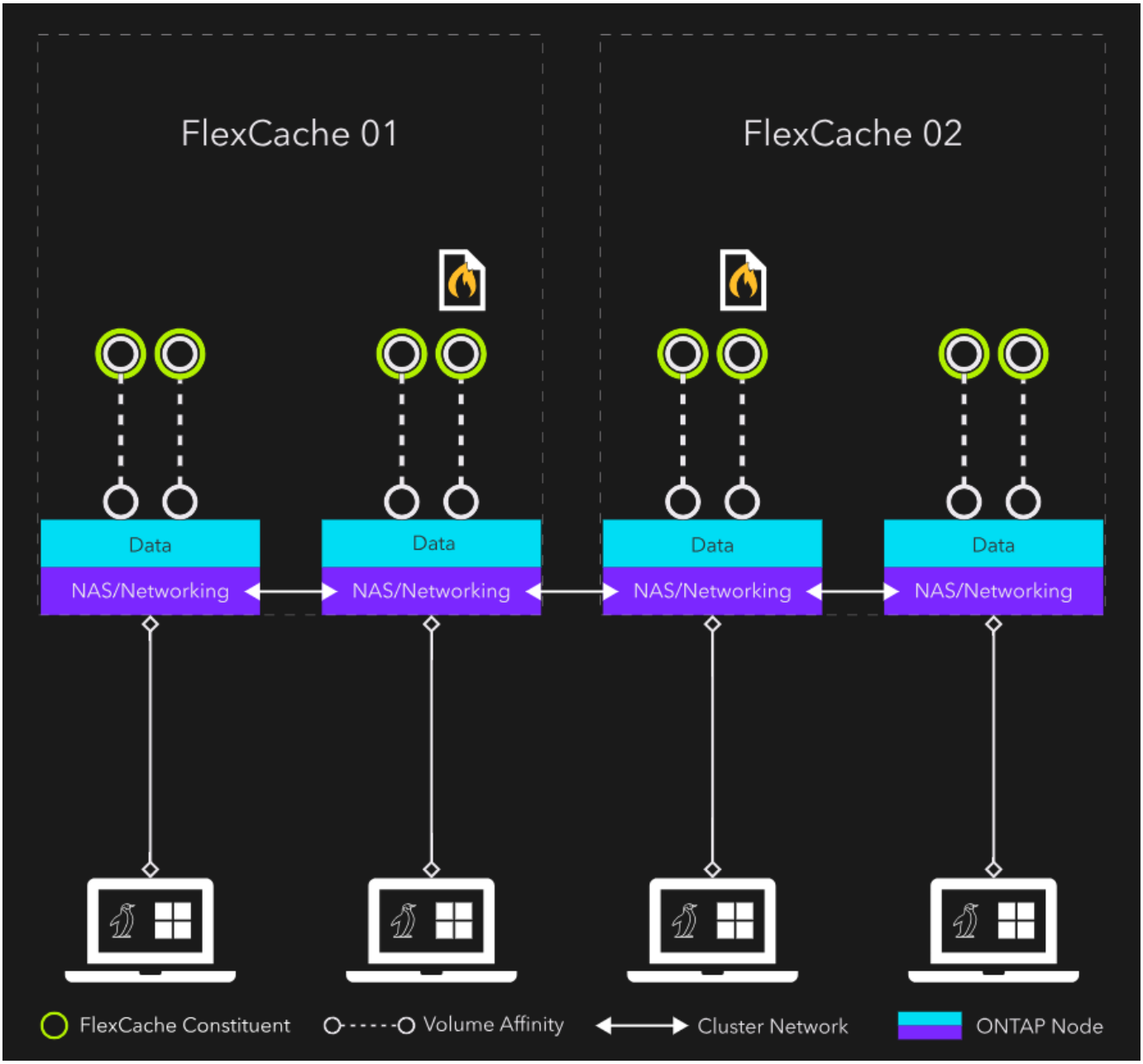
Although these examples aren't the only configurations you can use, you should understand the guiding design principle to make as many HDFs as your space requirements and available resources allow.

HDFAs are represented using the following syntax: `HDFs per HDFA x nodes per HDF x constituents per node per HDF`

2x2x2 HDFA configuration

Figure 1 is an example of a 2x2x2 HDFA configuration: two HDFs, each spanning two nodes, and each node containing two constituent volumes. In this example, each client has a 50% chance of having direct access to the hot file. Two of the four clients have east-west traffic. Importantly, there are now two HDFs, which means two distinct caches of the hot file. There are now two CPUs/volume affinities servicing access to the hot file.

