Storage configuration on AFF/FAS systems

Enterprise applications

NetApp
January 12, 2026

Table of Contents

Storage configuration on AFF/FAS systems

Overview

The combination of ONTAP storage solutions and Microsoft SQL Server enables enterprise-level database storage designs that can meet today's most demanding application requirements.

Optimizing a SQL Server on ONTAp solution requires understanding the SQL Server I/O pattern and characteristics. A well-designed storage layout for a SQL Server database must support the performance requirements of SQL Server while also delivering maximum managability of the infrastructure as a whole. A good storage layout also allows the initial deployment to be successful and the environment to grow smoothly over time as the business grows.

Data storage design

For SQL Server databases that do not use SnapCenter to perform backups, Microsoft recommends placing the data and log files on separate drives. For applications that simultaneously update and request data, the log file is write intensive, and the data file (depending on your application) is read/write intensive. For data retrieval, the log file is not needed. Therefore, requests for data can be satisfied from the data file placed on its own drive.

When you create a new database, Microsoft recommends specifying separate drives for the data and logs. To move files after the database is created, the database must be taken offline. For more Microsoft recommendations, see Place Data and Log Files on Separate Drives.

Aggregates

Aggregates are the lowest level storage containers for NetApp storage configurations. Some legacy documentation exists on the internet that recommends separating IO onto different sets of underlying drives. This is not recommended with ONTAP. NetApp has performed various I/O workload characterization tests using shared and dedicated aggregates with data files and transaction log files separated. The tests show that one large aggregate with more RAID groups and drives optimizes and improves storage performance and is easier for administrators to manage for two reasons:

- One large aggregate makes the I/O capabilities of all drives available to all files.
- One large aggregate enables the most efficient use of disk space.

For high availability (HA), place the SQL Server Always On Availability Group secondary synchronous replica on a separate storage virtual machine (SVM) in the aggregate. For disaster recovery purposes, place the asynchronous replica on an aggregate that is part of a separate storage cluster in the DR site, with content replicated by using NetApp SnapMirror technology. NetApp recommends having at least 10% free space available in an aggregate for optimal storage performance.

Volumes

volumes are created and reside inside aggregates. This term sometimes causes confusion because an ONTAP volume is not a LUN. An ONTAP volume is a management container for data. A volume could contain files, LUNs or even S3 objects. A volume does not take up space, it is only used for management of the contained data.

Volume design considerations

Before you create a database volume design, it is important to understand how the SQL Server I/O pattern and characteristics vary depending on the workload and on the backup and recovery requirements. See the following NetApp recommendations for flexible volumes:

- Avoid sharing volumes between hosts. For example, while it would be possible to create 2 LUNs in a single volume and share each LUN to a different host, this should be avoided because it can complicate management. In the case of running multiple SQL Server instances on the same host, unless you are close to the volume limit on a node, avoid volume sharing and instead have a separate volume per instance per host for ease of data management.
- Use NTFS mount points instead of drive letters to surpass the 26-drive-letter limitation in Windows. When using volume mount points, it is a general recommendation to give the volume label the same name as the mount point.
- When appropriate, configure a volume autosize policy to help prevent out-of-space conditions.
- If you install SQL Server on an SMB share, make sure that Unicode is enabled on the SMB volumes for creating folders.
- Set the snapshot reserve value in the volume to zero for ease of monitoring from an operational perspective.
- Disable snapshot schedules and retention policies. Instead, use SnapCenter to coordinate Snapshot copies of the SQL Server data volumes.
- Place the SQL Server system databases on a dedicated volume.
- tempdb is a system database used by SQL Server as a temporary workspace, especially for I/O intensive DBCC CHECKDB operations. Therefore, place this database on a dedicated volume with a separate set of spindles. In large environments in which volume count is a challenge, you can consolidate tempdb into fewer volumes and store it in the same volume as other system databases after careful planning. Data protection for tempdb is not a high priority because this database is recreated every time SQL Server is restarted.
- Place user data files (`.mdf`) on separate volumes because they are random read/write workloads. It is common to create transaction log backups more frequently than database backups. For this reason, place transaction log files (`.ldf`) on a separate volume or VMDK from the data files so that independent backup schedules can be created for each. This separation also isolates the sequential write I/O of the log files from the random read/write I/O of data files and significantly improves SQL Server performance.

LUNs

- Make sure that the user database files and the log directory to store log backup are on separate volumes to prevent the retention policy from overwriting snapshots when these are used with SnapVault technology.
- Do not mix database and non-database files, such as full-text search-related files, on the same LUN.
- Placing database secondary files (as part of a filegroup) on separate volumes improves the performance of the SQL Server database. This separation is valid only if the database's `.mdf` file does not share its LUN with any other `.mdf` files.
- If you create LUNs with DiskManager or other tools, make sure that the allocation unit size is set to 64K for partitions when formatting the LUNs.
- See the Microsoft Windows and native MPIO under ONTAP best practices for modern SAN to apply multipathing support on Windows to iSCSI devices in the MPIO properties.

Database files and filegroups

Proper SQL Server database file placement on ONTAP is critical during initial deployment stage. This ensures optimal performance, space management, backup and restore times that can be configured to match your business requirements.

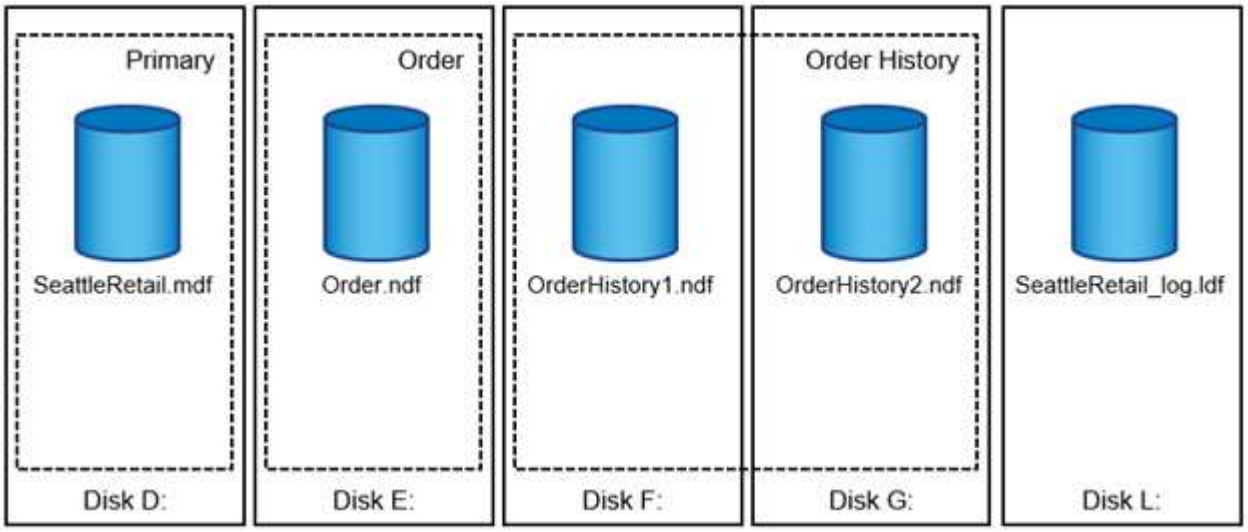
In theory, SQL Server (64-bit) supports 32,767 databases per instance and 524,272TB of database size, although the typical installation usually has several databases. However, the number of the databases SQL Server can handle depends on the load and hardware. It is not unusual to see SQL Server instances hosting dozens, hundreds, or even thousands of small databases.

Database files & filegroup

Each database consists of one or more data files and one or more transaction log files. The transaction log stores the information about database transactions and all data modifications made by each session. Every time the data is modified, SQL Server stores enough information in the transaction log to undo (roll back) or redo (replay) the action. A SQL Server transaction log is an integral part of SQL Server's reputation for data integrity and robustness. The transaction log is vital to the atomicity, consistency, isolation, and durability (ACID) capabilities of SQL Server. SQL Server writes to the transaction log as soon as any change to the data page happens. Every Data Manipulation Language (DML) statement (for example, select, insert, update, or delete) is a complete transaction, and the transaction log makes sure that the entire set-based operation takes place, making sure of the atomicity of the transaction.

Each database has one primary data file, which, by default, has the .mdf extension. In addition, each database can have secondary database files. Those files, by default, have .ndf extensions.

All database files are grouped into filegroups. A filegroup is the logical unit, which simplifies database administration. They allow the separation between logical object placement and physical database files. When you create the database objects tables, you specify in what filegroup they should be placed without worrying about the underlying data file configuration.



The ability to put multiple data files inside the filegroup allows you to spread the load across different storage devices, which helps to improve the I/O performance of the system. The transaction log in contrast does not benefit from the multiple files because SQL Server writes to the transaction log in a sequential manner.

The separation between logical object placement in the filegroups and physical database files allows you to fine-tune the database file layout, getting the most from the storage subsystem. The number of datafiles

supporting a giving workload can be varied as required to support I/O requirements and expected capacity without affecting the application. Those variations in database layout are transparent to the application developers, who are placing the database objects in the filegroups rather than database files.

NetApp recommends avoiding the use of the primary filegroup for anything but system objects. Creating a separate filegroup or set of filegroups for the user objects simplifies database administration and disaster recovery, especially in the case of large databases.

Database instance file initialization

You can specify initial file size and autogrowth parameters at the time when you create the database or add new files to an existing database. SQL Server uses a proportional fill algorithm when choosing which data file it should write data into. It writes an amount of data proportionally to the free space available in the files. The more free space in the file, the more writes it handles.

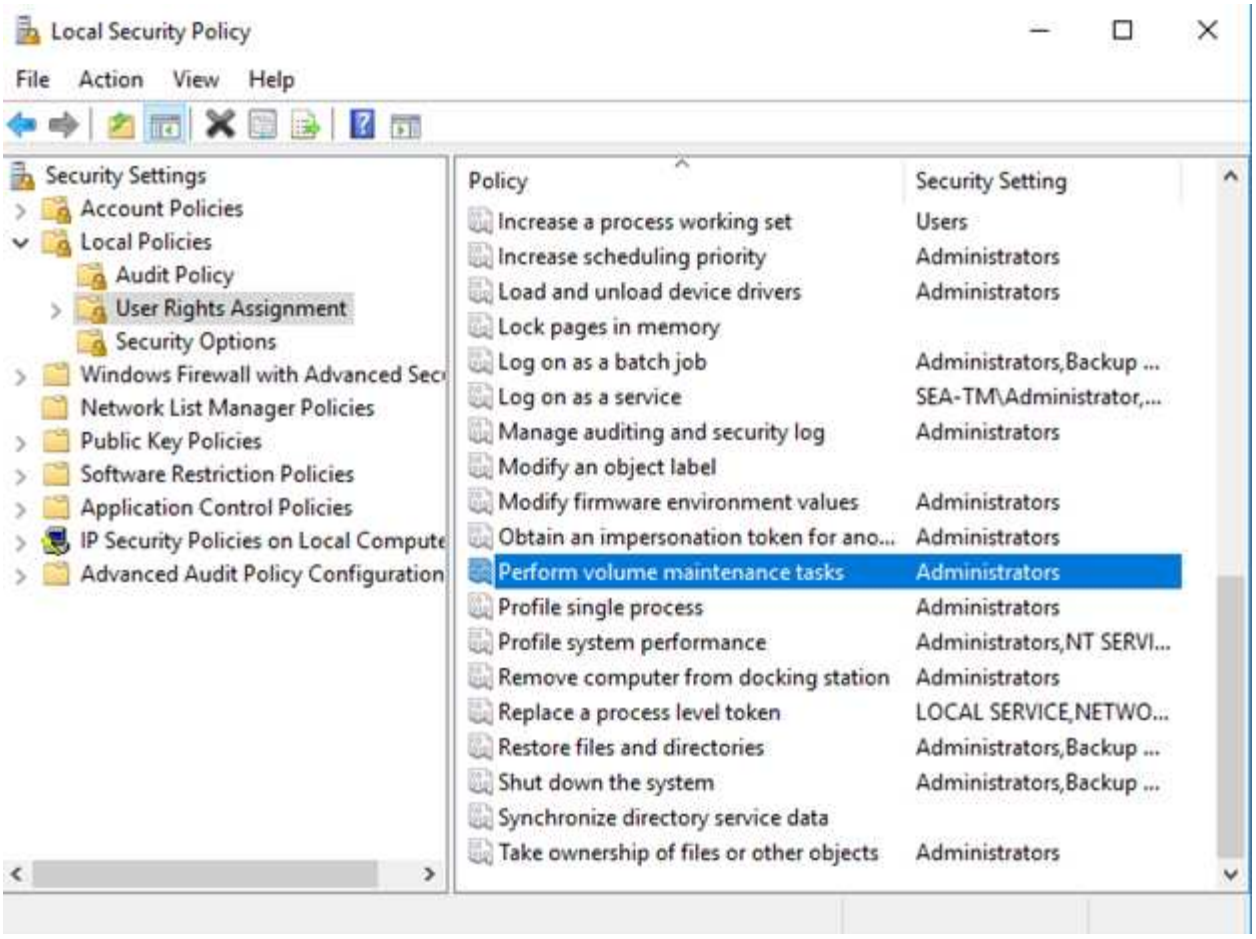
NetApp recommends that all files in the single filegroup have the same initial size and autogrowth parameters, with the grow size defined in megabytes rather than percentages. This helps the proportional fill algorithm evenly balance write activities across data files.

