EMC Unity Storage with Microsoft SQL Server

All-flash arrays

Abstract

Best practices guide for deploying Microsoft SQL Server with EMC Unity All-Flash arrays that includes recommendations and considerations for performance, availability, and scalability.

November 2017

Revisions

Date	Description
July 2017	Initial release for EMC Unity OE version 4.2
November 2017	Includes guidance for SQL Server on file (SMB) shares

Acknowledgements

This paper was produced by the following members of the EMC storage engineering team:

Author: Doug Bernhardt

Table of contents

Executive summary

This paper delivers straightforward guidance to customers using EMC™ Unity storage systems in a Microsoft® SQL Server® environment. SQL Server is an incredibly robust product that can be used in a variety of solutions. Therefore, the relative priorities of critical design goals such as performance, manageability, and flexibility depend on your environment. This paper provides important considerations and recommendations to help meet your design goals. For general best practices using EMC Unity see the *EMC Unity Best Practices Guide*.

These guidelines are intended to cover the majority of use cases. They are strongly recommended by EMC, however, they are not strictly required.

This paper was developed using the EMC Unity 650F All-Flash array, but is also applicable when using the 350F, 450F, or 550F EMC Unity All-Flash arrays.

If you have questions about the applicability of these guidelines in your environment, contact your EMC representative to discuss the appropriateness of the recommendations.

Audience

This document is intended for EMC Unity administrators, database administrators, architects, partners, and anyone responsible for configuring EMC Unity storage systems. Some familiarity with EMC unified storage systems is assumed.

We welcome your feedback along with any recommendations for improving this document. Send comments to StorageSolutionsFeedback@dell.com.

1 Storage configuration

EMC Unity is a virtually provisioned, flash optimized storage system designed for ease of use. This paper covers the all-flash array models. This section provides foundational array technologies that support the application-specific sections that follow. Additional information for this section can be found in the *EMC Unity Best Practices Guide*.

1.1 Storage pools

As of EMC Unity OE version 4.2, EMC Unity supports two different types of storage pools on all-flash storage systems: Traditional pools and dynamic pools. Traditional pools apply RAID protection to discrete groups of drives within the storage pool. Dynamic pools apply RAID to groups of drive extents from drives within the pool and allow for greater flexibility in managing and expanding the pool. Dynamic pools must be configured from all-flash drives; dynamic pools cannot be built with HDDs.

In general, it is recommended to use a small number of storage pools within EMC Unity to reduce complexity and increase flexibility. However, it may be appropriate to configure additional storage pools in order to:

- Separate workloads with different I/O profiles
- Dedicate resources to meet specific performance goals
- Separate resources for multi-tenancy
- Create smaller failure domains

Additional information can be found in the *EMC Unity: Dynamic Pools* white paper.

1.1.1 Storage pool capacity

Storage pool capacity is used for multiple purposes:

- Store all data written into storage objects (LUNs, file systems, and datastores, and VVols) in that pool
- Store data that is needed for snapshots of storage objects in the pool
- Track changes to replicated storage objects in that pool

Storage pools must maintain free capacity in order to operate properly. By default, EMC Unity will raise an alert if a storage pool has less than 30% free capacity, and will begin to automatically invalidate snapshots and replication sessions if the storage pool has less than 5% free capacity. EMC recommends that a storage pool always have at least 10% free capacity.

1.1.2 All-flash pool

All-flash pools provide the highest level of performance in EMC Unity. Use an all-flash pool when the application requires the highest storage performance at the lowest response time.

- EMC FAST™ Cache and FAST VP are not applicable to all-flash pools.
- Compression is only supported on an all-flash pool.
- Snapshots and replication operate most efficiently in all-flash pools.
- EMC recommends using only a single drive size and a single RAID width within an all-flash pool.

Example: For an all-flash pool, use only 1.6TB SAS flash 3 drives and configure them all with RAID 5 8+1.

1.1.3 Hybrid pool

Hybrid pools are not applicable when using EMC Unity All-Flash arrays.

2 EMC Unity features

This section describes some of the native features available on the EMC Unity platform. Additional information on each of these features can be found in the *EMC Unity Best Practices Guide*. Features not applicable to the all-flash array models covered by this paper are noted.

2.1 FAST VP

FAST VP accelerates performance of a specific storage pool by automatically moving data within that pool to the appropriate drive technology based on data access patterns. FAST VP is only applicable to hybrid pools within an EMC Unity Hybrid flash system.

2.2 FAST Cache

FAST Cache is a single global resource that can improve performance of one or more hybrid pools within an EMC Unity Hybrid flash system. FAST Cache can only be created with SAS flash 2 drives, and is only applicable to hybrid pools.

