SIMPLIFY AND ACCELERATE MICROSOFT SQL SERVER APPLICATION LIFECYCLE MANAGEMENT

EMC XtremIO integrated copy data management increases DBA productivity and accelerates lifecycle management

ABSTRACT

This white paper introduces the EMC XtremIO integrated copy data management (iCDM) technology stack and addresses common considerations for copy data management. The paper also discusses use cases and examples of using iCDM throughout the Microsoft SQL Server database application lifecycle to increase DBA productivity and accelerate lifecycle management.

March, 2016
TABLE OF CONTENTS

EXECUTIVE SUMMARY ........................................................................................................... 5
AUDIENCE .................................................................................................................................. 5

INTRODUCTION ........................................................................................................................... 5
EMC APPSYNC ............................................................................................................................. 6
EMC XTREMIO VIRTUAL COPY ..................................................................................................... 6
  PERFORMANCE ......................................................................................................................... 6
  STORAGE EFFICIENCY ............................................................................................................... 7
  INITIAL XVC CREATION ........................................................................................................... 7
  POST XVC CREATION .............................................................................................................. 8

ENHANCING DATABASE LIFECYCLE MANAGEMENT: USE CASE STUDIES ......................... 10
USE CASE I: TEST AND DEVELOPMENT .................................................................................... 10
USE CASE II: NEAR REAL-TIME ANALYTICS ............................................................................ 10
USE CASE III: DATABASE MAINTENANCE .................................................................................. 11
USE CASE IV: QUERY TUNING ................................................................................................... 11
USE CASE V: BACKUP ................................................................................................................ 11
USE CASE VI: ALWAYSON AVAILABILITY GROUPS ................................................................. 12
USE CASE VII: UPGRADE ......................................................................................................... 13

COMMON CONSIDERATIONS FOR COPY MANAGEMENT ..................................................... 13
APPLICATION-CONSISTENT VERSUS CRASH-CONSISTENT COPY ......................................... 13
  CRASH-CONSISTENT COPY .................................................................................................... 13
  APPLICATION-CONSISTENT COPY ........................................................................................ 13
  DETERMINING WHEN TO USE WHICH TYPE OF COPY ......................................................... 14
REFRESHING STALE COPIES ...................................................................................................... 15
DATA MASKING .......................................................................................................................... 17
IO RESOURCE GOVERNING ...................................................................................................... 18
  SQL SERVER RESOURCE GOVERNOR .................................................................................... 18
  VMWARE VSPHERE STORAGE TO CONTROL ....................................................................... 19
IMPACTS TO LOG TRUNCATION ............................................................................................... 20
  APPLICATION-CONSISTENT COPY ......................................................................................... 20
  CRASH-CONSISTENT COPY .................................................................................................. 21
LICENSING .................................................................................................................................. 21
EXECUTIVE SUMMARY

Throughout the Microsoft SQL Server database application lifecycle, copies of databases are not only needed for the purpose of data protection, but are also valuable for application features development, upgrades, database tuning, and effective testing. Database copies also can be used to reduce downtime during SQL Server service pack or version upgrades and to enable near real-time reporting on production data. High availability solutions such as AlwaysOn Availability Groups also require copies of the database for initial deployment. While copies are essential to SQL Server database application lifecycle management, the operation and costs for creating copies with traditional copy data management technologies can lead to serious issues. These issues include:

- Slow application development and lengthy test cycles due to inferior performance, long brute-force copy process, and outdated copies
- Application code that is not tested at scale prior to deployment because there are not enough copies or because copies are underperforming or go out of date quickly and can’t be created/regenerated at a pace required by the developer
- Inadequate analytics and reporting due to fixed and slow ETL processes that are never on-demand and always work old data
- Excess time spent by IT and Engineering on operations and risk reduction at the expense of innovation
- Higher infrastructure and administration costs due to excessive storage requirements

To address these issues, EMC® XtremIO® all-flash arrays feature integrated copy data management (iCDM). iCDM provides efficient copy data management at the storage layer to consolidate both primary data and its associated copies on the same scale-out, all-flash array for unprecedented agility and efficiency. When combined with XtremIO bullet-proof, consistent IOPS and latency, linear scale-out all-flash performance, and the ability to add more performance and capacity as needed with no application downtime, iCDM delivers incredible potential to consolidate both production and non-production applications without impacting production SLAs.