Every time SQL Server grows files, it fills newly allocated space with zeros. That process blocks all sessions that need to write to the corresponding file or, in case of transaction log growth, generate transaction log records.

SQL Server always zeroes out the transaction log, and that behavior cannot be changed. However, you can control whether data files are zeroing out by enabling or disabling instant file initialization. Enabling instant file initialization helps to speed up data file growth and reduces the time required to create or restore the database.

A small security risk is associated with instant file initialization. When this option is enabled, unallocated parts of the data file can contain information from previously deleted OS files. Database administrators can examine such data.

You can enable instant file initialization by adding the SA_MANAGE_VOLUME_NAME permission, also known as “perform volume maintenance task,” to the SQL Server startup account. You can do this under the local security policy management application (secpol.msc), as shown in the following figure. Open the properties for the “perform volume maintenance task” permission and add the SQL Server startup account to the list of users there.



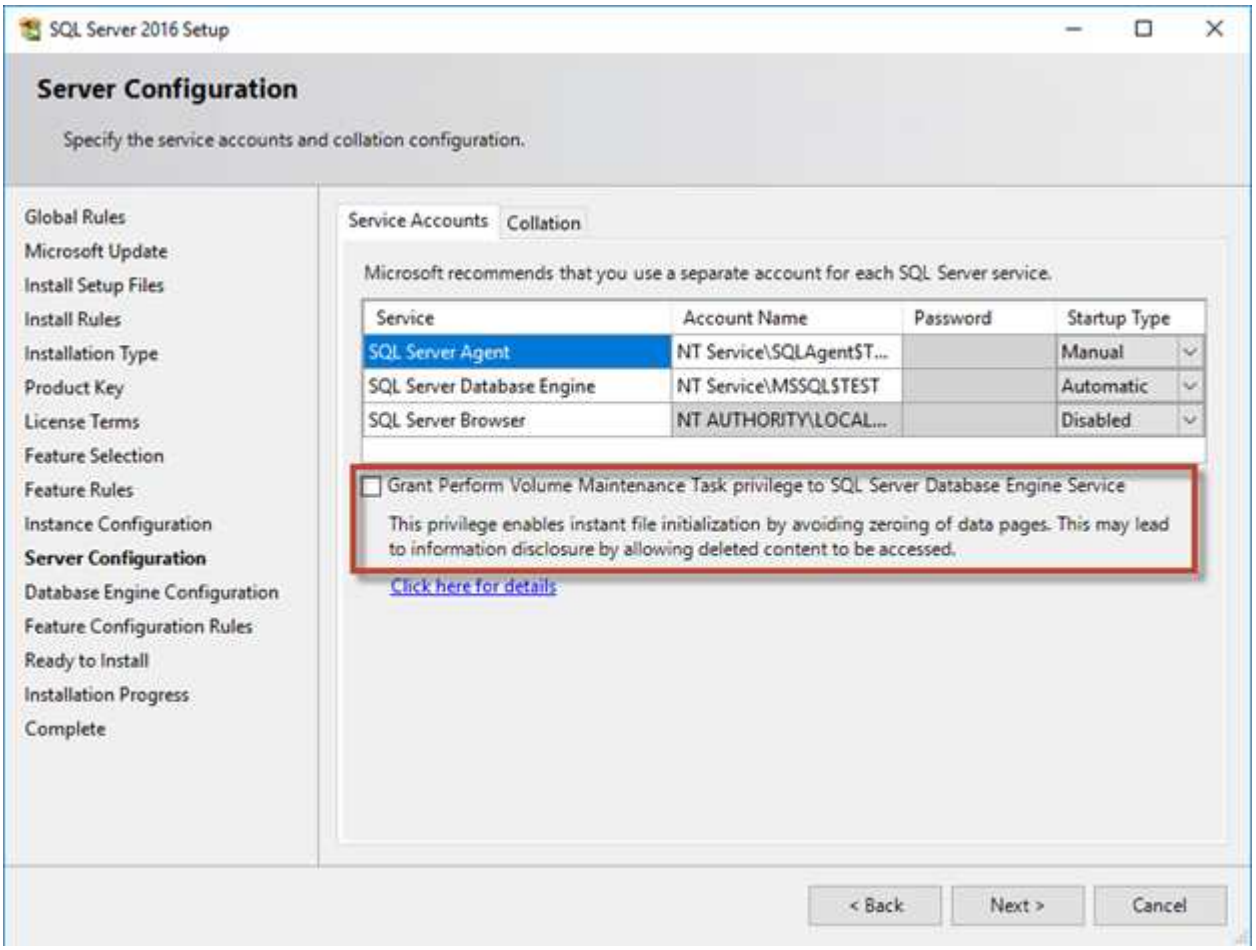
To check if the permission is enabled, you can use the code from the following example. This code sets two trace flags that force SQL Server to write additional information to the error log, create a small database, and read the content of the log.

```
DBCC TRACEON(3004,3605,-1)
GO
CREATE DATABASE DelMe
GO
EXECUTE sp_readerrorlog
GO
DROP DATABASE DelMe
GO
DBCC TRACEOFF(3004,3605,-1)
GO
```

When instant file initialization is not enabled, the SQL Server error log shows that SQL Server is zeroing the mdf data file in addition to zeroing the ldf log file, as shown in the following example. When instant file initialization is enabled, it displays only zeroing of the log file.

	LogDate	ProcessInfo	Text
365	2017-02-09 08:10:07.660	spid53	Ckpt dbid 3 flush delta counts.
366	2017-02-09 08:10:07.660	spid53	Ckpt dbid 3 logging active xact info.
367	2017-02-09 08:10:07.750	spid53	Ckpt dbid 3 phase 1 ended (8)
368	2017-02-09 08:10:07.750	spid53	About to log Checkpoint end.
369	2017-02-09 08:10:07.880	spid53	Ckpt dbid 3 complete
370	2017-02-09 08:10:08.130	spid53	Starting up database 'DelMe'.
371	2017-02-09 08:10:08.150	spid53	FixupLogTail(progress) zeroing C:\Program Files\Micros
372	2017-02-09 08:10:08.160	spid53	Zeroing C:\Program Files\Microsoft SQL Server\MSSQ
373	2017-02-09 08:10:08.170	spid53	Zeroing completed on C:\Program Files\Microsoft SQL
374	2017-02-09 08:10:08.710	spid53	Ckpt dbid 6 started
375	2017-02-09 08:10:08.710	spid53	About to log Checkpoint begin.

The Perform Volume Maintenance task is simplified in SQL Server 2016 and is later provided as an option during the installation process. This figure displays the option to grant the SQL Server database engine service the privilege to perform the volume maintenance task.



Another important database option that controls the database file sizes is autoshrink. When this option is enabled, SQL Server regularly shrinks the database files, reduces their size, and releases space to the operating system. This operation is resource intensive and is rarely useful because the database files grow again after some time when new data comes into the system. Autoshrink should not be enabled on the database.

Log directory

The log directory is specified in SQL Server to store transaction log backup data at the host level. If you are using SnapCenter to backup log files then each SQL Server host used by SnapCenter must have a host log directory configured to perform log backups. SnapCenter has a database repository, so metadata related to backup, restore, or cloning operations is stored in a central database repository.

The sizes of the host log directory is calculated as follows:
Size of host log directory = ((maximum DB LDF size x daily log change rate %) x (snapshot retention) ÷ (1 - LUN overhead space %)
The host log directory sizing formula assumes a 10% LUN overhead space

Place the log directory on a dedicated volume or LUN. The amount of data in the host log directory depends on the size of the backups and the number of days that backups are retained. SnapCenter allows only one host log directory per SQL Server host. You can configure the host log directories at SnapCenter -⟶ Host -⟶ Configure Plug-in.

NetApp recommends the following for a host log directory:

- Make sure that the host log directory is not shared by any other type of data that can potentially corrupt the backup snapshot data.
- Do not place user databases or system databases on a LUN that hosts mount points.
- Create the host log directory on a dedicated volume to which SnapCenter copies transaction logs.
- Use SnapCenter wizards to migrate databases to NetApp storage so that the databases are stored in valid locations, enabling successful SnapCenter backup and restore operations. Keep in mind that the migration process is disruptive and can cause the databases to go offline while the migration is in progress.
- The following conditions must be in place for failover cluster instances (FCIs) of SQL Server:
  - If you are using a failover cluster instance, the host log directory LUN must be a cluster disk resource in the same cluster group as the SQL Server instance being backed up SnapCenter.
  - If you are using a failover cluster instance, user databases must be placed on shared LUNs that are physical disk cluster resources assigned to the cluster group associated with the SQL Server instance.

Storage efficiency

ONTAP storage efficiency is optimized to store and manage SQL Server data in a way that consumes the least amount of storage space with no effect on performance.

Space efficiency features, such as compression, compaction, and deduplication are designed to increase the amount of logical data that fits on a given amount of physical storage. The result is lower costs and management overhead.

At a high level, compression is a mathematical process whereby patterns in data are detected and encoded in a way that reduces space requirements. In contrast, deduplication detects actual repeated blocks of data and removes the extraneous copies. Compaction allows multiple logical blocks of data to share the same physical block on media.

See the sections below on thin provisioning for an explanation of the interaction between storage efficiency and fractional reservation.

Compression

Prior to the availability of all-flash storage systems, array-based compression was of limited value because most I/O-intensive workloads required a very large number of spindles to provide acceptable performance. Storage systems invariably contained much more capacity than required as a side effect of the large number of drives. The situation has changed with the rise of solid-state storage. There is no longer a need to vastly overprovision drives purely to obtain good performance. The drive space in a storage system can be matched to actual capacity needs.

The increased IOPS capability of solid-state drives (SSDs) almost always yields cost savings compared to spinning drives, but compression can achieve further savings by increasing the effective capacity of solid-state media.

There are several ways to compress data. Many databases include their own compression capabilities, but this is rarely observed in customer environments. The reason is usually the performance penalty for a **change** to compressed data, plus with some applications there are high licensing costs for database-level compression. Finally, there is the overall performance consequences to database operations. It makes little sense to pay a high per-CPU license cost for a CPU that performs data compression and decompression rather than real database work. A better option is to offload the compression work on to the storage system.

Adaptive compression

Adaptive compression has been thoroughly tested with enterprise workloads with no observed effect on performance, even in an all-flash environment in which latency is measured in microseconds. Some customers have even reported a performance increase with the use of compression because the data remains compressed in cache, effectively increasing the amount of available cache in a controller.