2.3 Compression

EMC Unity compression is available for block LUNs and VMFS datastores in an all-flash pool starting with EMC Unity OE version 4.1. Compression is available for file systems and NFS datastores in an all-flash pool starting with EMC Unity OE version 4.2.

Be aware that compression increases the overall CPU load on the system when storage objects service reads or writes of compressible data, and may increase latency when accessing the data. Before enabling compression on a storage object, it is recommended to monitor the system and ensure the system has available resources to support compression. (Refer to the “Hardware Capability Guidelines” section and Table 2 in the *EMC Unity Best Practices Guide*.) Enable Compression on a few storage objects at a time and then monitor the system to be sure it is still within recommended operating ranges, before enabling compression on more storage objects. Additional information regarding compression can be found in the *EMC Unity: Compression* white paper.

Compression will only provide space savings if the data on the storage object is at least 25% compressible. Before enabling compression on a storage object, determine if it contains data that will compress; do not enable compression on a storage object if there will be no space savings. Contact your EMC representative for tools that can analyze the data compressibility. Additional information regarding compression can be found in the *EMC Unity: Compression* white paper.

2.4 Data at Rest Encryption (D@RE)

D@RE is controller based encryption that does not impact system performance; therefore EMC recommends ordering EMC Unity systems as encryption-enabled, if appropriate for your environment.

Note: Encryption can only be enabled at the time of system installation with the appropriate license.

If encryption is enabled, EMC recommends making external backups of the encryption keys after system installation as well as immediately following any change in the system’s drive configuration (such as creating or expanding a storage pool, adding new drives or replacing a faulted drive). Additional information regarding Data at Rest Encryption can be found in the *EMC Unity: Data at Rest Encryption* white paper.

2.5 Host I/O limits

EMC recommends setting Host I/O limits on workloads that might monopolize pool resources and starve other applications of their required performance.

> Example: Limit the bandwidth available to large-block applications that may be increasing the latency on other small-block workloads.

Additional information can be found in the *EMC Unity: Unisphere Overview* white paper.

3 SQL Server design considerations

The I/O storage system is a critical component of any SQL Server environment. Sizing and configuring a storage system without understanding the I/O requirements can have disastrous consequences. Analyzing performance in an existing environment using a tool like the Dell Performance Analysis Collection Kit (DPACK) or Mitrend can help define the I/O requirements. For best results, capture performance statistics for a time period of at least 24 hours that includes the system peak workload.

3.1 OLTP workloads

While every environment is unique, an Online Transaction Processing (OLTP) workload typically consists of small, random reads and writes. A storage system services this type of workload is primarily sized based on capacity and the number of IOPS required.

3.2 OLAP/DSS workloads

An Online Analytic Processing (OLAP) or Decision Support System (DSS) workload is typically dominated by large, sequential reads. A storage system services this type of workload is primarily sized based on throughput. When designing for throughput, the performance of the entire path between the server and the drives in the EMC Unity array needs to be considered. For best throughput, consider using 16Gbps Fiber Channel (FC) or 10 Gbps iSCSI connectivity to the array, and 12 Gbps SAS connectivity from the controllers to the disk enclosures. To meet high throughput requirements, multiple HBAs may be required in the server, the array or both.

3.3 Mixed workloads

The most common scenario is a mixed workload environment. Typically, SQL Server I/O patterns do not strictly fall into an OLTP or OLAP pattern. This is what can make SQL Server workloads challenging, because no two workloads behave the same. In addition, the same SQL Server host and/or instance may be servicing multiple applications or transaction workloads.

Mixed workload can also imply that multiple applications (in addition to SQL Server) are residing on the same host and/or accessing the same storage. The combined workload of these applications invalidates any typical application I/O usage pattern. For these reasons, it is important to gather actual performance metrics for best sizing results.

3.4 Storage pools

In general, it is recommended to use fewer storage pools within EMC Unity because this reduces complexity and increases flexibility. EMC recommends using a single virtual disk pool when implementing SQL Server. This provides better performance by leveraging the aggregate I/O bandwidth of all disks to service I/O requests from SQL Server. A single drive pool is also easier to manage, allowing an administrator to quickly and easily adapt the storage system to satisfy the ever-changing workloads that are common in SQL Server environments. Before creating multiple storage pools to segregate workloads, understand the various EMC Unity features that are available for managing and throttling specific workloads.

3.4.1 RAID configurations

By default, EMC Unity chooses RAID 5 as the RAID protection level when creating a new storage pool. This contradicts traditional guidance advocating RAID 1/0 for database workloads. The issue with traditional guidance is that it assumes spinning disk and does not take into account SSD and storage systems such as EMC Unity that have been optimized for flash storage. Internal testing has shown that the performance difference in most configurations between RAID 5 and RAID 1/0 in EMC Unity All-Flash systems is negligible unless the workload is extremely write intensive for an extended period of time. In most cases, the small performance gain of RAID 1/0 is not worth the reduced capacity and therefore it is recommended to use the default configuration of RAID 5. For extremely heavy write workloads where maximum write performance is required, RAID 1/0 can be used.