AUDIENCE

This paper is intended for:

- SQL Server database administrators (DBAs)
- IT administrators
- Storage/data center architects
- Technical managers

It is also intended for other IT personnel who are responsible for design, deployment, and management of SQL Server databases, infrastructure, and data center.

INTRODUCTION

Building on a unique performance foundation, XtremIO arrays leverage XtremIO Virtual Copy, or XVC technology. XVC abstracts the copy operations as a unique in-memory metadata operation with no back-end media or IO performance impact. XVC enables instant, high-performance copies of any data set in nearly any quantity desired, in a way that is entirely space efficient, with inline data services like deduplication and compression, and with no impact on production or other copies.
**Figure 1. XtremIO iCDM**

*Figure 1* shows the different layers of XtremIO iCDM. On top of XtremIO performance architecture and XVC capability, XtremIO iCDM offers integration into rich application workflows. These include creating a development copy, instantly refreshing it to the latest production data, pushing the development copy to a QA host, pushing the QA copy to a scalability testbed, and rolling the output back into production.

For analytics processes, production data can be extracted and pushed to all downstream analytics applications on-demand instantly as a simple in-memory operation. With XtremIO iCDM, you can deploy SQL Server database copies in any physical infrastructure or virtualized infrastructure, such as VMware® or Hyper-V, in seconds with centralized management. Copies are immediately usable by the application, enabling fast deployment and ensuring that those deployed copies are fully functional. Rounding out the XtremIO iCDM stack is the self-service copy management tool, EMC AppSync®. AppSync provides the ability for application, database, and development owners to self-service manage the generation and provisioning of copies—reducing demand on infrastructure management and greatly enhancing business agility.

**EMC APPSYNC**

EMC AppSync offers a simple, SLA-driven, self-service approach for protecting, restoring, and repurposing critical SQL Server databases. After defining service plans, application owners can protect, restore, and repurpose production databases quickly. AppSync also provides an application protection monitoring service that generates alerts when the SLAs are not met.

AppSync components include the AppSync server, agent (host plug-in software), and user interfaces (UI or console). AppSync also provides a REST interface that allows application programmers to access information controlled by AppSync.

**EMC XTREMIO VIRTUAL COPY**

**PERFORMANCE**

XVC shares the same metadata and physical data blocks with the production source volume on initial copy creation. With the unique redirect-on-unique-write technology, changes to the production source volume or XVC volume are tracked with separate metadata entries. This is unlike products with copy-on-write technologies, where data needs to be copied to the copies before being changed and performance overhead increases as the number of copies increases. With XVC, you can have as many XtremIO Virtual Copies as the support matrix allows, without impacting performance on a production source volume. When being read from or written to, the code paths within XtremIO for accessing a XVC volume or a production source volume are identical.
EMC studied the performance of two SQL Server databases that were created from a regular volume versus a copy of the regular volume in the lab. As shown in Figure 2, the regular volume v-sql1 and the XVC volume snap.v-sql1 exhibit identical performance traits. Both SQL Servers were generating about 71K IOPs, with an average read latency of 212us, and an average write latency of about 400us.

Figure 2. XVC performance

STORAGE EFFICIENCY

XtremIO virtual copy (XVC) technology is implemented by leveraging the array’s content-addressing capabilities along with in-memory metadata and the system’s dual-stage metadata. As a result, copies not only receive deduplication at the physical data block level, but also share the in-memory metadata.