ONTAP manages physical blocks in 4KB units. Adaptive compression uses a default compression block size of 8KB, which means data is compressed in 8KB units. This matches the 8KB block size most often used by relational databases. Compression algorithms become more efficient as more data is compressed as a single unit. A 32KB compression block size would be more space-efficient than an 8KB compression block unit. This does mean that adaptive compression using the default 8KB block size does lead to slightly lower efficiency rates, but there is also a significant benefit to using a smaller compression block size. Database workloads include a large amount of overwrite activity. Overwriting a 8KB of a compressed 32KB block of data requires reading back the entire 32KB of logical data, decompressing it, updating the required 8KB region, recompressing, and then writing the entire 32KB back to the drives. This is a very expensive operation for a storage system and is the reason some competing storage arrays based on larger compression block sizes also incur a significant performance penalty with database workloads.

The block size used by adaptive compression can be increased up to 32KB. This may improve storage efficiency and should be considered for quiescent files such as transaction logs and backup files when a substantial amount of such data is stored on the array. In some situations, active databases that use a 16KB or 32KB block size may also benefit from increasing the block size of adaptive compression to match. Consult a NetApp or partner representative for guidance on whether this is appropriate for your workload.

Compression block sizes larger than 8KB should not be used alongside deduplication on streaming backup destinations. The reason is small changes to the backed-up data affect the 32KB compression window. If the window shifts, the resulting compressed data differs across the entire file. Deduplication occurs after compression, which means the deduplication engine sees each compressed backup differently. If deduplication of streaming backups is required, only 8KB block adaptive compression should be used. Adaptive compression is preferable, because it works at a smaller block size and does not disrupt deduplication efficiency. For similar reasons, host-side compression also interferes with deduplication efficiency.

Compression alignment

Adaptive compression in a database environment requires some consideration of compression block alignment. Doing so is only a concern for data that is subject to random overwrites of very specific blocks. This approach is similar in concept to overall file system alignment, where the start of a filesystem must be aligned to a 4K device boundary and the blocksize of a filesystem must be a multiple of 4K.

For example, an 8KB write to a file is compressed only if it aligns with an 8KB boundary within the file system itself. This point means that it must fall on the first 8KB of the file, the second 8KB of the file, and so forth. The simplest way to ensure correct alignment is to use the correct LUN type, any partition created should have an offset from the start of the device that is a multiple of 8K, and use a filesystem block size that is a multiple of the database block size.

Data such as backups or transaction logs are sequentially written operations that span multiple blocks, all of which are compressed. Therefore, there is no need to consider alignment. The only I/O pattern of concern is the random overwrites of files.

Data compaction

Data compaction is a technology that improves compression efficiency. As stated previously, adaptive compression alone can provide at best 2:1 savings because it is limited to storing an 8KB I/O in a 4KB WAFL block. Compression methods with larger block sizes deliver better efficiency. However, they are not suitable for data that is subject to small block overwrites. Decompressing 32KB units of data, updating an 8KB portion, recompressing, and writing back to the drives creates overhead.

Data compaction works by allowing multiple logical blocks to be stored within physical blocks. For example, a database with highly compressible data such as text or partially full blocks may compress from 8KB to 1KB. Without compaction, that 1KB of data would still occupy an entire 4KB block. Inline data compaction allows that 1KB of compressed data to be stored in just 1KB of physical space alongside other compressed data. It is not a compression technology; it is simply a more efficient way of allocating space on the drives and therefore should not create any detectable performance effect.

The degree of savings obtained vary. Data that is already compressed or encrypted cannot generally be further compressed, and therefore such datasets do not benefit from compaction. In contrast, newly initialized datafiles that contain little more than block metadata and zeros compress up to 80:1.

Temperature sensitive storage efficiency

Temperature sensitive storage efficiency (TSSE) is available in ONTAP 9.8 and later. It relies on block access heat maps to identify infrequently accessed blocks and compress them with greater efficiency.

Deduplication

Deduplication is the removal of duplicate block sizes from a dataset. For example, if the same 4KB block existed in 10 different files, deduplication would redirect that 4KB block within all 10 files to the same 4KB

physical block. The result would be a 10:1 improvement in efficiency for that data.

Data such as VMware guest boot LUNs usually deduplicate extremely well because they consist of multiple copies of the same operating system files. Efficiency of 100:1 and greater have been observed.

Some data does not contain duplicate data. For example, an Oracle block contains a header that is globally unique to the database and a trailer that is nearly unique. As a result, deduplication of an Oracle database rarely delivers more than 1% savings. Deduplication with MS SQL databases is slightly better, but unique metadata at the block level is still a limitation.

Space savings of up to 15% in databases with 16KB and large block sizes have been observed in a few cases. The initial 4KB of each block contains the globally unique header, and the final 4KB block contains the nearly unique trailer. The internal blocks are candidates for deduplication, although in practice this is almost entirely attributed to the deduplication of zeroed data.

Many competing arrays claim the ability to deduplicate databases based on the presumption that a database is copied multiple times. In this respect, NetApp deduplication could also be used, but ONTAP offers a better option: NetApp FlexClone technology. The end result is the same; multiple copies of a database that share most of the underlying physical blocks are created. Using FlexClone is much more efficient than taking the time to copy database files and then deduplicating them. It is, in effect, nonduplication rather than deduplication, because a duplicate is never created in the first place.

Efficiency and thin provisioning

Efficiency features are forms of thin provisioning. For example, a 100GB LUN occupying a 100GB volume might compress down to 50GB. There are no actual savings realized yet because the volume is still 100GB. The volume must first be reduced in size so that the space saved can be used elsewhere on the system. If later changes to the 100GB LUN result in the data becoming less compressible, then the LUN grows in size and the volume could fill up.

Thin provisioning is strongly recommended because it can simplify management while delivering a substantial improvement in usable capacity with associated cost savings. The reason is simple - database environments frequently include a lot of empty space, a large number of volumes and LUNs, and compressible data. Thick provisioning results in the reservation of space on storage for volumes and LUNs just in case they someday become 100% full and contain 100% uncompressible data. That is unlikely to ever occur. Thin provisioning allows that space to be reclaimed and used elsewhere and allows capacity management to be based on the storage system itself rather than many smaller volumes and LUNs.

Some customers prefer to use thick provisioning, either for specific workloads or generally based on established operational and procurement practices.

If a volume is thick provisioned, care must be taken to completely disable all efficiency features for that volume, including decompression and the removal of deduplication using the `sis undo` command. The volume should not appear in `volume efficiency show` output. If it does, the volume is still partially configured for efficiency features. As a result, overwrite guarantees work differently, which increases the chance that configuration oversights cause the volume to unexpectedly run out of space, resulting in database I/O errors.

Efficiency best practices

NetApp recommends the following:

AFF defaults

Volumes created on ONTAP running on an all-flash AFF system are thin provisioned with all inline efficiency features enabled. Although databases generally do not benefit from deduplication and may include uncompressible data, the default settings are nevertheless appropriate for almost all workloads. ONTAP is designed to efficiently process all types of data and I/O patterns, whether or not they result in savings. Defaults should only be changed if the reasons are fully understood and there is a benefit to deviating.

General recommendations

- If volumes and/or LUNs are not thin provisioned, you must disable all efficiency settings because using these features provides no savings and the combination of thick provisioning with space efficiency enabled can cause unexpected behavior, including out-of-space errors.
- If data is not subject to overwrites, such as with backups or database transaction logs, you can achieve greater efficiency by enabling TSSE with a low cooling period.
- Some files might contain a significant amount of uncompressible data, for example when compression is already enabled at the application level of files are encrypted. If any of these scenarios are true, consider disabling compression to allow more efficient operation on other volumes containing compressible data.
- Do not use both 32KB compression and deduplication with database backups. See the section Adaptive compression for details.

Database compression

SQL Server itself also has features to compress and efficiently manage data. SQL Server currently supports two types of data compression: row compression and page compression.

Row compression changes the data storage format. For example, it changes integers and decimals to the variable-length format instead of their native fixed-length format. It also changes fixed-length character strings to the variable-length format by eliminating blank spaces. Page compression implements row compression and two other compression strategies (prefix compression and dictionary compression). You can find more details about page compression in Page Compression Implementation.

Data compression is currently supported in the Enterprise, Developer, and Evaluation editions of SQL Server 2008 and later. Although compression can be performed by the database itself, this is rarely observed in a SQL Server environment.

Here are the recommendation for managing space for SQL Server data files

- Use thin provisioning in SQL Server environments to improve space utilization and to reduce the overall storage requirements when the space guarantee functionality is used.
  - Use autogrow for most common deployment configurations because the storage admin only needs to monitor space usage in the aggregate.
- Do not enable deduplication on any volumes on FAS containing SQL Server data files unless the volume is known to contain multiple copies of the same data, such as restoring database from backups to a single volume.

Space reclamation

Space reclamation can be initiated periodically to recover unused space in a LUN. With SnapCenter, you can use the following PowerShell command to start space reclamation.

```
Invoke-SdHostVolumeSpaceReclaim -Path drive_path
```

If you need to run space reclamation, this process should be run during periods of low activity because it initially consumes cycles on the host.

Data protection

Database backup strategies should be based on identified business requirements, not theoretical capabilities. By combining ONTAP's Snapshot technology and leveraging Microsoft SQL Server API's, you can quickly take application consistent backup irrespective of size of user databases. For more advanced or scale-out data management requirements, NetApp offers SnapCenter.

SnapCenter

SnapCenter is the NetApp data protection software for enterprise applications. SQL Server databases can be quickly and easily protected with the SnapCenter Plug-in for SQL Server and with OS operations managed by the SnapCenter Plug-in for Microsoft Windows.

SQL Server instance can be a standalone setup, failover cluster instance or it can be always on availability group. The result is that from single-pane-of-glass, databases can be protected, cloned and restored from primary or secondary copy. SnapCenter can manage SQL Server databases both on-premises, in the cloud, and hybrid configurations.Database copies can also be created in few minutes on the orignal or alternate host for development or for reporting purpose.

SQL Server also requires coordination between the OS and the storage to ensure the correct data is present in snapshots at the time of creation. In most cases, the only safe method to do this is with SnapCenter or T-SQL. Snapshots created without this additional coordination may not be reliably recoverable.

For more details about the SQL Server Plug-in for SnapCenter, see TR-4714: Best practice guide for SQL Server using NetApp SnapCenter.

Protecting database using T-SQL snapshots

In SQL Server 2022, Microsoft introduced T-SQL snapshots that offers a path to scripting and automation of backups operations. Rather than performing full-sized copies, you can prepare the database for snapshots. Once the database is ready for backup, you can leveraging ONTAP REST API's to create snapshots..

The following is a sample backup workflow:

1. Freeze a database with ALTER command. This prepares the database for a consistent snapshot on the underlying storage. After the freeze you can thaw the database and record the snapshot with BACKUP command.
2. Perform snapshots of multiple databases on the storage volumes simultaneously with the new BACKUP GROUP and BACKUP SERVER commands.
3. Perform FULL backups or COPY_ONLY FULL backups. These backups are recorded in msdb as well.
4. Perform point-in-time recovery using log backups taken with the normal streaming approach after the snapshot FULL backup. Streaming differential backups are also supported if desired.

To learn more, see Microsoft documentation to know about the T-SQL snapshots.

NetApp recommends using SnapCenter to create Snapshot copies. The T-SQL method described above also works, but SnapCenter offers complete automation over the backup, restore, and cloning process. It also performs discovery to ensure the correct snapshots are being created. No pre-configuration is required.

SQL Server availability group with SnapCenter

SnapCenter supports backup of SQL Server availability group database configured with Windows failover cluster.

SnapCenter plugin for Microsoft SQL Server must be installed on all nodes of Windows server failover cluster. Refer the documentation on prerequisites and the steps to setup the SnapCenter plugins.

SnapCenter discovers all the databases, instances and availability groups in Windows hosts and resources are enumerated on the SnapCenter resource page.

Protecting databases in always on availability group

Databases in availability group can be protected in multiple ways.