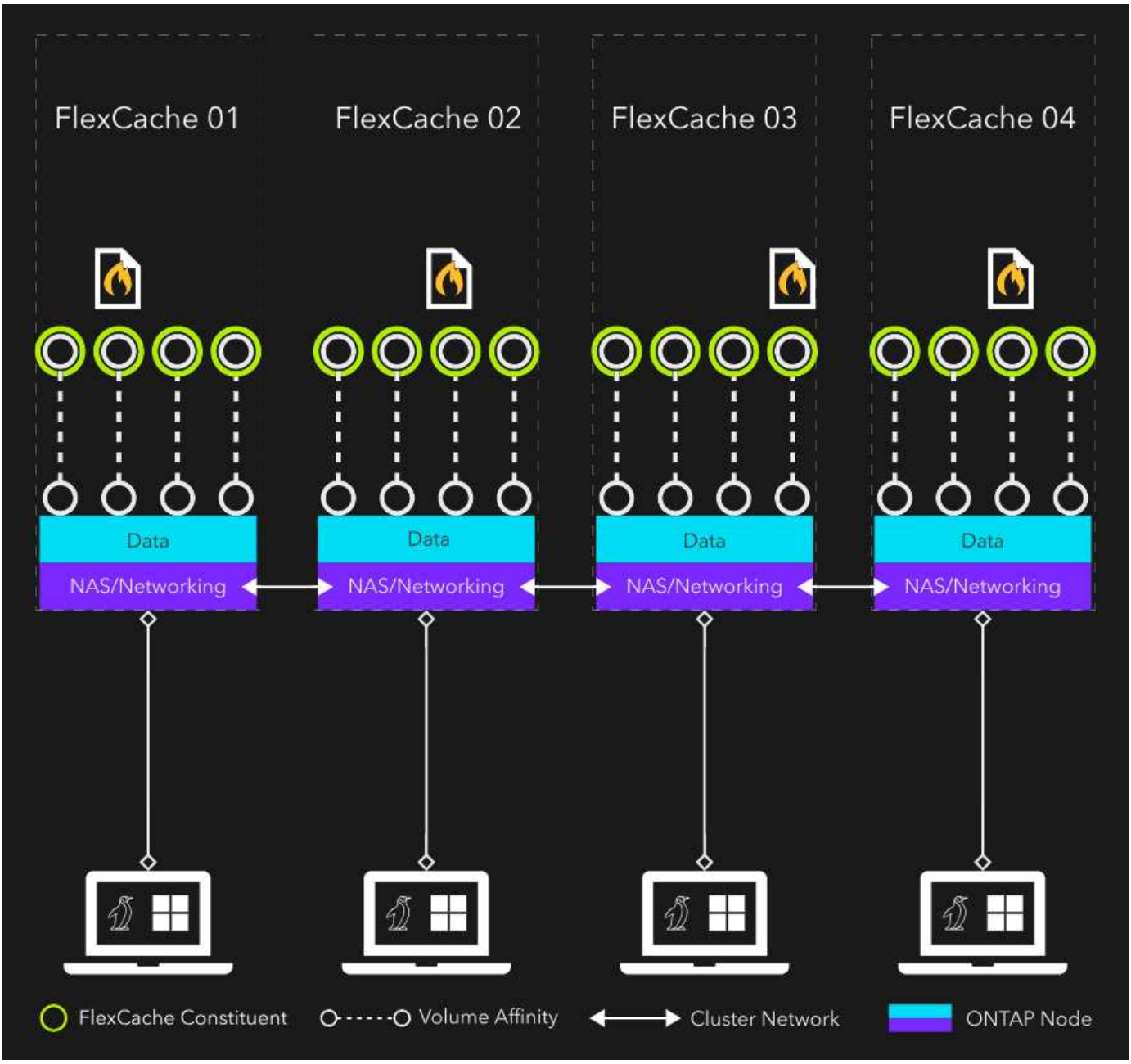
Figure 1: 2x2x2 HDFA configuration



4x1x4 HDFA configuration

Figure 2 represents an optimal configuration. It is an example of a 4x1x4 HDFA configuration: four HDFs, each contained to a single node, and each node containing four constituents. In this example, each client is guaranteed to have direct access to a cache of the hot file. Since there are four cached files on four different nodes, four different CPUs/volume affinities help service access to the hot file. Additionally, there is zero east-west traffic generated.