INITIAL XVC CREATION

When an XVC is created, the metadata for the source volume becomes an “ancestor” entity that is shared between the production source volume and the copy. New empty containers are created for subsequent changes to the production source volume and the snapshot volume. Thus, the act of creating a copy is extremely efficient that requires in-memory metadata operation only.

Zero space is consumed at the time an XVC is created. The operation of creating a copy on a database of a few megabytes or hundreds of terabytes is instantaneous.
POST XVC CREATION

After creation of an XVC, both the production source volume and the copy continue to be written to and read from. Any INSERT or UPDATE operation to the source or copy will result in new metadata entries in the corresponding volume. If unique data is identified, it will be compressed, and then written to the SSD. As shown in Figure 4, an UPDATE triggered data block B to change on the production source volume and a new entry is written to the production metadata volume. As the content of the data block is identified as unique, it is compressed and subsequently written to the physical media.

XVC efficiently shares metadata across the source and its copies. Database operations such as UPDATE or DELETE essentially trigger a decrease reference count of the underlying physical data block(s). A data block is marked for removal from the physical media only when there are no more references to it.

Conventional copy-on-write technology requires data blocks to be copied to each of the copies before the source can be updated. Writes to a source volume could potentially trigger an explosion of space consumption when the number of copies is large. XVC effectively solves the data explosion challenge with the unique redirect-on-unique-write copy technology—only changed data blocks that are unique and have never been written to the physical media will take up additional space.

As Figure 4 illustrates, existing copies of the source volume do not consume additional space due to changes in the source volume. They continue to share common metadata and physical data blocks. This enables XtremIO to support a large number of copies on any volume without worrying about an explosion of disk space.
Figure 4. Changes to production source volume post-XVC creation

Similarly, changes to a copy will affect the copy only. The space usage and performance of the production source volume or other copies will not be impacted. See Figure 5 on changes to XVC.

Figure 5. Changes to copy volume post-XVC creation

Any data inserted or updated to the production source database or copy database post XVC creation will be automatically compressed. However data is not expected to receive any data deduplication savings due to the unique page header SQL Server writes to each of its 8K data pages.

In general, a copy will exhibit the same space efficiency pattern as a normal volume for any data written post XVC creation. As the data on the production source volume and the copies diverge over time, the storage efficiency savings will decrease. As such, it is a good practice to refresh copies periodically or after major data load or maintenance operations to receive up-to-date production data as well as regain the space savings.
ENHANCING DATABASE LIFECYCLE MANAGEMENT: USE CASE STUDIES

Throughout the SQL Server database application lifecycle, copies of production databases are essential and valuable to database administrators for continuing development, effective troubleshooting, reporting, and managing the database environment. XtremIO iCDM significantly simplifies the copy operation and eliminates the cost of copies, opens up many new use cases of using copies to increase database administrator productivity, and accelerates database application lifecycle management.

USE CASE I: TEST AND DEVELOPMENT

As application features continue to be developed, tested, and rolled out to production, copies of the existing production database are needed to support development efforts, perform user acceptance testing, and so on. As the production database continues to change and grow, database administrators face some serious challenges. These include:

- How to deploy and refresh test and development environments efficiently—traditional backup/restore method takes too long and is too resource intensive
- Where to find the additional storage to host multiple test and development copies of the database
- How to ensure test and development are running on storage with performance similar to the production

Example: XYZ company has a 5 TB production database. The company’s business processes require it to host 12 versions of the database in order to support 100 developers for development, staging, testing, and other needs. Before XtremIO, XYZ company provided its developers with copies of the database by restoring backups of the production database. It took a few hours for the database to be restored in each environment. In addition, it was a constant struggle to find disk space to accommodate the copies. Copies were restored into a separate storage array so that the development copies would not impact production. The developer always struggled for storage IO because the spinning disk array just couldn’t keep up.