3.5 Validating the storage design

Once the I/O requirements have been defined, it is easy to get a feel for whether the hardware can provide the desired performance by running some simple tests. Diskspd is a free Microsoft utility that can simulate I/O patterns generated by SQL Server. There are several other utilities available as well. When selecting a utility to simulate I/O, verify that it meets the following requirements:

- Ability to configure block size
- Ability to specify number of outstanding requests
- Ability to configure test file size
- Ability to configure number of threads
- Support for multiple test files
- Does not write blocks of zeros during tests

3.5.1 Validating the I/O path

The first thing to test on a new configuration is the path between the server and the array. Running a large block sequential read test using small files should saturate the path between the server and the array. This test verifies that all paths are fully functional and can be used for I/O traffic. Run this test on a dedicated server and array; a live system could cause significant performance issues.

To validate the I/O path, run a large block sequential read test using the following guidelines:

- Create one LUN per storage processor
- Format the volumes using a 64KB allocation unit
- Use a block size of 512KB for the test
- Configure the test for 32 outstanding I/Os
- Use multiple threads. Eight is the recommended starting point.

If the displayed throughput matches the expected throughput for the number of HBA ports in the server, the paths between the server and EMC Unity array are set up correctly.

3.5.2 Validating the drives

Once the I/O path has been validated, the next step is to test the drives. For best results when testing drives on a EMC Unity array, use the following guidelines when configuring the test.

- In a dual controller system, use at least one volume per controller. This ensures that I/O will be distributed across both controllers. Using both controllers more closely simulates real world activity. For best results, use the same number of volumes on each controller.
- When performing I/O tests on any storage platform, it is important to use files that are larger than the controller cache. For more accurate results, use a file size that matches the amount of data being stored. In an environment where that is not practical due to a large data set, use a file size of at least 100GB.
- Some I/O test tools, including Diskspd, SQLIO and IOMeter, generate files full of zeros. This behavior causes inaccurate results when testing with files containing only zeros. Avoid using test utilities that write zeros for drive validation. The contents of the test file can be verified by viewing the test file with a hex editor after different stages of a test. For example, create a small test file and view it after the initial creation, as well as after the test has run for a few seconds. If the file is filled with zeros, select another utility. Diskspd and IOMeter initially create test files filled with zeros, and then writes random characters when performing write tests. To properly initialize a Diskspd or IOMeter test file, run a sequential write test until the entire file has been overwritten with non-zero data. Unfortunately, SQLIO writes zeros during write tests and therefore is not recommended for drive validation.

The purpose of this type of testing is to validate that the storage design will provide the required throughput and IOPS with acceptable latency. It is important that the test does not exceed the designed capacity of the array. For example, an array designed for a workload of 5,000 IOPS is likely to perform poorly with a workload of 10,000 IOPS. If a test is generating a workload higher than the designed capacity, adjust the workload being generated by reducing the number of threads and/or outstanding I/Os.

The results of the DPACK analysis provide an I/O target to simulate using these tests. To get an idea of the performance capabilities of the array, run I/O tests with a range of I/O sizes commonly seen with SQL Server. When testing random I/O, test with an I/O size of 8KB and 64KB. When testing sequential I/O, start with I/O sizes of 8KB and 64KB. Since processes like read ahead scans and backups can issue much larger sequential I/O, it is a good idea to also test block sizes up to 1024KB.

4 Deploying Microsoft SQL Server on EMC Unity

Proper architecture and configuration of the SQL Server environment is critical to optimize performance and manageability of the EMC Unity and SQL Server environment. Apply the following best practices when designing, configuring, and managing SQL Server databases on EMC Unity.

4.1 Comparing traditional block versus file (SMB) storage

EMC Unity supports robust features for both block and file storage. Block storage is the traditional way storage is presented to SQL Server in a SAN environment. Starting with Microsoft SQL Server 2012, both system and user databases can be stored on SMB file shares. Since SQL Server is used in a variety of configurations and solutions, choosing block or file storage is based on several different factors. Although SQL Server has been deployed primarily on block storage, there are some scenarios where deploying SQL Server on file storage is an attractive option.

4.1.1 Performance

If maximum performance is a primary concern, traditional block storage is the typical choice. Block storage will provide lower latencies and better throughput than file storage. However, since SQL Server is used in a variety of applications and environments, there are many cases where block storage will provide acceptable performance as well.