Figure 2: 4x1x4 HDFA configuration



What's next

After you decide how dense you want to make your HDFs, you must make another design decision if you will be accessing the HDFs with NFS with inter-SVM HDFAs and intra-SVM HDFAs.

Determine an ONTAP inter-SVM or intra-SVM HDFA option

After you determine the density of your HDFs, decide whether you will be accessing the HDFs using NFS and learn about inter-SVM HDFA and intra-SVM HDFA options.

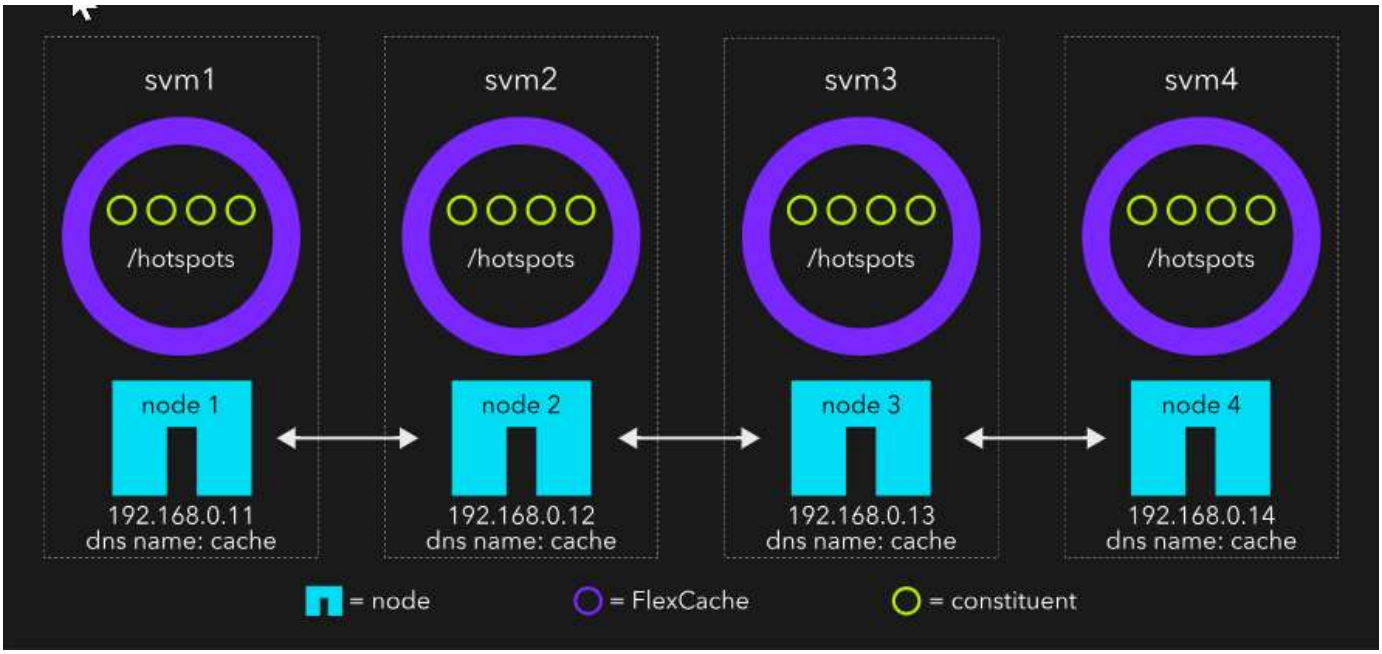
If only SMB clients will be accessing the HDFs, you should create all HDFs in a single SVM. Refer to Windows client configuration to see how to use DFS targets for load balancing.

Inter-SVM HDFA deployment

An inter-SVM HDFA requires an SVM be created for each HDF in the HDFA. This allows all HDFs within the HDFA to have the same junction-path, allowing for easier configuration on the client side.

In the figure 1 example, each HDF is in its own SVM. This is an inter-SVM HDFA deployment. Each HDF has a junction-path of /hotspots. Also, every IP has a DNS A record of hostname cache. This configuration leverages DNS round-robin to load balance mounts across the different HDFs.