After implementing XtremIO and taking advantage of XtremIO iCDM, XYZ company was able to improve its production database performance. With iCDM, the company could deploy multiple copies of the database in seconds on the same array as the production database in order to support its development and test environments. With XtremIO inline storage efficiency and space-efficient XVC, the company achieved a 10:1 data reduction ratio. DBAs no longer have to beg and borrow storage space. With the consistent, superior performance, and space-efficient copy provided by XtremIO all-flash storage, XYZ company has plenty of performance and capacity to support its consolidated production, test, and development environment.

XtremIO iCDM can be leveraged to provide test and development copies of production data. Multiple master copies can be created and each copy can be processed (such as an anonymization/sanitization process) to build a golden image for different development and test stages. Child copies can then be created from each master copy and presented to various development or test teams. Provisioning copies is an easy and instantaneous process.

For a sample deployment model of repurposing copies for test and development environment using PowerShell, see Appendix I.

USE CASE II: NEAR REAL-TIME ANALYTICS

One of the primary purposes of a database is to service reporting. Reporting queries are typically long running, require a large amount of CPU and memory resources, and are heavy on disk IOs. Mixing reporting with online user activities on the same SQL Server machine may cause resource contention, locking, and deadlocking issues. The ability to offload reporting and separate reporting activities from primary online user workload helps to reduce load and improve performance on the production environment.

Example: XYZ company provides online education services offering K-12 education to kids nationwide. August is the company’s peak season with students coming back to school and new student enrollments. Last summer, the company enrolled so many new students that the CPU on the production SQL Server cluster was completely overwhelmed.

To alleviate the CPU pressure from the production SQL Server serving online user requests, XYZ company was able to leverage the XtremIO iCDM feature to create near real-time copies of the production database every 15 minutes and offload some heavy reporting workloads from the production SQL Server to a secondary SQL Server host.

XtremIO iCDM enables you to easily build a solution that scales out a database to provide near real-time copies of the production data for reporting purposes.
USE CASE III: DATABASE MAINTENANCE

dbcc checkdb is a maintenance task that checks for the logical and physical integrity of all the objects in the database. It is an important task to perform to ensure the health of the SQL Server database. However, dbcc checkdb can be very disruptive to a production SQL Server. The SQL Server buffer pool may be completely trashed as checkdb reads all of the pages of all the objects. Running dbcc checkdb on a production server is not a recommended practice. As a result, many database administrators went through a lengthy process to restore a copy of the production database onto a non-production environment to perform dbcc checkdb. Others may not even get the luxury of running dbcc checkdb, thus incurring more risks of data corruption.

Example: Tommy is the DBA of XYZ company. Tommy used to run dbcc checkdb on the production SQL Server database every week during the regular maintenance window. Ever since the recent company merger, the size of the database has grown by 70 percent. The dbcc checkdb job now takes longer to complete. Tommy is getting complaints from users on applications running slow during certain hours of the day. Performance data pinpointed the cause of performance degradation to the dbcc checkdb job.

Fortunately, XYZ company recently moved its SQL Server environment to XtremIO storage. To reduce the performance impacts to the production environment, Tommy was able to create a copy of the production database instantly by using XtremIO iCDM, attaching the copy to a secondary SQL Server host, and offloading the dbcc checkdb job to run on the secondary host. The copy is deleted after the dbcc checkdb job is done.

With XtremIO iCDM, database administrators can easily attach a copy of the production database to a secondary SQL Server instance to run dbcc checkdb without worrying about trashing the production SQL Server buffer pool. In addition, with the XtremIO snapshot refresh feature, database integrity can be checked periodically on a copy to ensure the ongoing health of the database.

USE CASE IV: QUERY TUNING

As an application continues to mature, with new features added, often the existing indexes on the database may cease to be sufficient for supporting user activities. Index tuning is part of the daily life for any SQL Server database administrator. Effective query tuning requires running on a production-like database and may take many trials and errors. Trying to tune on a production environment is not practical. However, it would be challenging to get a copy of the terabyte-sized production database for query tuning—both from a storage capacity requirement perspective and a deployment time perspective. Many SQL Server database administrators face this dilemma with traditional storage.