- Database level backup: Select the availability database for the database resource page, add the policy consisting of full/log backup, schedule the backup. SnapCenter takes the backup irrespective of the database role whether it is a primary replica or a secondary replica. The protection can also be configured by adding databases to resource group.
- Instance level backup: Select the instance and all the databases running on the instance are protected based on the selected policy. All the databases, including the availability database running as primary or secondary replica are backed up using SnapCenter. The protection can also be configured by adding instance to resource group.
- Availability group level backup: While configuring the policy, SnapCenter have a advance option for availability group level backup. The availability group setting in policy allows users to select the replica preference for backup. You could select primary, secondary replica or all of them. The default option is based on backup replica set in SQL Server availability group configuration.

The availability group setting in SnapCenter policy will apply only if availability group level backup is used to protect availability group databases and do not apply for database or instance level backup.

NetApp recommends to use availability level backup to backup across all the replica running on NetApp ONTAP storage.

Configuring log backup in SnapCenter

If availability group is setup on standalone SQL Server setup then a dedicated disk must be mounted on each node of a Windows server failover cluster. Dedicated disk should be used to configure log directory to save transaction log backups.

If availability group is setup on SQL Server failover cluster then clustered disk should be created on SQL Server failover cluster instance to host log directory.

Restoring database in availability group setup with SnapCenter

- SnapCenter provide reseed option to automatically recover the database from the latest snapshot available at the secondary replica. Reseed operation will automatically restore and join the database backup to availability group.
- Alternate way to restore replica database in availability group is by breaking the availability group and performing the complete full and log restore. Use SnapCenter to restore database in norecovery mode and then use SQL Server management studio or T-SQL to join the database back to availability group.
- To recover just subset of data, clone capability from SnapCenter can be used to create clone copy of database. Database copy is created within few minutes using SnapCenter, then export the data to primary replica using SQL Server native tools.

For best practice to setup database storage layout to meet the RTO and RPO requirement, please see TR-4714 Best practices for Microsoft SQL Server using NetApp SnapCenter.

SnapCenter do not support distributed availability group and contained availability group.

Disaster recovery

Disaster recovery

Enterprise databases and application infrastructures often require replication to protect from natural disaster or unexpected business disruption with minimal downtime.

The SQL Server Always-On availability group replication feature can be an excellent option, and NetApp offers options to integrate data protection with Always-On. In some cases, however, you might want to consider ONTAP replication technology. There are three basic options.

SnapMirror

SnapMirror technology offers a fast and flexible enterprise solution for replicating data over LANs and WANs. SnapMirror technology transfers only changed data blocks to the destination after the initial mirror is created, significantly reducing network bandwidth requirements. It can be configured in either synchronous or asynchronous mode.

NetApp MetroCluster and SnapMirror active sync

For many customers, DR requires more than just possessing a remote copy of data, it requires the ability to rapidly make use of that data. NetApp offers two technologies that address this need - MetroCluster and SnapMirror active sync

MetroCluster refers to ONTAP in a hardware configuration that includes low-level synchronously mirrored storage and numerous additional features. Integrated solutions such as MetroCluster simplify today's complicated, scale-out database, application, and virtualization infrastructures. It replaces multiple, external data protection products and strategies with one simple, central storage array. It also provides integrated backup, recovery, disaster recovery, and high availability (HA) within a single clustered storage system.

SnapMirror active sync is based on SnapMirror Synchronous. With MetroCluster, each ONTAP controller is responsible for replicating its drive data to a remote location. With SnapMirror active sync, you essentially have two different ONTAP systems maintaining independent copies of your LUN data, but cooperating to present a single instance of that LUN. From a host point of view, it's a single LUN entity.

SM-as and MCC comparison

SM-as and MetroCluster are similar in overall functionality, but there are important differences in the way in which RPO=0 replication was implemented and how it is managed. SnapMirror asynchronous and synchronous can also be used as part of a DR plan, but they are not designed as HA repliation technologies.

- A MetroCluster configuration is more like one integrated cluster with nodes distributed across sites. SM-as behaves like two otherwise independent clusters that are cooperating in serving up select RPO=0 synchronously replicated LUNs.
- The data in a MetroCluster configuration is only accessible from one particular site at any given time. A second copy of the data is present on the opposite site, but the data is passive. It cannot be accessed without a storage system failover.
- MetroCluster and SM-as perform mirroring occurs at different levels. MetroCluster mirroring is performed at the RAID layer. The low-level data is stored in a mirrored format using SyncMirror. The use of mirroring is virtually invisible up at the LUN, volume, and protocol layers.
- In contrast, SM-as mirroring occurs at the protocol layer. The two clusters are overall independent clusters. Once the two copies of data are in sync, the two clusters only need to mirror writes. When a write occurs on one cluster, it is replicated to the other cluster. The write is only acknowledged to the host when the write has completed on both sites. Other than this protocol splitting behavior, the two clusters are otherwise normal ONTAP clusters.
- The primary role for MetroCluster is large-scale replication. You can replicate an entire array with RPO=0 and near-zero RTO. This simplifies the failover process because there is only one "thing" to fail over, and it scales extremely well in terms of capacity and IOPS.
- One key use case for SM-as is granular replication. Sometimes you don’t want to replicate all data as a single unit, or you need to be able to selectively fail over certain workloads.
- Another key use case for SM-as is for active-active operations, where you want fully usable copies of data to be available on two different clusters located in two different locations with identical performance characteristics and, if desired, no requirement to stretch the SAN across sites. You can have your applications already running on both sites, which reduces the overall RTO during failover operations.

SnapMirror

The following are recommendations for SnapMirror for SQL Server:

- If SMB is used, the destination SVM must be a member of the same Active Directory domain of which the source SVM is a member so that the access control lists (ACLs) stored within NAS files are not broken during recovery from a disaster.
- Using destination volume names that are the same as the source volume names is not required but can make the process of mounting destination volumes into the destination simpler to manage. If SMB is used, you must make the destination NAS namespace identical in paths and directory structure to the source namespace.
- For consistency purposes, do not schedule SnapMirror updates from the controllers. Instead, enable SnapMirror updates from SnapCenter to update SnapMirror after either full or log backup is completed.
- Distribute volumes that contain SQL Server data across different nodes in the cluster to allow all cluster nodes to share SnapMirror replication activity. This distribution optimizes the use of node resources.
- Use synchronous replication where demand for quick data recovery is higher and asynchronous solutions for flexibility in RPO.

For more information about SnapMirror, see TR-4015: SnapMirror Configuration and Best Practices Guide for ONTAP 9.

MetroCluster

Architecture

Microsoft SQL Server deployment with MetroCluster environment requires some explanation of physical design of a MetroCluster system.

MetroCluster synchronously mirrors data and configuration between two ONTAP clusters in separate locations or failure domains. MetroCluster provides continuously available storage for applications by automatically managing two objectives:

- Zero recovery point objective (RPO) by synchronously mirroring data written to the cluster.
- Near zero recovery time objective (RTO) by mirroring configuration and automating access to data at the second site.

MetroCluster provides simplicity with automatic mirroring of data and configuration between the two independent clusters located in the two sites. As storage is provisioned within one cluster, it is automatically mirrored to the second cluster at the second site. NetApp SyncMirror® provides a complete copy of all data with a zero RPO. This means that workloads from one site could switch over at any time to the opposite site and continue serving data without data loss. MetroCluster manages the switchover process of providing access to NAS and SAN-provisioned data at the second site. The design of MetroCluster as a validated solution contains sizing and configuration that enables a switchover to be performed within the protocol timeout periods or sooner (typically less than 120 seconds). This results in a near zero RPO and applications can continue accessing data without incurring failures.MetroCluster is available in several variations defined by the back-end storage fabric.

MetroCluster is avialable in 3 different configurations

- HA pairs with IP connectivity
- HA pairs with FC connectivity
- Single controller with FC connectivity

The term 'connectivity' refers to the cluster connection used for cross-site replication. It does not refer to the host protocols. All host-side protocols are supported as usual in a MetroCluster configuration irrespective of the type of connection used for inter-cluster communication.

MetroCluster IP

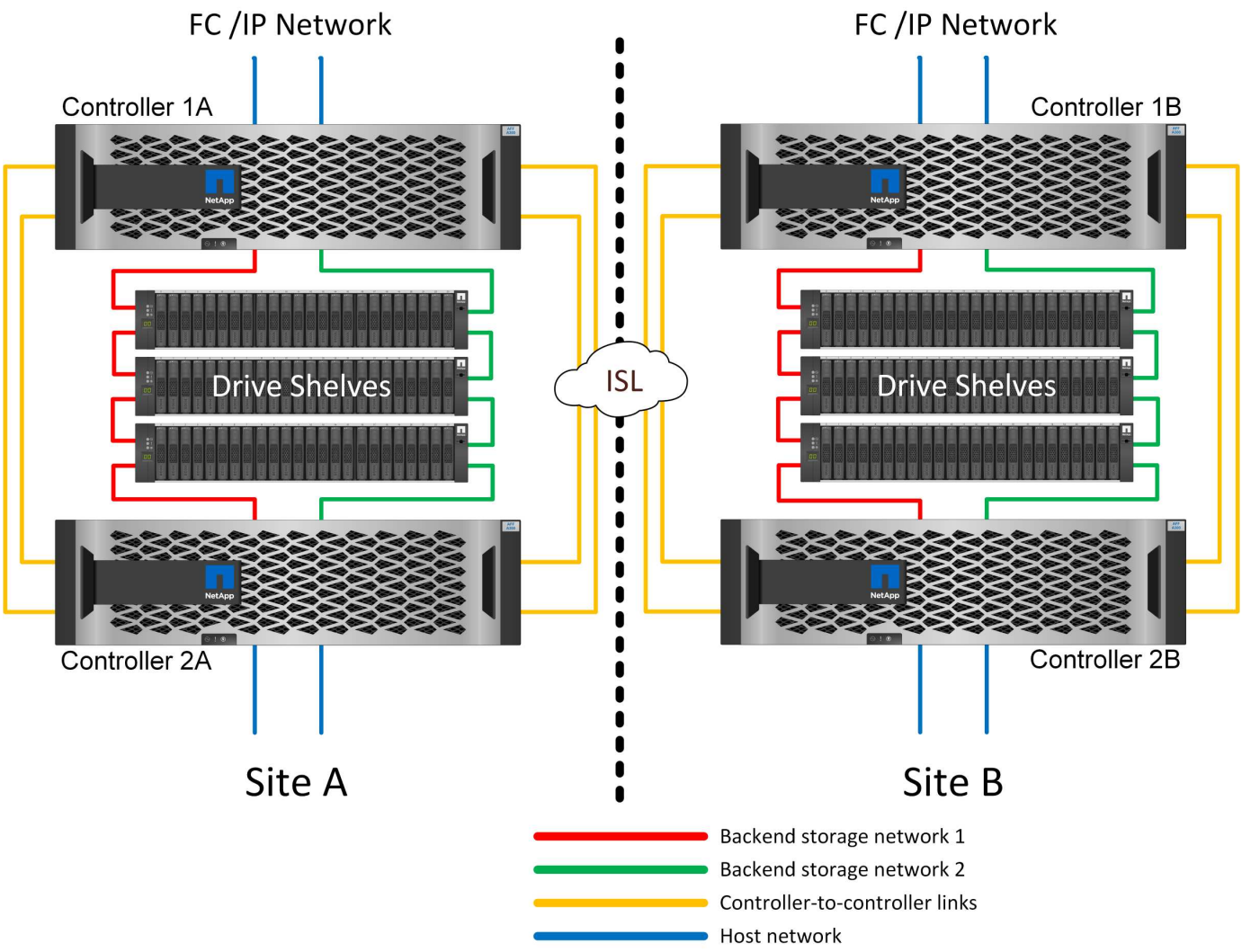
The HA-pair MetroCluster IP configuration uses two or four nodes per site. This configuration option increases the complexity and costs relative to the two-node option, but it delivers an important benefit: intrasite redundancy. A simple controller failure does not require data access across the WAN. Data access remains local through the alternate local controller.

Most customers are choosing IP connectivity because the infrastructure requirements are simpler. In the past, high-speed cross-site connectivity was generally easier to provision using dark fibre and FC switches, but today high-speed, low latency IP circuits are more readily available.