4.1.2 Configuration

Configuring SMB file storage is a simple process. Once the SMB share is created, is it accessible to hosts without additional zoning or mapping tasks, which greatly simplifies the host configuration. This can be advantageous when automating installation tasks and/or reassigning databases to different instances as it reduces the number of steps required in the process. Many hosts can utilize the same share, so there may be little or no storage configuration required once the initial file shares are created. Block storage configuration tasks such as network zoning, mapping and formatting volumes, configuring iSCSI and MPIO are not required for file storage.

4.1.3 Flexibility versus manageability

Most options for block storage such as sizing, snapshots and replication are configured at the volume level and most options for file storage are configured at the file system level. Therefore, block storage provides more granularity and in turn more flexibility in configuring storage options for individual hosts when compared to file storage. However, if that flexibility is not needed, configuring these options at the file server level can be much simpler to manage.

4.1.4 Utilizing both block and file storage

In many environments, it may make sense to use both block and file storage for SQL Server, depending on the application. Block storage can be leveraged for its extreme performance in mission-critical environments while file storage can be used in development and test environments that are dynamic in nature and where performance is less critical. In addition, SMB file shares make a great backup target for SQL Server in all environments.

4.2 Block storage configuration

4.2.1 Creating volumes

EMC Unity storage is virtualized to take advantage of all the drives in the storage pool. Characteristics such as RAID level and storage tier can have a big impact on performance and are configured at the volume level. EMC Unity snapshots are also configured at the volume level. There are many different types of files that are part of a SQL Server instance. Those different types of data often have different performance and snapshot requirements. For performance sensitive applications, EMC recommends creating at least five volumes for an instance of SQL Server as shown in the following table.

Table 1 Volume provisioning recommendations

File type	Number of volumes	Typical performance requirements	Typical snapshot requirements
User DB data	At least 1 per instance	Lower performance may be acceptable	Frequent snapshots, same consistency group as log volume
User DB transaction log	At least 1 per instance	High performance required	Frequent snapshots, same consistency group as data volume(s)
Data root directory (includes system DBs)	1 per instance	Lower performance may be acceptable	Infrequent snapshots, independent schedule
Tempdb data and transaction log	1 per instance	High performance may be required	No snapshots
Native SQL Server backup	1 per instance	Lower performance may be acceptable	Snapshots optional, independent schedule
Memory-Optimized Filegroup (if used)	At least 1 per instance	High performance required	Frequent snapshots, same schedule as log volume

Contain databases that span multiple LUNs within a consistency group. This helps ensure that other features such as snapshots, replication and Thin Clones will be configured properly and maintain data consistency.

4.2.2 Performance considerations

When there is one group of databases that require high performance and another group that does not, consider creating a set of volumes for each group of databases, placing each set in its own Consistency Group. Even if there is only one tier of storage, this strategy will make it easier to adjust the storage configuration in the future.

Having multiple volumes has an additional benefit in dual controller systems. All I/O requests for a given volume are processed by the controller that owns the volume. While a volume can be owned by either controller, a volume is only owned by one controller at a time. Having many volumes makes it easier to distribute the I/O load evenly across both controllers. Databases that have very high performance requirements, can be spread across two or more data files on separate volumes to leverage resources on both controllers.

4.2.3 Flexibility versus manageability

For ultimate flexibility, create a volume for each user database file. This provides the ability to independently optimize the storage and snapshot configuration for each individual database. With thin provisioning, there is no space penalty for creating a lot of volumes. However, a large number of volumes can be difficult to manage, especially in virtualized environments. It is up to the DBA and/or storage administrator to find the right balance between flexibility and maintainability when determining the number of volumes to create. Virtualized SQL Server environments are a good example of where it may make sense to place multiple file types on a single volume. Understanding the database I/O patterns is critical to making the best decisions.

4.2.4 Windows setup and configuration

4.2.4.1 Allocation unit size

Use a 64KB allocation unit size when formatting volumes that will contain database files (transaction log and data) or database backups.

4.2.4.2 MPIO

Set the MPIO policy to Round Robin for all database volumes. This is the default for Windows 2008 and newer. It allows all paths to be used, enabling higher throughput between the server and the array. This setting works best for most environments as it is easy to manage and performs very well.

For database servers with a large number of I/O ports, the overhead of the Round Robin MPIO policy can reduce the maximum throughput to the storage array. To maximize throughput, create a volume for each port on the server and use the Fail Over Only MPIO policy to define a single, unique, active path for each volume. Define all other paths as standby. By using a data file of the same size on each volume, the data will be evenly distributed across the volumes. This strategy is more complex to setup and maintain. Use it only in environments that require maximum throughput.

4.2.4.3 Partition alignment

An offset of 64KB, or a multiple of 64KB, is recommended for volumes created on Windows 2003 and earlier. New volumes created on Windows Server 2008 or newer should already be aligned since 1024KB is the default offset.