Example: Tommy is the DBA of XYZ company. Tommy has been receiving user complaints about performance issues of the production SQL Server database. Given the increase in the size of the database due to a recent company merger, Tommy suspects that the existing indexes on the database might no longer be sufficient. Tommy likes to do a thorough examination of the existing indexes, remove some old indexes, and check out some new ones to see if that could improve the performance. He needs an environment with a production-like database to assess the impacts of index changes before rolling out the changes to production.

With the production SQL Server database running on XtremIO storage, Tommy was able to create a copy of the database instantly, mount and recover the copy onto a secondary SQL Server host, and perform exhaustive index tuning on production data without worrying about impacting the performance of the production environment.

XtremIO iCDM can create a fully writable copy of a database instantly without using additional physical storage space. When attaching the copy to a secondary SQL Server instance, you can have an environment dedicated for tuning purposes without impacts to the production workload.

USE CASE V: BACKUP

Backup is essential in order to provide data protection to a SQL Server database. Backing up SQL Server databases running on a traditional storage can be time-consuming and resource-intensive, and requires a large amount of storage. A common backup storage strategy involves first backing up the local SAN, copying the backup to a network file share, and then copying the backups to an archive system onsite and creating copies to ship offsite.
Example: Tommy is the DBA of XYZ company. Tommy has been receiving user complaints that the nightly batch process takes too long time to complete on some days. Further investigation showed that the nightly batch process was overlapping with the weekly full database backup. Both processes were competing for server sources, hence the slowdown in performance.

Tommy has been taking local backups of the production database for fast database recovery in case of logical data corruption or user errors. He found that by using the XtremIO iCDM feature with AppSync, he could perform a full backup of the 5 TB production database in a couple of minutes. In addition, the backup copy consumed near zero space. For long term retention, he can move a physical copy of the full database backup to the less expensive cloud storage weekly.

To alleviate the resource contention issue on the production SQL Server, Tommy decided to move the weekly backup process off of the primary SQL Server. Instead of running a full backup of the database on the production SQL Server, he ran a periodic restore of the copies taken by AppSync onto a secondary SQL Server host, copied only the backup from the secondary SQL Server host, and moved the backup off of primary storage for long-term retention.

XtremIO iCDM can significantly enhance SQL Server database backup and restore to the local SAN by:

- Using copy technology to perform rapid backups and restores
- Offloading resource-intensive backup processes to a secondary compute unit
- Storing multiple backup images efficiently with inline data deduplication technology

Restoring from a copy is effortless. Database administrators can easily perform full recovery of the database or mount the copy with a different database name to partially restore table(s) or row(s) of a database.

USE CASE VI: ALWAYSON AVAILABILITY GROUPS

Many organizations deploy SQL Server AlwaysOn Availability Groups (AlwaysOn AG) to provide local high-availability protection for SQL Server databases. With the AlwaysOn AG architecture, each replica has its own copy of the database. SQL Server manages the data synchronization between the replicas. AlwaysOn AG has many benefits over other local high-availability solutions, including: a less than five-seconds failover, easy setup and maintenance, a readable secondary, and others. However, organizations using AlwaysOn AG to protect their databases typically encounter the following issues:

- Storage costs multiply due to the number of replicas deployed
- Seeding and reseeding secondary replicas can be time-consuming (especially for large databases or databases with high transaction rates) and databases can be running without protection in the meantime.

Example: Tommy is the DBA of XYZ company. He recently completed the migration of the company’s 5 TB production SQL Server database onto an XtremIO all-flash array. Next, he needs to deploy two AlwaysOn AG secondary replicas to provide local high-availability protection for the production database and to service reporting needs.