The architecture is also simpler because the only cross-site connections are for the controllers. In FC SAN attached MetroClusters, a controller writes directly to the drives on the opposite site and thus requires additional SAN connections, switches, and bridges. In contrast, a controller in an IP configuration writes to the opposite drives via the controller.

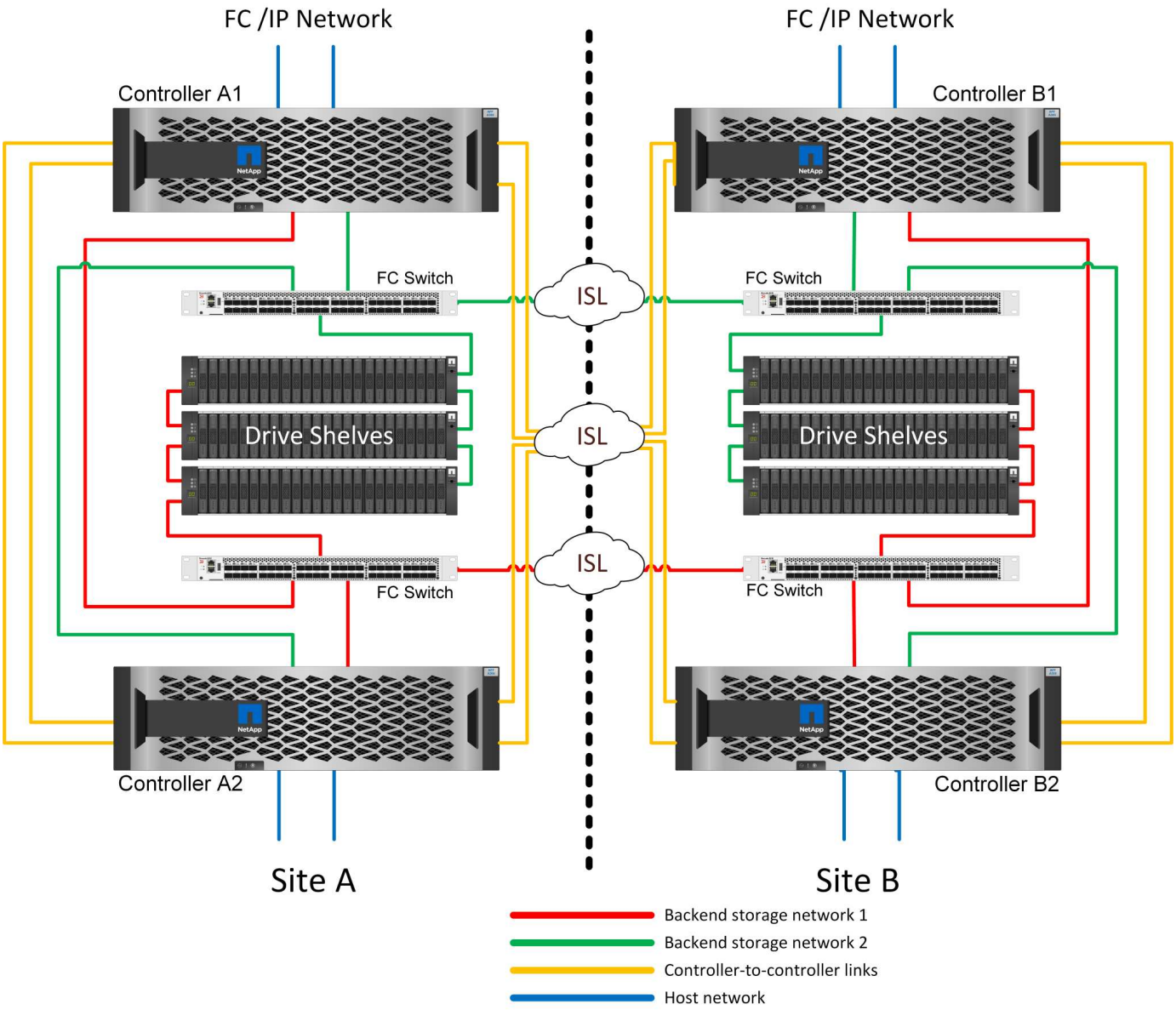
For additional information, refer to the official ONTAP documentation and MetroCluster IP Solution Architecture

and Design.



HA-Pair FC SAN-attached MetroCluster

The HA-pair MetroCluster FC configuration uses two or four nodes per site. This configuration option increases the complexity and costs relative to the two-node option, but it delivers an important benefit: intrasite redundancy. A simple controller failure does not require data access across the WAN. Data access remains local through the alternate local controller.

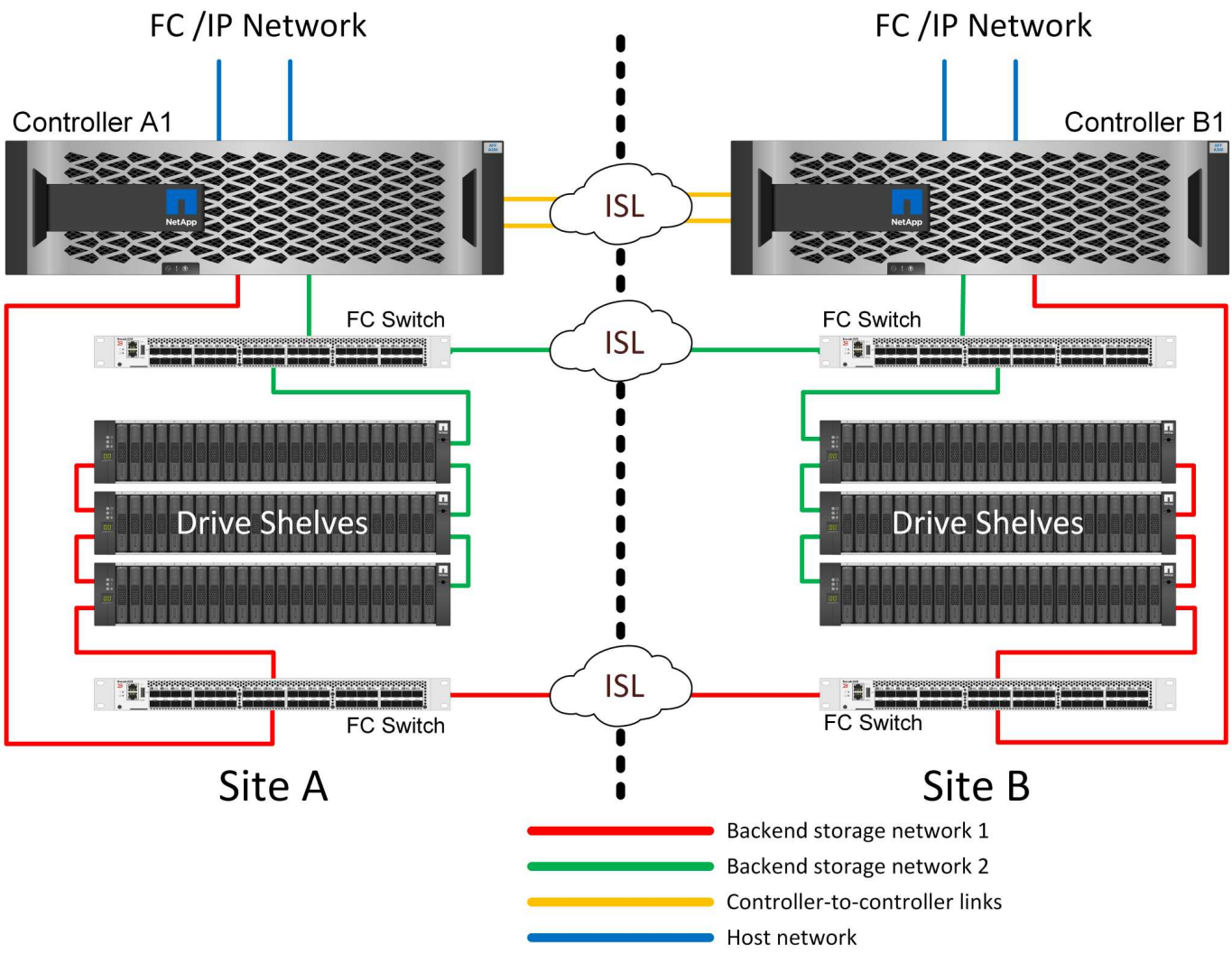


Some multisite infrastructures are not designed for active-active operations, but rather are used more as a primary site and disaster recovery site. In this situation, an HA-pair MetroCluster option is generally preferable for the following reasons:

- Although a two-node MetroCluster cluster is an HA system, unexpected failure of a controller or planned maintenance requires that data services must come online on the opposite site. If the network connectivity between sites cannot support the required bandwidth, performance is affected. The only option would be to also fail over the various host OSs and associated services to the alternate site. The HA-pair MetroCluster cluster eliminates this problem because loss of a controller results in simple failover within the same site.
- Some network topologies are not designed for cross-site access, but instead use different subnets or isolated FC SANs. In these cases, the two-node MetroCluster cluster no longer functions as an HA system because the alternate controller cannot serve data to the servers on the opposite site. The HA-pair MetroCluster option is required to deliver complete redundancy.
- If a two-site infrastructure is viewed as a single highly available infrastructure, the two-node MetroCluster configuration is suitable. However, if the system must function for an extended period of time after site failure, then an HA pair is preferred because it continues to provide HA within a single site.

Two-node FC SAN-attached MetroCluster

The two-node MetroCluster configuration uses only one node per site. This design is simpler than the HA-pair option because there are fewer components to configure and maintain. It also has reduced infrastructure demands in terms of cabling and FC switching. Finally, it reduces costs.



The obvious impact of this design is that controller failure on a single site means that data is available from the opposite site. This restriction is not necessarily a problem. Many enterprises have multisite data center operations with stretched, high-speed, low-latency networks that function essentially as a single infrastructure. In these cases, the two-node version of MetroCluster is the preferred configuration. Two-node systems are currently used at petabyte scale by several service providers.

MetroCluster resiliency features

There are no single points of failure in a MetroCluster solution:

- Each controller has two independent paths to the drive shelves on the local site.
- Each controller has two independent paths to the drive shelves on the remote site.
- Each controller has two independent paths to the controllers on the opposite site.
- In the HA-pair configuration, each controller has two paths to its local partner.

In summary, any one component in the configuration can be removed without compromising the ability of MetroCluster to serve data. The only difference in terms of resiliency between the two options is that the HA-pair version is still an overall HA storage system after a site failure.

SyncMirror

Protection for SQL Server with MetroCluster is based on SyncMirror, which gives a maximum-performance, scale-out synchronous mirroring technology.

Data protection with SyncMirror

At the simplest level, synchronous replication means any change must be made to both sides of mirrored storage before it is acknowledged. For example, if a database is writing a log, or a VMware guest is being patched, a write must never be lost. As a protocol level, the storage system must not acknowledge the write until it has been committed to nonvolatile media on both sites. Only then is it safe to proceed without the risk of data loss.

The use of a synchronous replication technology is the first step in designing and managing a synchronous replication solution. The most important consideration is understanding what could happen during various planned and unplanned failure scenarios. Not all synchronous replication solutions offer the same capabilities. If you need a solution that delivers a recovery point objective (RPO) of zero, meaning zero data loss, all failure scenarios must be considered. In particular, what is the expected result when replication is impossible due to loss of connectivity between sites?

SyncMirror data availability

MetroCluster replication is based on NetApp SyncMirror technology, which is designed to efficiently switch into and out of synchronous mode. This capability meets the requirements of customers who demand synchronous replication, but who also need high availability for their data services. For example, if connectivity to a remote site is severed, it is generally preferable to have the storage system continue operating in a non-replicated state.

Many synchronous replication solutions are only capable of operating in synchronous mode. This type of all-or-nothing replication is sometimes called domino mode. Such storage systems stop serving data rather than allowing the local and remote copies of data to become un-synchronized. If replication is forcibly broken, resynchronization can be extremely time consuming and can leave a customer exposed to complete data loss during the time that mirroring is reestablished.