4.3 File storage configuration

Creating SMB file shares for SQL Server deployment is a simple process that involves three basic steps. These steps include creating a NAS server, the file system, and finally the SMB share. EMC Unisphere contains setup wizards that walk through the entire process. However, there are some important considerations to keep in mind for SQL Server workloads. These recommendations are specific to hosting SQL Server databases on EMC Unity and do not necessarily apply to using SMB file shares as backup files.

4.3.1 NAS server

The configuration of the NAS server has the greatest impact on the performance of the SMB file shares. When configuring the NAS server, this is where the storage pool and storage processor are selected. Since the NAS server resides on the selected storage processor, balancing NAS servers across both storage processors is important for efficient system utilization and best performance. A NAS server can only use resources on the storage processor that it is running on. Therefore, for a workload to use the resources of both storage processors, a minimum of two NAS servers would need to be created and the workload could be balanced across them.

Network interfaces are configured on the NAS server and are important to performance and availability. Multiple network interfaces are available for performance and redundancy. These interfaces map to a specific I/O port and multiple NAS servers can share the same port.

4.3.2 File system

When configuring the file system for SQL Server make sure both the Sync Writes Enabled and Oplocks Enabled settings are turned on. These SMB Protocol settings are found under the advanced file system properties. These are the only two SMB protocol settings that are recommended for use with SQL Server.

4.3.3 SMB shares

It is important to enable the Continuous Availability option in the SMB properties when configuring a SMB share for SQL Server. This option, along with the SMB protocol settings Sync Writes Enabled and Oplocks Enabled ensures that the file handles used by SQL Server will persist in the event of a NAS server failover to the other storage processor.

4.3.4 VMware datastore

When creating file-based VMware datastores for SQL Server on file based systems, select SQL Server for the host IO size. This will result in a 64KB block size, which is recommended for SQL Server.

4.4 Reducing SQL Server I/O

4.4.1 Memory

Unnecessary I/O can be avoided and performance can be increased by allocating the proper amount of memory to SQL Server. SQL Server performs all I/O through the buffer pool (cache) and therefore uses a large portion of its memory allocation for the buffer pool. Ideally, when SQL Server performs I/O, the data is already in the buffer pool and it does not need to go to disk. This type of I/O is referred to as logical I/O and is the most desirable because it results in the best performance. If the SQL Server data does not need to reside in the buffer pool, it will need to access disk resulting in physical I/O.

Proper memory allocation is critical to SQL Server performance and can improve storage performance as well. In many cases, SQL Server and storage performance can be further improved by adding additional memory. Generally speaking, the more memory the better, but there is a point of diminishing returns that is unique to each environment.

4.4.2 Buffer pool extension

With SQL Server 2014, the buffer pool can be extended to a file on the file system to provide additional space to cache data or index pages. Using this feature can provide significant performance benefits without adding memory to the database server in some cases. By caching more pages on the server, the I/O load on the array is reduced.

When placing the buffer pool extension on the array, create a separate volume for the buffer pool extension and do not take snapshots of the buffer pool extension volume. The buffer pool data is repopulated by SQL Server when the instance is restarted therefore data recovery does not apply.

4.4.3 Database compression

The overall I/O workload can be reduced by enabling database compression in SQL Server. While there is a tradeoff in terms of CPU utilization on the database server, it is still a viable option to consider and test in any environment. Database compression reduces I/O by reducing the amount of data that needs to be stored. The SQL Server data pages are compressed in memory before being written to disk resulting in fewer pages needed to store the same number of rows and therefore less I/O.

4.4.4 Instant file initialization

By default, SQL Server writes zeros to the data file during the allocation process. The process of zeroing out the data files consumes I/O and acquires locks as the SQL Server data pages are written. This activity can occur for minutes or even hours depending on the file size. While this may seem minor, writing zeros to these files can occur at critical periods when time and performance are critical such as database auto growth, expanding a full data file, replication, or restoring a database as part of a disaster recovery event.

When Instant File Initialization is enabled, SQL Server will skip the process of zeroing out its data files when allocating space. EMC recommends enabling Instant File Initialization.

4.4.5 Resource Governor

The Resource Governor was added in SQL Server 2008 to allow database administrators to limit the CPU and memory resources a query is able to consume. This feature was enhanced in SQL Server 2014 to allow I/O resources to be limited as well. For example, the Resource Governor can be used to reduce the impact of a user running an I/O intensive report by limiting the maximum number of IOPS that user can perform. While a query throttled by the Resource Governor will take more time to complete, overall database performance will be better.

4.4.6 Database design considerations

Reducing SQL Server I/O requires a holistic approach. Many of the items in this section will require involvement from the whole team responsible for the SQL Server applications including the business owner, architect, developer, database administrator, and system administrator. Decisions at the design level have a multiplied downstream impact as data is written and read multiple times and duplicated in various types of database copies including databases copied for other uses such as testing and reporting, replicated databases, replicated storage and backups.