Because each replica will have its own copy of the database, Tommy estimated that he will need an additional 10 TB of storage to store the databases of the two replicas. To minimize the impact on production, Tommy estimated it would take him one week to complete the AlwaysOn AG deployment. Backing up the 5 TB database on production and restoring onto two replicas would probably take up most of that one week.

Tommy was concerned that the production database would be unprotected for a week. He learned about XtremIO iCDM and immediately decided to give it a try. Tommy found that following the exact AlwaysOn AG deployment steps from the SQL Server instructions (simply replacing the traditional backup/restore method with XtremIO iCDM using AppSync) reduced the AlwaysOn AG deployment time dramatically. Tommy was able to complete the deployment of two replicas within ten minutes instead of one week. Furthermore, initial deployment of two replicas consumed nearly zero storage space. Only changes to the primary after the initial deployment take up physical storage space. The space consumption for the replicas is significantly less than that of traditional storage.
AppSync integrates with both the XtremIO all-flash array and SQL Server to enable an application-consistent copy of the database. SQL Server has full knowledge of the application consistent copy and treats the storage copy as a full database backup. Hence an application consistent copy can be mounted and recovered as a database with "no recovery" mode, and served as an AAG secondary replica to continue receiving log stream synchronization from the primary. The nature of XtremIO iCDM enables not only rapid deployment, but also extreme storage efficiency.

For a sample of using XtremIO iCDM for rapid deployment of an AAG secondary replica, see Appendix II.

**USE CASE VII: UPGRADE**

As applications and SQL Server continue to mature, more features are released. Environments need to be upgraded to take advantage of the new features. XtremIO iCDM can enhance the upgrade experience by simplifying the upgrade process, minimizing planned downtime, and reducing the risks of the upgrade.

Example: XYZ company is about to go through a major upgrade of its online education system. A number of schema and data changes are planned for the SQL Server database. As the lead DBA, Tommy is expected to provide database support during the application upgrade. Tommy has been challenged to provide a quick failback plan for the 5 TB production database in case anything goes wrong during the upgrade.

Tommy immediately thought of getting an application consistent copy of the database using AppSync. If the database needs to be rolled back for any reason, he can restore the older version of the database from the application-consistent copy in seconds. A copy takes virtually zero space on XtremIO. So he does not need to beg and borrow for additional storage.

With the capability to create an instant writable copy of the database without any additional storage space, you can enhance the upgrade process by using XtremIO iCDM to:

- Test the upgrade on a copy of the database to resolve any potential issues
- Run through the upgrade process using a copy of the database to get a real sense for how long the upgrade will take
- Protect the database by getting a copy of it prior to the upgrade; if the upgrade fails for any reason, simply point the SQL Server instance at the copy and recover

**COMMON CONSIDERATIONS FOR COPY MANAGEMENT**

**APPLICATION-CONSISTENT VERSUS CRASH-CONSISTENT COPY**

XtremIO iCDM can be used to enhanced many SQL Server usage scenarios and simplify day-to-day operations for a database administrator. It supports both crash-consistent copy and application-consistent copies. So what is the difference between a crash-consistent copy and an application-consistent copy and when do you use each one?

**CRASH-CONSISTENT COPY**

A crash-consistent copy captures the state of the data volumes at a particular point in time. SQL Server has no knowledge of the copy. As such, there is no impact to running a SQL Server instance. A crash-consistent copy of a SQL Server database is equivalent to or consistent with the state of running the database during a power failure. In that state, there may be data pages in memory not yet flushed to disk and open transactions. SQL Server uses write ahead logging (WAL). All committed transactions are logged in the SQL Server transaction log on disk. SQL Server can recover from a crash using information in the transaction log. Data pages not yet flushed to disk can be re-created and in-flight transactions will be marked as failures.