Not only can SyncMirror seamlessly switch out of synchronous mode if the remote site is unreachable, it can also rapidly resync to an RPO = 0 state when connectivity is restored. The stale copy of data at the remote site can also be preserved in a usable state during resynchronization, which ensures that local and remote copies of data exist at all times.

Where domino mode is required, NetApp offers SnapMirror Synchronous (SM-S). Application-level options also exist, such as Oracle DataGuard or SQL Server Always On Availability Groups. OS-level disk mirroring can be an option. Consult your NetApp or partner account team for additional information and options.

SQL Server with MetroCluster

One option for protecting SQL Server databases with a zero RPO is MetroCluster. MetroCluster is a simple, high-performance RPO=0 replication technology that allows you to easily replicate an entire infrastructure across sites.

SQL Server can scale up to thousands of databases on a single MetroCluster system. There could be SQL

Server standalone instances or failover cluster instances, MetroCluster system does not necessarily add to or change any best practices for managing a database.

A complete explanation of MetroCluster is beyond the scope of this document, but the principles are simple. MetroCluster can provide an RPO=0 replication solution with rapid failover. What you build on top of this foundation depends on your requirements.

For example, a basic rapid DR procedure after sudden site loss could use the following basic steps:

- Force a MetroCluster switchover
- Performing discovery of FC/iSCSI LUNs (SAN only)
- Mount file systems
- Start SQL Services

The primary requirement of this approach is a running OS in place on the remote site. It must be preconfigured with SQL Server setup and should be updated with equivalent build version. SQL Server system databases can also be mirrored to the remote site and mounted if a disaster is declared.

If the volumes, file systems and datastore hosting virtualized databases are not in use at the disaster recovery site prior to the switchover, there is no requirement to set `dr-force-nvfail` on associated volumes.

SnapMirror active sync

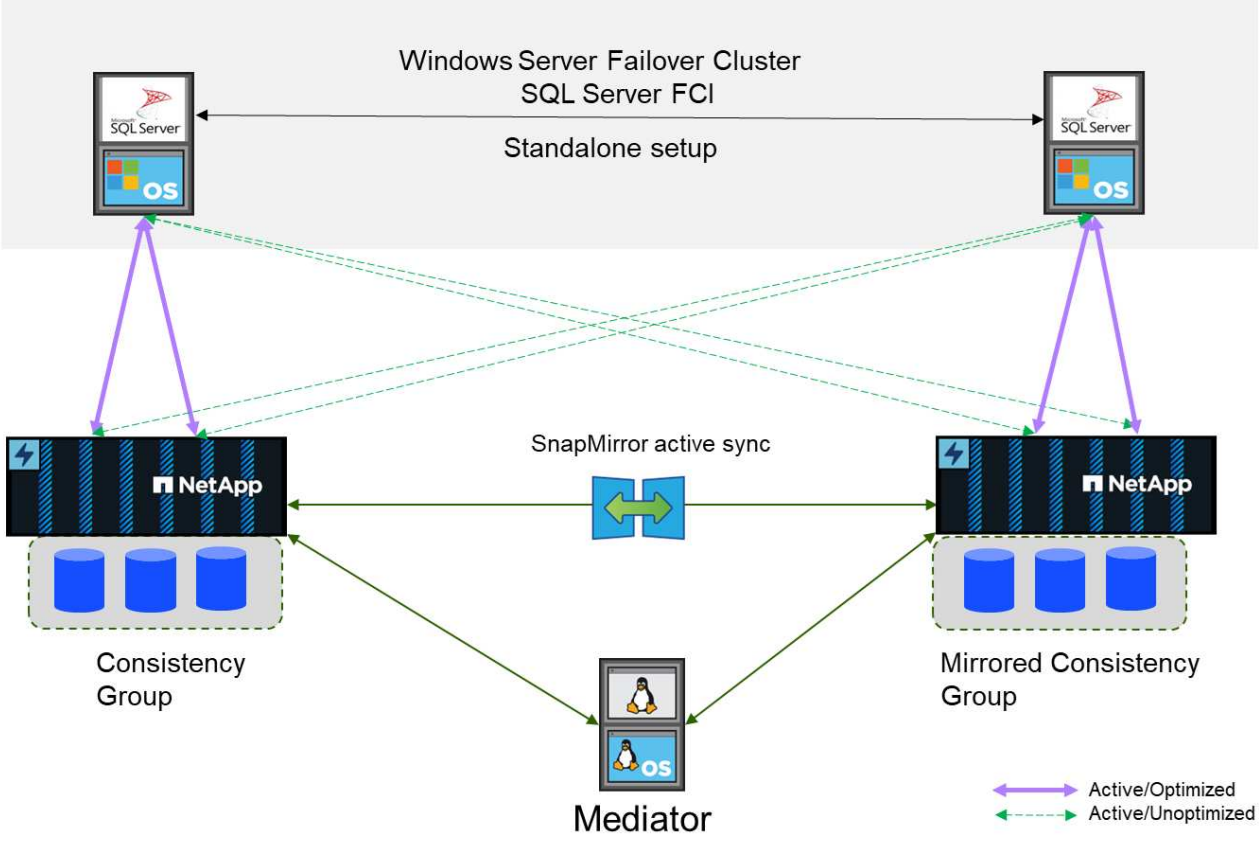
Overview

SnapMirror active sync enables individual SQL Server databases and applications to continue operations during storage and network disruptions, with transparent storage failover without any manual intervention.

Starting ONTAP 9.15.1, SnapMirror active sync supports symmetric active/active architecture in addition to the existing asymmetric configuration. Symmetric active/active capability provides synchronous bi-directional replication for business continuity and disaster recovery. It helps you protect your data access for critical SAN workloads with simultaneous read and write access to data across multiple failure domains, ensuring uninterrupted operations and minimizing downtime during disasters or system failures.

SQL Server hosts access storage using either Fiber Channel(FC) or iSCSI LUNs. Replication between each cluster hosting a copy of the replicated data. Since this feature is storage level replication, SQL Server instances running on standalone host or failover cluster instances can perform read/write operations either cluster. For planning and configuration steps, refer ONTAP documentation on SnapMirror active sync .

Architecture of SnapMirror active sync with symmetric active/active



Synchronous replication

In normal operation, each copy is an RPO=0 synchronous replica at all times, with one exception. If data cannot be replicated, ONTAP will release the requirement to replicate data and resume serving IO on one site while the LUNs on the other site are taken offline.

Storage hardware

Unlike other storage disaster recovery solutions, SnapMirror active sync offers asymmetric platform flexibility. The hardware at each site does not need to be identical. This capability allows you to right-size the hardware used to support SnapMirror active sync. The remote storage system can be identical to the primary site if it needs to support a full production workload, but if a disaster results in reduced I/O, than a smaller system at the remote site might be more cost-effective.

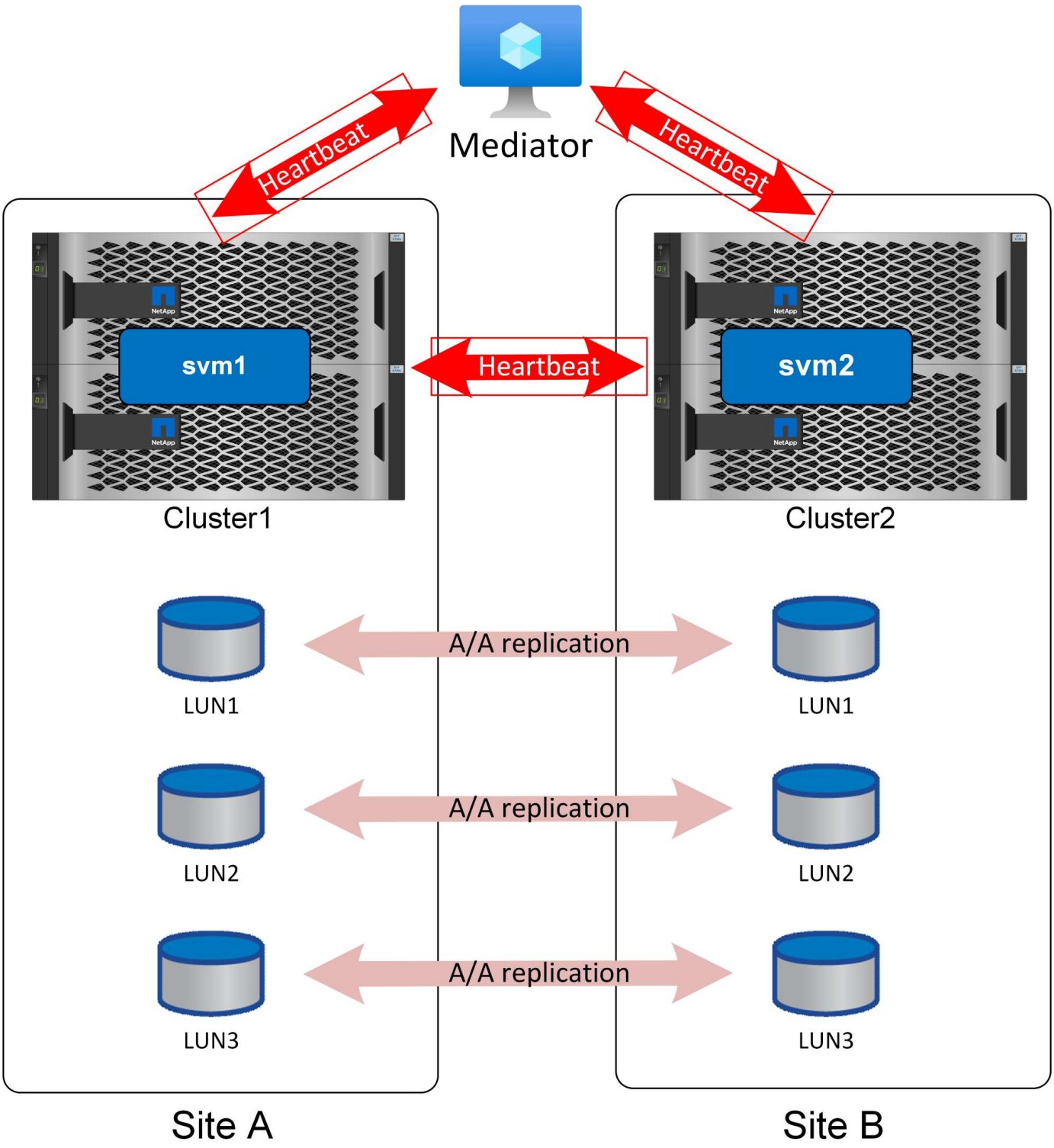
ONTAP Mediator

The ONTAP Mediator is a software application that is downloaded from NetApp support, and is typically deployed on a small virtual machine. The ONTAP Mediator is not a tiebreaker. It is an alternate communication channel for the two clusters that participate in SnapMirror active sync replication. Automated operations are driven by ONTAP based on the responses received from the partner via direct connections and via the mediator.

ONTAP mediator

The Mediator is required for safely automating failover. Ideally, it would be placed on an independent 3rd site, but it can still function for most needs if colocated with one of the clusters participating in replication.

The mediator is not really a tiebreaker, although that is effectively the function it provides. The mediator helps in determining the state of the cluster nodes and assists in the automatic switchover process in the event of a site failure. Mediator does not transfer data under any circumstances.



The #1 challenge with automated failover is the split-brain problem, and that problem arises if your two sites lose connectivity with each other. What should happen? You do not want to have two different sites designate themselves as the surviving copies of the data, but how can a single site tell the difference between actual loss of the opposite site and an inability to communicate with the opposite site?

This is where the mediator enters the picture. If placed on a 3rd site, and each site has a separate network connection to that site, then you have an additional path for each site to validate the health of the other. Look at the picture above again and consider the following scenarios.

- What happens if the mediator fails or is unreachable from one or both sites?
  - The two clusters can still communicate with each other over the same link used for replication services.
  - Data is still served with RPO=0 protection
- What happens if Site A fails?
  - Site B will see both of the communication channels go down.
  - Site B will take over data services, but without RPO=0 mirroring
- What happens if Site B fails?
  - Site A will see both of the communication channels go down.
  - Site A will take over data services, but without RPO=0 mirroring

There is one other scenario to consider: Loss of the data replication link. If the replication link between sites is lost, RPO=0 mirroring will obviously be impossible. What should happen then?