Figure 1: 4x1x4 inter-SVM HDFA configuration

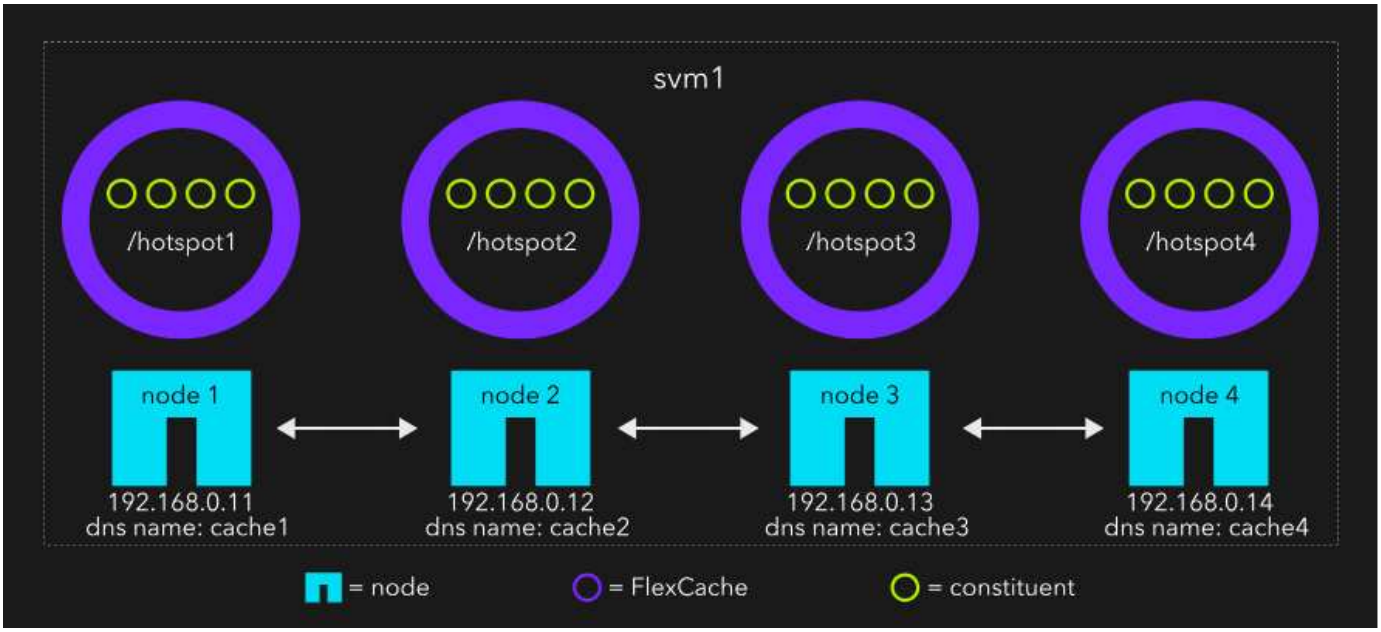


Intra-SVM HDFA deployment

An intra-SVM requires each HDF to have a unique junction-path, but all HDFs are in one SVM. This setup is easier in ONTAP because it requires only one SVM, but it needs more advanced configuration on the Linux side with `autofs` and data LIF placement in ONTAP.

In the figure 2 example, every HDF is in the same SVM. This is an intra-SVM HDFA deployment and requires the junction-paths to be unique. To make load balancing work appropriately, you'll need to create a unique DNS name for each IP and place the data LIFs the hostname resolves to only on the nodes where the HDF resides. You'll also need to configure `autofs` with multiple entries as covered in Linux client configuration.

Figure 2: 4x1x4 intra-SVM HDFA configuration



What's next

Now that you have an idea of how you want to deploy your HDFAs, deploy the HDFA and configure the clients to access them in a distributed fashion.

Configure HDFAs and ONTAP data LIFs

You'll need to configure the HDFA and the data LIFs appropriately to realize the benefits of this hotspot remediation solution. This solution uses intracluster caching with the origin and HDFA in the same cluster.