One of the most challenging aspects of SQL Server is that the I/O pattern and the amount of I/O that is generated can vary greatly depending on the application, even if those applications have databases of the same size. This is because the design of both the database and the data access code control SQL Server I/O.

Database tuning can be one of the most cost effective ways to reduce I/O and improve scalability. At a high level, consider the following areas when tuning a database to reduce I/O.

4.4.6.1 Database design

The foundation of the entire database and the schema for how data will be stored and ultimately accessed is determined by the database design. The database design should support both usability and efficient data access. This includes efficient table design and data types as well as indexes, partitioning, and other features that can improve efficiency. It is common for database design to only be focused on usability while performance and scale are overlooked.

4.4.6.2 Query design

How a query is written can greatly affect the amount of I/O SQL Server needs to perform when executing the query. Queries should return only the required amount of data in the most efficient manner possible. Tune the queries responsible for consuming a relatively large number of resources as well as possible for best performance and scale.

4.4.6.3 Application design

Consider how applications are using the data and the manner in which it is requested. Sometimes code and component reuse can result in the same data being unnecessarily retrieved over and over again. All data access should be purposeful.

4.4.6.4 Maintenance

SQL Server uses a cost based optimizer to generate query plans for data access. These plans are based on the statistics regarding how data is distributed in the tables. If the statistics are inaccurate, bad query plans may result and unnecessary I/O will be performed. Proper database maintenance includes ensuring that statistics are up to date.

Frequent data modifications can also lead to fragmentation within SQL Server data files, producing unnecessary I/O. Fragmentation can be addressed through index reorganization or rebuilds as part of regular database maintenance.

The database maintenance process itself can also have a large I/O impact. Typically, every table and index does not need to be rebuilt or reorganized every time maintenance is run. In addition, table partitioning strategies can also be leveraged to make the maintenance process more selective. Consider implementing intelligent maintenance scripts that utilize usage statistics to perform maintenance on an as-needed basis.

For mission critical databases, maintenance activities need to be considered as part of the overall design. If maintenance is not considered as part of the overall process, issues can arise, such as unmanageable sizes and feature incompatibilities that limit available options and strategies.

5 EMC Unity features with SQL Server

There are several features included in EMC Unity that when used in a SQL Server environment warrant additional discussion. The following sections provide some best practices on these features and their integration with SQL Server.

5.1 Compression

Since both EMC Unity and Microsoft SQL Server offer data compression, there are several factors to consider. There is no single recommendation since the best choice will depend on several factors such as the contents of the database, the amount of available CPU on both the storage and the SQL Server hosts, and the amount of I/O resources.

EMC Unity offers in-line compression for all-flash storage pools. The actual compression ratio will depend completely on the type of data being stored within the database and the array will only apply compression if there is sufficient savings. It is unlikely that this will be achieved on files that are already compressed. Therefore, it is recommended that compression be applied by either the array or the database engine, not both.

Compression requires CPU resources regardless of whether it is done within SQL Server or on the array and is another factor to consider when making a decision. Enabling SQL Server compression will increase the CPU load on the SQL Server host but will result in I/O savings on the storage array.

5.2 Snapshots

Snapshots are a fast, and space efficient way to protect SQL Server databases. When using snapshots with SQL Server databases there are some important considerations to ensure a successful database recovery.

All components of a SQL Server database must be protected as a set. When data and log files are located on separate LUNs, these LUNs need to be part of a Consistency Group. The Consistency Group will ensure that the Snapshot is taken at the exact same time on all LUNs in that group. When data and log files are located on multiple SMB file shares, the shares need to reside on the same file system.

When performing a SQL Server database recovery from a block-based snapshot, if the SQL Server instance is to remain online the Attach to Host action in Unisphere is recommended. For file-based recovery, an additional SMB share is created using the snapshot as the source. Once the volumes are attached, the database can be attached under a different name or used to replace the existing one.

When performing a recovery using the Snapshot Restore method in Unisphere, bring the SQL Server instance offline. SQL Server is unaware of the underlying recovery operations. Taking the instance offline will ensure the volumes are not corrupted by database write activity prior to doing a recovery. Once the instance is restarted, SQL Server crash recovery will run and bring the databases to a consistent state.

5.3 Thin Clones

Thin Clones are based on snapshot technology and a great way to present a read-write copy of databases. They are the preferred way to utilize snapshots for presenting database copies because they have many of the same data services as a regular LUN. Thin Clones are a great way to present SQL Server database copies in a fast, and space efficient manner that can’t be performed with traditional SQL Server tools. After the Thin Clone is presented to the host, the volumes can be brought online and the database is attached using the attach database method in SQL Server.