This is controlled by the preferred site status. In an SM-as relationship, one of the sites is secondary to the other. This has no effect on normal operations, and all data access is symmetric, but if replication is interrupted then the tie will have to be broken to resume operations. The result is the preferred site will continue operations without mirroring and the secondary site will halt IO processing until replication communication is restored.

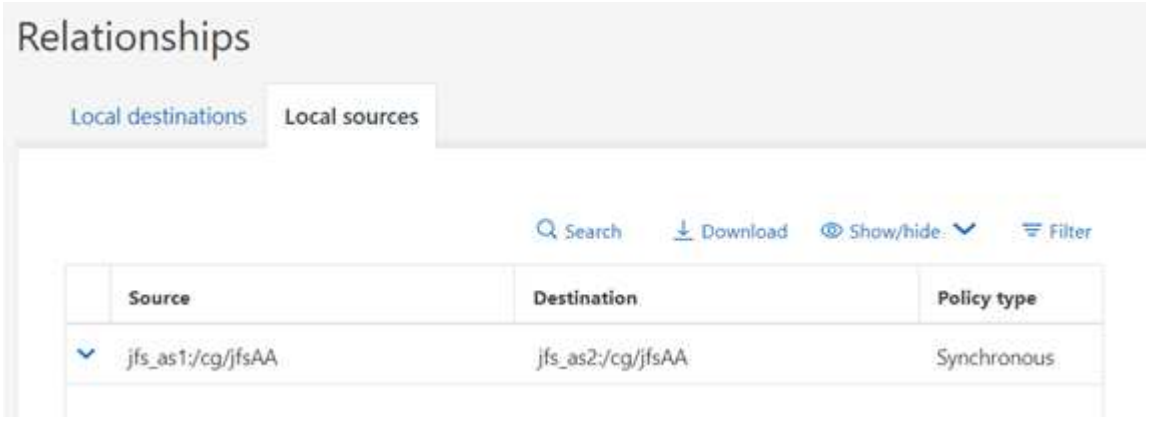
Preferred site

SnapMirror active sync behavior is symmetric, with one important exception - preferred site configuration.

SnapMirror active sync will consider one site the "source" and the other the "destination". This implies a one-way replication relationship, but this does not apply to IO behavior. Replication is bidirectional and symmetric and IO response times are the same on either side of the mirror.

The `source` designation is controls the preferred site. If the replication link is lost, the LUN paths on the source copy will continue to serve data while the LUN paths on the destination copy will become unavailable until replication is reestablished and SnapMirror reenters a synchronous state. The paths will then resume serving data.

The sourced/destination configuration can be viewed via SystemManager:



or at the CLI:

```
Cluster2::> snapmirror show -destination-path jfs_as2:/cg/jfsAA

                           Source Path: jfs_as1:/cg/jfsAA
                     Destination Path: jfs_as2:/cg/jfsAA
                    Relationship Type: XDP
                Relationship Group Type: consistencygroup
                    SnapMirror Schedule: -
                 SnapMirror Policy Type: automated-failover-duplex
                      SnapMirror Policy: AutomatedFailOverDuplex
                            Tries Limit: -
                      Throttle (KB/sec): -
                           Mirror State: Snapmirrored
                    Relationship Status: InSync
```

The key is that the source is the SVM on cluster1. As mentioned above, the terms "source" and "destination" don’t describe the flow of replicated data. Both sites can process a write and replicate it to the opposite site. In effect, both clusters are sources and destinations. The effect of designating one cluster as a source simply controls which cluster survives as a read-write storage system if the replication link is lost.

Network topology

Uniform access

Uniform access networking means hosts are able to access paths on both sites (or failure domains within the same site).

An important feature of SM-as is the ability to configure the storage systems to know where the hosts are located. When you map the LUNs to a given host, you can indicate whether or not they are proximal to a given storage system.

Proximity settings

Proximity refers to a per-cluster configuration that indicates a particular host WWN or iSCSI initiator ID belongs to a local host. It is a second, optional step for configuring LUN access.

The first step is the usual igroup configuration. Each LUN must be mapped to an igroup that contains the WWN/iSCSi IDs of the hosts that need access to that LUN. This controls which host has *access* to a LUN.

The second, optional step is to configure host proximity. This does not control access, it controls *priority*.

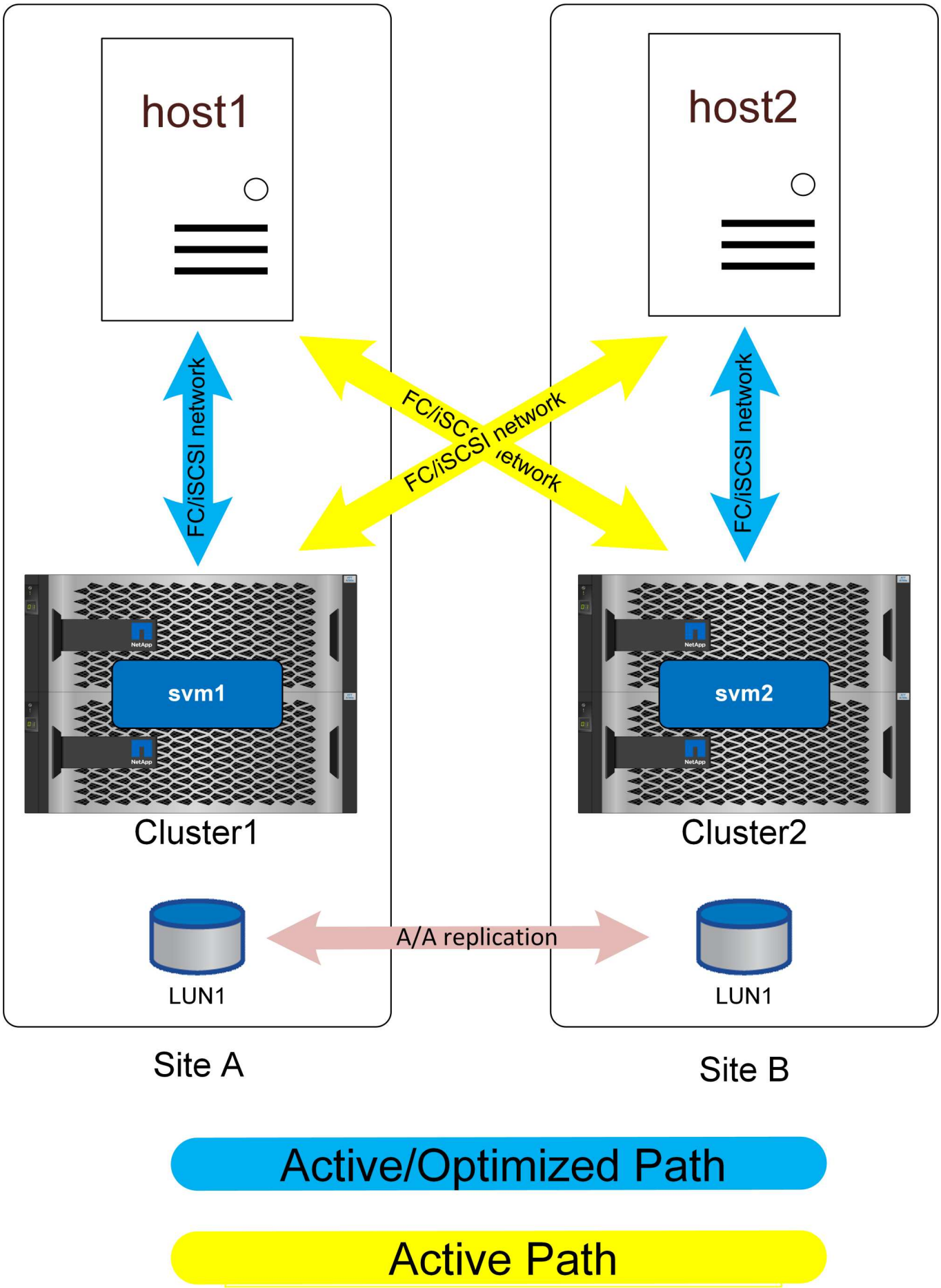
For example, a host at site A might be configured to access a LUN that is protected by SnapMirror active sync, and since the SAN is extended across sites, paths are available to that LUN using storage on site A or storage on site B.

Without proximity settings, that host will use both storage systems equally because both storage systems will advertise active/optimized paths. If the SAN latency and/or bandwidth between sites is limited, this may not be desireable, and you may wish to ensure that during normal operation each host preferentially uses paths to the local storage system. This is configured by adding the host WWN/iSCSI ID to the local cluster as a proximal

host. This can be done at the CLI or SystemManager.

AFF

With an AFF system, the paths would appear as shown below when host proximity has been configured.


host1
host2
FC/iSCSI network
FC/iSCSI network
FC/iSCSI network
FC/iSCSI network
svm1
svm2
Cluster1
Cluster2
A/A replication
LUN1
LUN1
Site A
Site B
Active/Optimized Path
Active Path

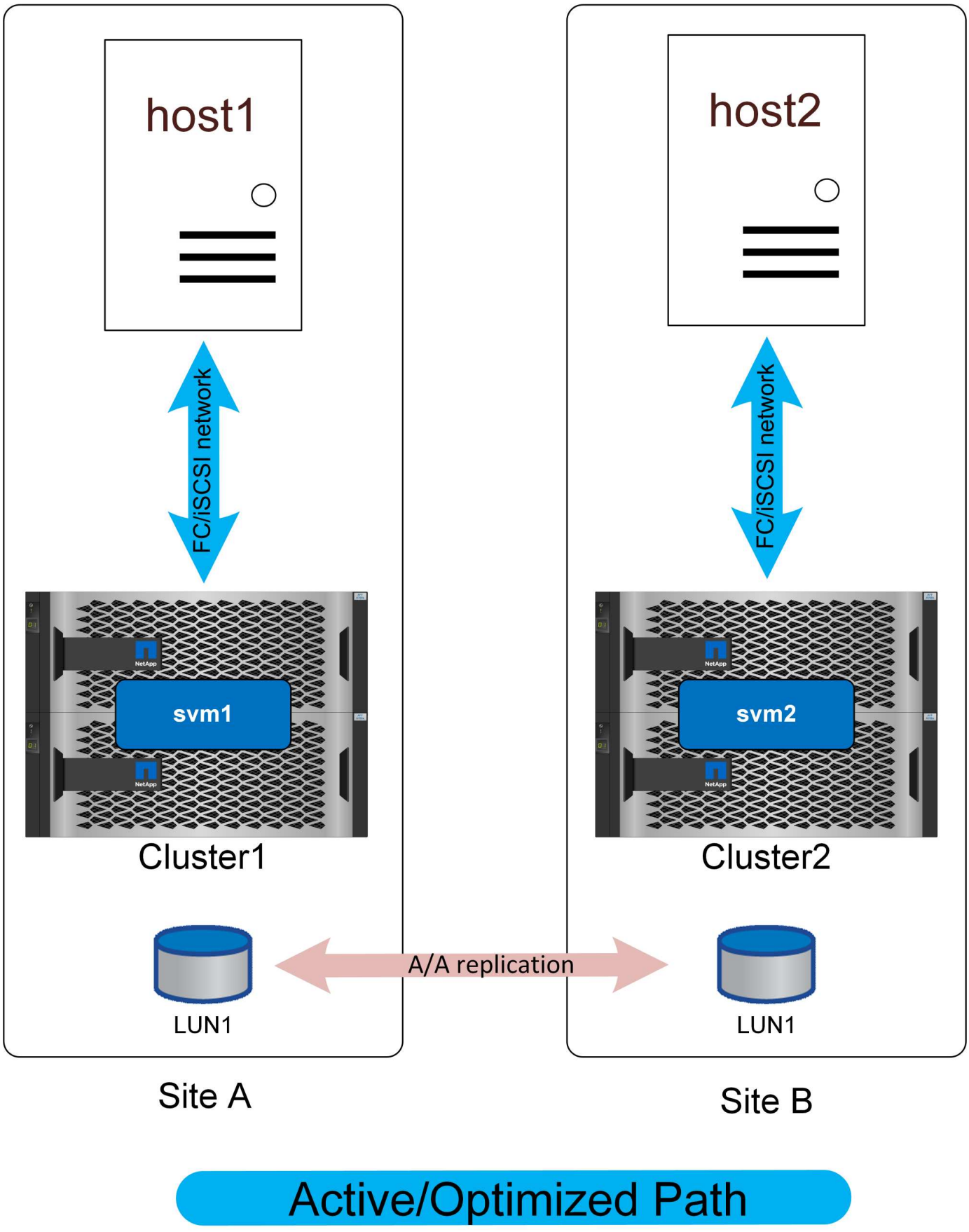
In normal operation, all IO is local IO. Reads and writes are serviced from the local storage array. Write IO will, of course, also need to be replicated by the local controller to the remote system before being acknowledged, but all read IO will be serviced locally and will not incur extra latency by traversing the SAN link between sites.