The following are two HDFA sample configurations:

- 2x2x2 inter-SVM HDFA
- 4x1x4 intra-SVM HDFA

About this task

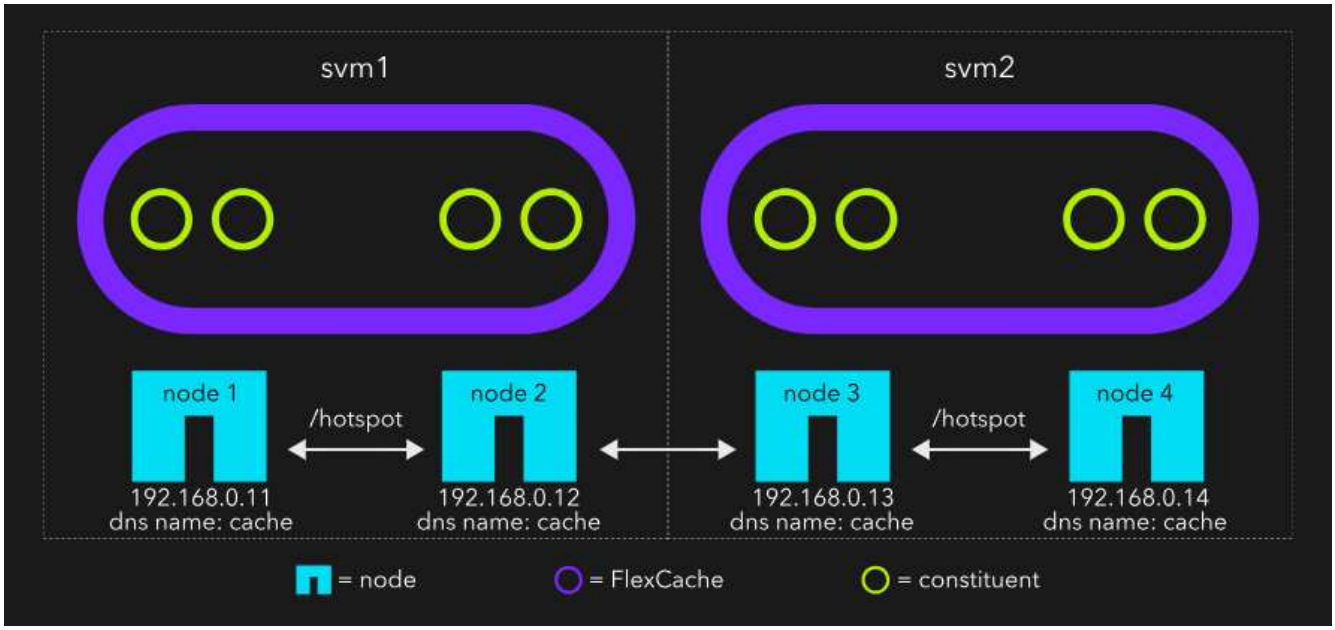
Perform this advanced configuration using the ONTAP CLI. There are two configurations you must use in the `flexcache create` command, and one configuration you must make sure isn't configured:

- `-aggr-list`: Provide an aggregate, or list of aggregates, that reside on the node or subset of nodes you want to restrict the HDF to.
- `-aggr-list-multiplier`: Determine how many constituents will be created per aggregate listed in the `aggr-list` option. If you have two aggregates listed, and set this value to `2`, you will end up with four constituents. NetApp recommends up to 8 constituents per aggregate, but 16 is also sufficient.
- `-auto-provision-as`: If you tab out, the CLI will try to autofill and set the value to `flexgroup`. Make sure this isn't configured. If it appears, delete it.

Create a 2x2x2 inter-SVM HDFA configuration

1. To assist in configuring a 2x2x2 inter-SVM HDFA as shown in Figure 1, complete a prep sheet.

   Figure 1: 2x2x2 Inter-SVM HDFA layout



SVM	Nodes per HDF	Aggregates	Constituents per node	Junction path	Data LIF IPs
svm1	node1, node2	aggr1, aggr2	2	/hotspot	192.168.0.11,192.168.0.12
svm2	node3, node4	aggr3, aggr4	2	/hotspot	192.168.0.13,192.168.0.14

2. Create the HDFs. Run the following command twice, once for each row in the prep sheet. Make sure you adjust the `vserver` and `aggr-list` values for the second iteration.

```
cache::> flexcache create -vserver svm1 -volume hotspot -aggr-list
aggr1,aggr2 -aggr-list-multiplier 2 -origin-volume <origin_vol> -origin
-vserver <origin_svm> -size <size> -junction-path /hotspot
```

3. Create the data LIFs. Run the command four times, creating two data LIFs per SVM on the nodes listed in the prep sheet. Make sure you adjust the values appropriately for each iteration.

```
cache::> net int create -vserver svm1 -home-port e0a -home-node node1
-address 192.168.0.11 -netmask-length 24
```