When using the Refresh feature with Thin Clones, make sure to take all databases offline that are contained on the Thin Clone prior to performing the refresh operation. SQL Server is unaware of the Thin Clone based LUNs being reverted. Failure to offline databases prior to this operation can result in data inconsistency errors and/or incorrect data results from SQL Server.

5.4 Replication

Creating a high availability solution for SQL Server often involves creating a copy of the data on another storage device and synchronizing that data in some manner. EMC Unity replication provides data synchronization between EMC Unity Systems. Data is replicated at the consistency group or at the LUN level allowing a choice of replication settings on a per volume basis. Using EMC Unity Replication can be an effective way to protect SQL Server databases due to the flexibility and configuration options that it provides. The variety of options provide a robust way to develop a replication scheme that provides the proper mix of performance and bandwidth efficiency while still meeting RTO and RPO requirements.

SQL Server databases consist of a set of one or more data files and one or more transaction log files that can be distributed across one or more volumes. When using EMC Unity replication to protect SQL Server databases that are located on multiple volumes, contain all data and log volumes for a database within a single consistency group or file system. Replication is then configured on the consistency group or file system which can contain volumes or shares for more than one database. Databases that require different replication settings will need to be on separate LUNs, consistency groups, or file systems.

When replicating volumes for SQL Server database protection, only replicate the volumes required for the user databases and ensure only those database files exist on replicated volumes. In addition, there is no need to replicate the tempdb database as it is used for temporary workspace only and is rebuilt when SQL Server is restarted. Carefully selecting which volumes to replicate and the contents of those volumes will eliminate unnecessary replication traffic.

In addition to EMC Unity replication features, SQL Server also contains various methods of data replication such as its own replication types and Always On Availability Group features as well as asynchronous and synchronous modes.

5.5 Data at Rest Encryption (D@RE)

Many SQL Server applications have data encryption requirements, specifically on data at rest. Data at Rest Encryption or D@RE can be used as an encryption solution for SQL Server without requiring any database or application changes. This also avoids any potential performance impact to the database server or the applications and has no performance impact on the array.

Note: D@RE is a license-able feature and must be selected during the ordering processes and licensed at system initialization. D@RE cannot be enabled at a later time.

5.6 Host I/O Limits (QoS)

Typically a EMC Unity array is used to service multiple hosts and applications. These applications can have different service levels and different storage demands. In addition, a single array can be providing services to multiple environments such as development and testing as well as production. Traditionally, these scenarios have been difficult to manage to ensure that critical applications get the resources they need while managing less critical resources to make sure they don’t over-consume.

Host I/O limits or QoS (quality of service) provide an excellent means to manage these types of workloads. Instead of trying to manage these workloads with multiple storage pools, Host I/O limits allow LUNs to be restricted to a specified amount of IOPS or Bandwidth so they don’t adversely impact other applications. Also, this allows storage administrators to ensure applications and environments adhere to budgeted limits which greatly simplifies planning and management.

Host I/O Limits are great for SQL Server environments for several reasons. First, storage administrators can ensure that demanding SQL Server instances don’t overwhelm the entire array by setting limits on database volumes. Also, if SQL Server is the priority application they can set limits on other LUNs on the system to ensure that SQL Server can get the resources it requires. Another great component of Host I/O Limits is the ability to burst for a given limit for a specific period of time, both are which are user-configurable. In this way, small exceptions can still be allowed while maintaining balanced performance.

In development and testing environments, it can be difficult to determine if an application meets performance requirements. Typically, test and development databases are smaller than production databases and it’s not always feasible to keep a copy of production data in these environments due to the storage costs or privacy concerns surrounding confidential data. The issue with smaller datasets is that the application can run much faster on a smaller database during development or testing phases, then encounter serious performance issues when deployed on a real dataset in production.

Host I/O Limits can be used to restrict the I/O on smaller datasets to highlight I/O intensive queries. Setting limits on databases in development and testing environments will help identify problem areas so they can be resolved prior to production deployment. The end result is improved SQL Server service levels and greater scalability.

6 Data protection

This section covers features or products used for data protection. It also covers how application data protection features may be used or are integrated. These items would be actionable for someone in either the storage administrator or application administrator role.

6.1 AppSync

EMC AppSync™ is software that enables integrated Copy Data Management (iCDM) with the EMC primary storage systems, including EMC Unity arrays.

AppSync simplifies and automates the process of creating and using snapshots of production data. By abstracting the underlying storage and replication technologies, and through application integration, AppSync empowers application owners to manage data copy needs themselves. The storage administrator, in turn, need only be concerned with initial setup and policy management, resulting in a more agile environment. Additional information on AppSync can be found in the *AppSync User and Administration Guide* and the *AppSync Performance and Scalability Guidelines*.