The only time the nonoptimized paths will be used is when all active/optimized paths are lost. For example, if the entire array on site A lost power, the hosts at site A would still be able to access paths to the array on site B and therefore remain operational, although they would be experiencing higher latency.

There are redundant paths through the local cluster that are not shown on these diagrams for the sake of simplicity. ONTAP storage systems are HA themselves, so a controller failure should not result in site failure. It should merely result in a change in which local paths are used on the affected site.

Nonuniform access

Nonuniform access networking means each host only has access to ports on the local storage system. The SAN is not extended across sites (or failure domains within the same site).



The primary benefit to this approach is SAN simplicity - you remove the need to stretch a SAN over the network. Some customers don't have sufficiently low-latency connectivity between sites or lack the

infrastructure to tunnel FC SAN traffic over an intersite network.

The disadvantage to nonuniform access is that certain failure scenarios, including loss of the replication link, will result some hosts losing access to storage. Applications that run as single instances, such as a non-clustered database that is inherently only running on a single host at any given mount would fail if local storage connectivity was lost. The data would still be protected, but the database server would no longer have access. It would need to be restarted on a remote site, preferably through an automated process. For example, VMware HA can detect an all-paths-down situation on one server and restart a VM on another server where paths are available.

In contrast, a clustered application such as Oracle RAC can deliver a service that is simultaneously available at two different sites. Losing a site doesn't mean loss of the application service as a whole. Instances are still available and running at the surviving site.

In many cases, the additional latency overhead of an application accessing storage across a site-to-site link would be unacceptable. This means that the improved availability of uniform networking is minimal, since loss of storage on a site would lead to the need to shut down services on that failed site anyway.

There are redundant paths through the local cluster that are not shown on these diagrams for the sake of simplicity. ONTAP storage systems are HA themselves, so a controller failure should not result in site failure. It should merely result in a change in which local paths are used on the affected site.

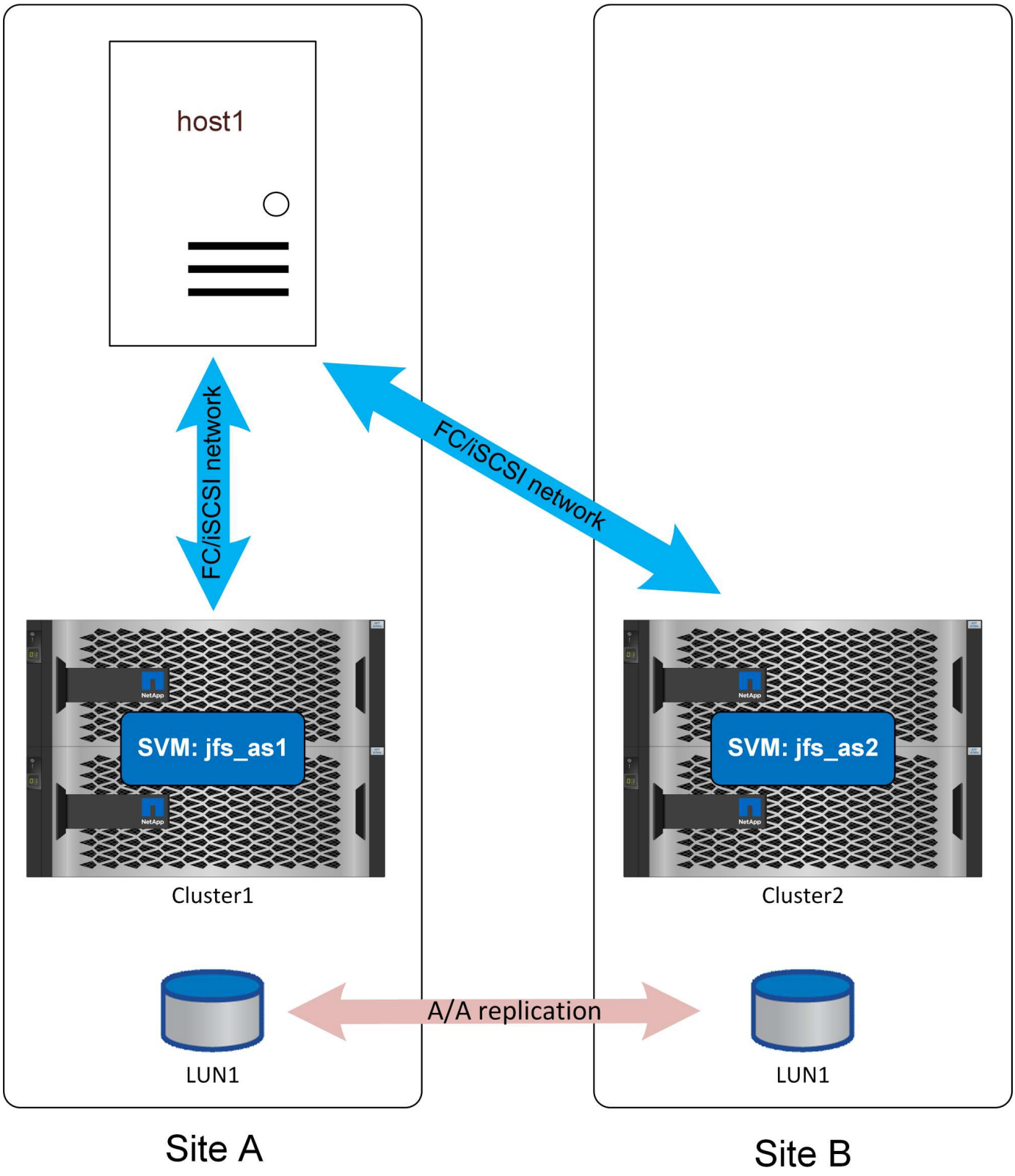
Overview

SQL Server can be configured to work with SnapMirror active sync in several ways. The right answer depends on the available network connectivity, RPO requirements, and availability requirements.

Standalone instance of SQL Server

The best practices for file layout and server configuration are the same as recommended in SQL Server on ONTAP documentation.

With a standalone setup, SQL Server could be running only at one site. Presumably uniform access would be used.



With uniform access, a storage failure at either site would not interrupt database operations. A complete site failure on the site that included the database server would, of course, result in an outage.

Some customers could configure an OS running at the remote site with a preconfigured SQL Server setup, updated with an equivalent build version as that of the production instance. Failover would require activating that standalone instance of SQL Server at the alternate site, discovering the LUNS, and and starting the

database. The complete process can be automated with the Windows Powershell cmdlet as no operations are required from the storage side.

Nonuniform access could also be used, but the result would be a database outage if the storage system where the database server was located had failed because the database would have no available paths to storage. This still may be acceptable in some cases. SnapMirror active sync would still be providing RPO=0 data protection, and, in the event of site failure, the surviving copy would be active and ready to resume operations using the same procedure used with uniform access as described above.

A simple and automated failover process can be more more easily configured with the use of a virtualize host. For example, if SQL Server data files are synchronously replicated to secondary storage along with a boot VMDK, then, in the event of a disaster, the complete environment could be activated at the alternate site. An administrator could either manually activate the host at the surviving site, or automate the process through a service such as VMware HA.

SQL Server failover cluster instance

SQL Server failover instances could be also hosted on a Windows failover cluster running on a physical server or virtual server as the guest operating system. This multi-host architecture provides SQL Server instance and storage resiliency. Such deployment is helpful in high-demand environments seeking robust failover processes while maintaining enhanced performance. In a failover cluster setup, when a host or primary storage is affected then SQL Services will be failover to the secondary host, and at the same time, secondary storage will be available to serve IO. No automation script or administrator intervention is required.

Failure scenarios

Planning a complete SnapMirror active sync application architecture requires understanding how SM-as will respond in various planned and unplanned failover scenarios.

For the following examples, assume that site A is configured as the preferred site.

Loss of replication connectivity

If SM-as replication is interrupted, write IO cannot be completed because it would be impossible for a cluster to replicate changes to the opposite site.

Site A (Preferred site)

The result of replication link failure on the preferred site will be an approximate 15 second pause in write IO processing as ONTAP retries replicated write operations before it determines that the replication link is genuinely unreachable. After the 15 seconds elapses, the site A system resumes read and write IO processing. The SAN paths will not change, and the LUNs will remain online.

Site B

Since site B is not the SnapMirror active sync preferred site, its LUN paths will become unavailable after about 15 seconds.

Storage system failure

The result of a storage system failure is nearly identical to the result of losing the replication link. The surviving site should experience a roughly 15 second IO pause. Once that 15 second period elapses, IO will resume on that site as usual.

Loss of the mediator

The mediator service does not directly control storage operations. It functions as an alternate control path between clusters. It exists primarily to automate failover without the risk of a split-brain scenario. In normal operation, each cluster is replicating changes to its partner, and each cluster therefore can verify that the partner cluster is online and serving data. If the replication link failed, replication would cease.

The reason a mediator is required for safe automated failover is because it would otherwise be impossible for a storage cluster to be able to determine whether loss of bidirectional communication was the result of a network outage or actual storage failure.

The mediator provides an alternate path for each cluster to verify the health of its partner. The scenarios are as follows:

- If a cluster can contact its partner directly, replication services are operational. No action required.
- If a preferred site cannot contact its partner directly or via the mediator, it will assume the partner is either actually unavailable or was isolated and has taken its LUN paths offline. The preferred site will then proceed to release the RPO=0 state and continue processing both read and write IO.
- If a non-preferred site cannot contact its partner directly, but can contact it via the mediator, it will take its paths offline and await the return of the replication connection.
- If a non-preferred site cannot contact its partner directly or via an operational mediator, it will assume the partner is either actually unavailable or was isolated and has taken its LUN paths offline. The non-preferred site will then proceed to release the RPO=0 state and continue processing both read and write IO. It will assume the role of the replication source and will become the new preferred site.

If the mediator is wholly unavailable:

- Failure of replication services for any reason, including failure of the nonpreferred site or storage system, will result in the preferred site releasing the RPO=0 state and resuming read and write IO processing. The non-preferred site will take its paths offline.
- Failure of the preferred site will result in an outage because the non-preferred site will be unable to verify that the opposite site is truly offline and therefore it would not be safe for the nonpreferred site to resume services.

Restoring services

After a failure is resolved, such as restoring site-to-site connectivity or powering on a failed system, the SnapMirror active sync endpoints will automatically detect the presence of a faulty replication relationship and bring it back to an RPO=0 state. Once synchronous replication is reestablished, the failed paths will come online again.

In many cases, clustered applications will automatically detect the return of failed paths, and those applications will also come back online. In other cases, a host-level SAN scan may be required, or applications may need to be brought back online manually. It depends on the application and how it is configured, and in general such tasks can be easily automated. ONTAP itself is self-healing and should not require any user intervention to resume RPO=0 storage operations.

Manual failover

Changing the preferred site requires a simple operation. IO will pause for a second or two as authority over replication behavior switches between clusters, but IO is otherwise unaffected.

Copyright information

Trademark information