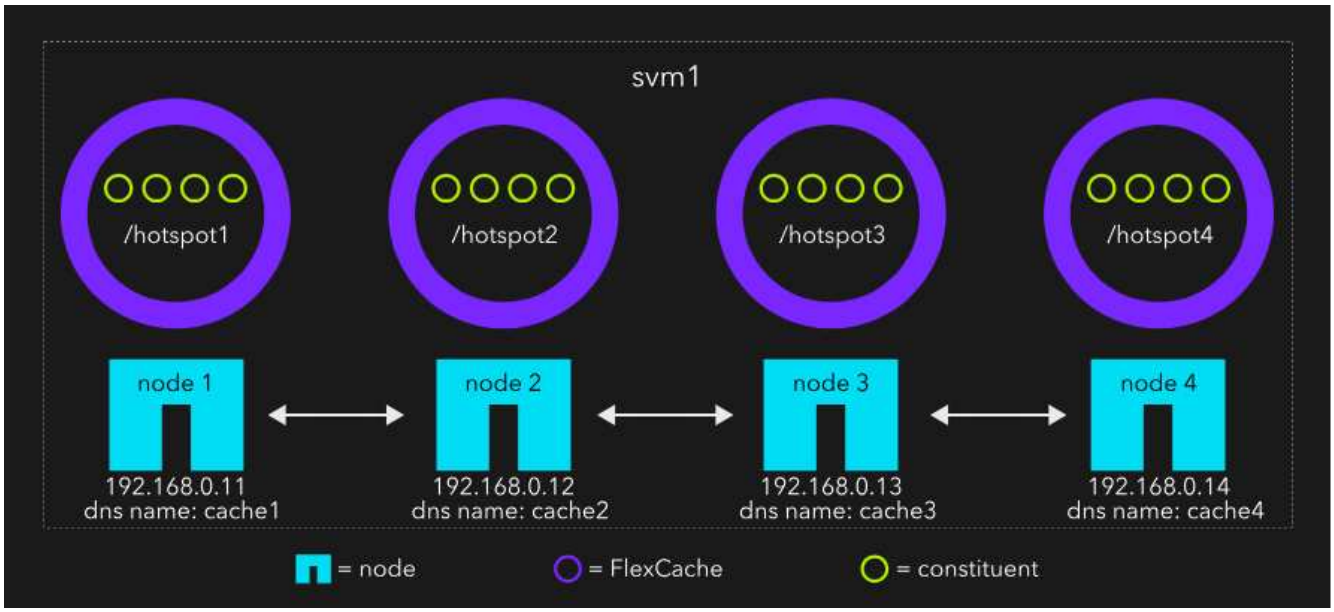
What's next

Now you need to configure your clients to utilize the HDFA appropriately. See client configuration.

Create a 4x1x4 intra-SVM HDFA

1. To assist in configuring a 4x1x4 inter-SVM HDFA as shown in figure 2, fill out a prep sheet.

Figure 2: 4x1x4 intra-SVM HDFA layout



SVM	Nodes per HDF	Aggregates	Constituents per node	Junction path	Data LIF IPs
svm1	node1	aggr1	4	/hotspot1	192.168.0.11
svm1	node2	aggr2	4	/hotspot2	192.168.0.12
svm1	node3	aggr3	4	/hotspot3	192.168.0.13
svm1	node4	aggr4	4	/hotspot4	192.168.0.14

2. Create the HDFs. Run the following command four times, once for each row in the prep sheet. Make sure you adjust the `aggr-list` and `junction-path` values for each iteration.

```
cache::> flexcache create -vserver svm1 -volume hotspot1 -aggr-list
aggr1 -aggr-list-multiplier 4 -origin-volume <origin_vol> -origin
-vserver <origin_svm> -size <size> -junction-path /hotspot1
```

3. Create the data LIFs. Run the command four times, creating a total of four data LIFs in the SVM. There should be one data LIF per node. Make sure you adjust the values appropriately for each iteration.

```
cache::> net int create -vserver svm1 -home-port e0a -home-node node1
-address 192.168.0.11 -netmask-length 24
```

What's next

Now you need to configure your clients to utilize the HDFA appropriately. See client configuration.

Configure clients to distribute ONTAP NAS connections

To remedy hotspotting, configure the client properly to do its part in preventing CPU bottleneck.