6.2 Snapshots

Be aware that snapshots increase the overall CPU load on the system, and increase the overall drive IOPS in the storage pool. Snapshots also use pool capacity to store the older data being tracked by the snapshot, which increases the amount of capacity used in the pool until the snapshot is deleted. Consider the overhead of snapshots when planning both performance and capacity requirements for the storage pool.

Before enabling snapshots on a storage object, it is recommended to monitor the system and ensure that existing resources can meet the additional workload requirements. (Refer to the “Hardware Capability Guidelines” section and Table 2 in the *EMC Unity Best Practices Guide*.) Enable snapshots on a few storage objects at a time, and then monitor the system to be sure it is still within the recommended operating ranges before enabling more snapshots. Additional information can be found in the *EMC Unity: Snapshots and Thin Clones* white paper.

6.3 Asynchronous replication

EMC recommends including a flash tier in a hybrid pool where asynchronous replication will be active. This is applicable to both the source and the destination pools.

Asynchronous replication takes snapshots on the replicated storage objects in order to create the point-in-time copy, determining the changed data to transfer and maintain consistency during the transfer. Consider the overhead of snapshots when planning performance and capacity requirements for a storage pool that will have replication objects.

Setting smaller RPO values on replication sessions will not make them transfer data more quickly, but smaller RPOs will result in more snapshot operations. Choosing larger RPOs, or manually synchronizing during non-production hours, may provide more predictable levels of performance. Additional information can be found in the *EMC Unity: Replication Technologies* white paper.

6.4 Synchronous replication

EMC recommends including a flash tier in a hybrid pool where synchronous replication will be active.

Synchronous replication transfers data to the remote system over the first Fibre Channel port on each SP. When planning to use synchronous replication, it may be appropriate to reduce the number of host connections on this port. When the CNA ports are configured as FC, CNA port 4 is defined as the synchronous replication port. If the CNA ports are configured as 10GbE, then port 0 of the lowest numbered FC I/O Module is the replication port. Additional information can be found in the *EMC Unity: Replication Technologies* white paper.

6.5 RecoverPoint virtual edition

EMC RecoverPoint virtual edition provides continuous data protection with multiple recovery points to restore applications instantly to a specific point in time. RecoverPoint virtual edition consists of RecoverPoint Appliance (RPA) software deployed as a virtual appliance in an existing VMware ESXi VM environment. RecoverPoint virtual edition is a flexible deployment option which offers maximum simplicity with no dependency on a physical appliance, lowering TCO.

A Technical support and resources

The EMC Data Center Transformation site is focused on enabling customers to gain understanding regarding the transformation of their IT infrastructure utilizing new technologies.

EMC Unity Info Hub is a source providing helpful links to document and tools to customers.

Hardware and software support for the EMC Unity platform is found at EMC Support.

A.1 EMC resources

EMC Publications:

- EMC Unity Best Practices Guide

  https://www.emc.com/collateral/white-papers/h15093-dell-emc-unity-best-practices-guide.pdf

- EMC Unity NAS Capabilities

  https://www.emc.com/collateral/white-papers/h15572-dell-emc-unity-nas-capabilities.pdf

- EMC titles on EMC Online Support:
  - EMC Unity: Dynamic Pools
  - EMC Unity: Compression
  - EMC Unity: Performance Metrics
  - EMC Unity: Replication Technologies
  - EMC Unity: Snapshots and Thin Clones
  - EMC Unity: Unisphere Overview

A.2 SQL Server resources

There is a tremendous amount of SQL Server information available online. While not a complete, the list below contains several useful links. Additional information can be found by using any Internet search engine.

- SQL Server I/O Basics

  http://technet.microsoft.com/en-us/library/cc966500.aspx

- SQL Server I/O Basics, Chapter 2

  http://technet.microsoft.com/en-us/library/cc917726.aspx

- Pre-deployment I/O Best Practices

  http://technet.microsoft.com/en-us/library/cc966412.aspx

- Analyzing I/O Characteristics and Sizing Storage Systems for SQL Server Database Applications

  http://msdn.microsoft.com/en-us/library/ee410782(v=SQL.100).aspx

- Dell SQL Server Solutions

http://www.dell.com/sql

- Microsoft SQL Server homepage

http://www.microsoft.com/sql

- SQL Server Customer Advisory Team

https://blogs.msdn.microsoft.com/sqlcat

- Microsoft online SQL Server forum

http://social.msdn.microsoft.com/Forums/en-US/category/sqlserver

- Professional Association for SQL Server

http://sqlpass.org

- Online SQL Server resources

http://www.sqlservercentral.com

http://www.sqlteam.com

http://www.sql-server-performance.com