Linux client configuration

Whether you chose an intra-SVM or inter-SVM HDFA deployment, you should use `autofs` in Linux to make sure clients are load-balancing across the different HDFs. The `autofs` configuration will differ for inter- and intra-SVM.

Before you begin

You'll need `autofs` and the appropriate dependencies installed. For help with this, refer to Linux documentation.

About this task

The steps described will use an example `/etc/auto_master` file with the following entry:

```
/flexcache auto_hotspot
```

This configures `autofs` to look for a file called `auto_hotspot` in the `/etc` directory any time a process tries to access the `/flexcache` directory. The contents of the `auto_hotspot` file will dictate which NFS server and junction-path to mount inside the `/flexcache` directory. The examples described are different configurations for the `auto_hotspot` file.

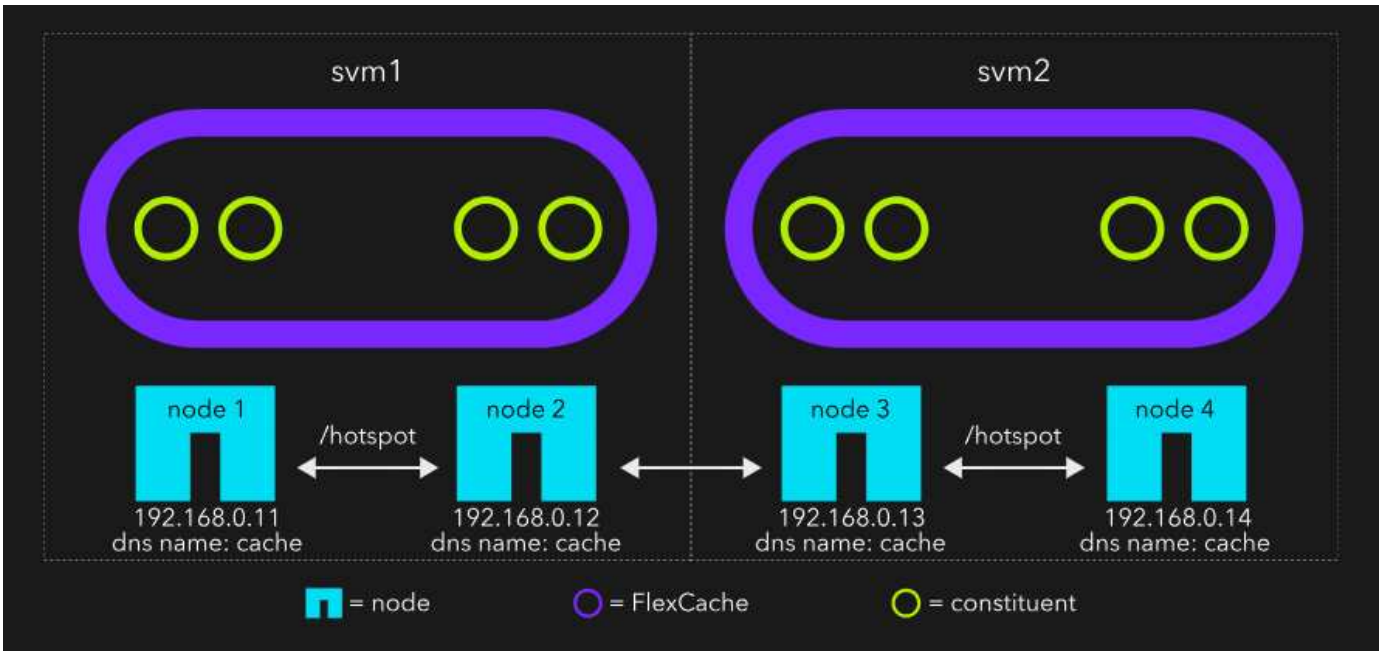
Intra-SVM HDFA autofs configuration

In the following example, we'll create an `autofs` map for the diagram in figure 1. Because each cache has the same junction-path, and the hostname `cache` has four DNS A records, we only need one line:

```
hotspot cache:/hotspot
```

This one simple line will cause the NFS client to do a DNS lookup for hostname `cache`. DNS is setup to return the IPs in a round-robin fashion. This will result in an even distribution of front-end NAS connections. After the client receives the IP, it will mount the junction-path `/hotspot` at `/flexcache/hotspot`. It could be connected to SVM1, SVM2, SVM3, or SVM4, but the particular SVM doesn't matter.

Figure 1: 2x2x2 inter-SVM HDFA



Intra-SVM HDFA autofs configuration

In the following example, we'll create an `autofs` map for the diagram in figure 2. We need to make sure the NFS clients mount the IPs that are a part of the HDF junction-path deployment. In other words, we don't want to mount `/hotspot1` with anything other than IP 192.168.0.11. To do this, we can list all four IP/junction-path pairs for one local mount location in the `auto_hotspot` map.

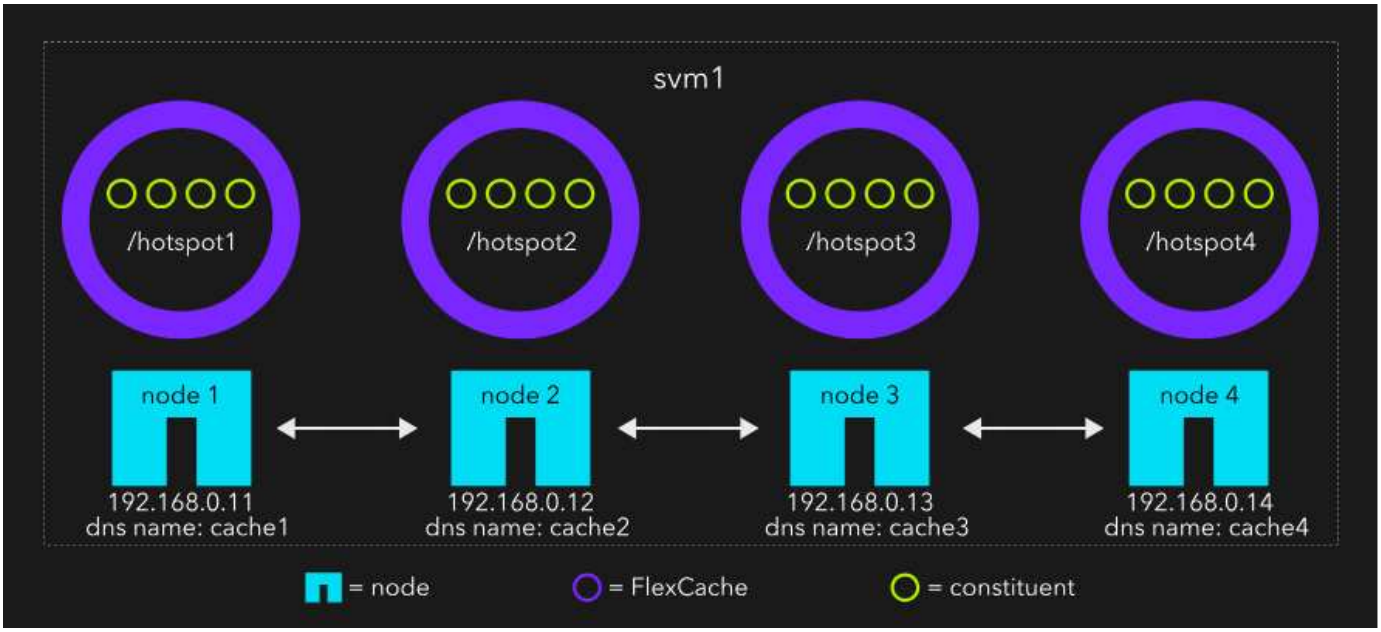
The backslash (`\`) in the following example continues the entry to the next line, making it easier to read.

```
hotspot     cache1:/hostspot1 \
            cache2:/hostspot2 \
            cache3:/hostspot3 \
            cache4:/hostspot4
```

When the client tries to access `/flexcache/hotspot`, `autofs` is going to do a forward-lookup for all four hostnames. Assuming all four IPs are either in the same subnet as the client or in a different subnet, `autofs` will issue an NFS NULL ping to each IP.

This NULL ping requires the packet to be processed by ONTAP's NFS service, but it doesn't require any disk access. The first ping to return is going to be the IP and junction-path `autofs` chooses to mount.

Figure 2: 4x1x4 intra-SVM HDFA



Windows client configuration

With Windows clients, you should use an intra-SVM HDFA. To load balance across the different HDFs in the SVM, you must add a unique share name to each HDF. After that, follow the steps in Microsoft documentation to implement multiple DFS targets for the same folder.

Copyright information

Trademark information