**APPLICATION-CONSISTENT COPY**

An application-consistent copy requires coordination of the copy with SQL Server. XtremIO integrates with EMC AppSync to support an application-consistent copy with SQL Server. AppSync coordinates SQL Server (via SQL Server Virtual Device Interface) and XtremIO to makes a copy. XtremIO also includes a native VSS Provider that allows developers or third-party vendors to write their own AppSync-like utilities to make application-consistent copies.
Before making an application-consistent copy, SQL Server is notified that it is about to backed up, so it can prepare by:

- Either committing or rolling back any in-flight transactions
- Running checkpoint to flush dirty pages to disk and making a note of the log sequence number so that the data files and the log files are synchronized to minimize the amount of work during restore
- Freezing IO
- Taking a backup of the metadata

An application-consistent copy is considered a full backup for SQL Server. With AppSync, when an application-consistent copy is created with the database in full recovery mode, the copy can then be mounted and restored into various recovery states—recovery, norecovery, or standby (see Figure 6). When mounting and recovering copy in norecovery mode, the database is restored and left in a non-operational state, thus additional transaction logs can be restored.

**Figure 6. EMC AppSync-supported database recovery types**

**DETERMINING WHEN TO USE WHICH TYPE OF COPY**

The key differences between application-consistent and crash-consistent copies are:

- **The amount of work needed during recovery of the database.** In an application-consistent copy, the data files and log files are in sync. There is no additional work required to redo or undo any transactions. While both application-consistent copies and crash-consistent copies have the same RPO, application-consistent copies offer better RTO.

- **Whether SQL Server freezes IO during copy.** SQL Server allows up to ten seconds for a copy to be taken. Applications may experience a performance drop during the time the IO is frozen.

- **Whether SQL Server has a metadata backup of the copy operation or treats the copy as a full backup.** With application-consistent copy, SQL Server treats the copy operation as a full backup and stores metadata of the copy operation. An application-consistent copy can be restored into norecovery mode to continue roll-forward log backups.

To determine which type of copy to use, you should consider the availability requirement, frequency of the copy, and its impacts to performance, and whether you need to support log backups. The following are sample scenarios for using application-consistent versus crash consistent copies.
Application-consistent copy

- Database backup
- Creating AlwaysOn availability groups secondary replica, database mirroring mirror server, or log ship secondary

Crash-consistent copy

- Repurposing for test, development, or reporting
- Offload processing, for example, tuning, dbcc

REFRESHING STALE COPIES

Whether you are using XtremIO iCDM to provide copies for test, development, or reporting, the copies eventually become outdated. Therefore, there should be a process in place to periodically update the reporting data, or provide new versions of the database for test or development usage. The XtremIO snapshot refresh feature simplifies the process for updating a stale copy.

The following steps describe the detailed Transact-SQL, Windows PowerShell, and XtremIO CLI scripts and processes to refresh the data of a SQL Server database using the XtremIO snapshot refresh feature.

Step 1. Detach the stale database

Execute the following SQL statement:

```
sp_detach_db @dbname='<database name>'
```

Step 2. Dismount the database volume(s)

In PowerShell, first list all XtremIO disks, then offline the specific database disk(s), verify disk is offline.

```
Get-disk -FriendlyName XtremIO*
Set-Disk -Number 6 -IsOffline $True
Get-Disk | Where-Object IsOffline –Eq $True
```

![Figure 7. Sample PowerShell scripts on dismounting a volume](image)

Step 3. Run snapshot refresh on XtremIO

Figure 8 shows the parent-child relationship for the production source volume, the master copy, and the second level copies for the databases.
Run "create-snapshot-and-reassign" CLI command to refresh the master copy from the production source volume, then run "create-snapshot-and-reassign" on each of the reporting volumes. The "create-snapshot-and-reassign" command updates the copy volume with data from the source and saves and renames the old copy with a new name. For example, SnapshotSet.1437607649 as shown in Figure 9 is the old version of sql1-master-snap.

The "create-snapshot-and-reassign" CLI command supports snapshot refresh at volume, consistency group, or snapshot set level. See Figure 10 for details on syntax.

In PowerShell, list all offline XtremIO disks, and then online the specific database disk(s), verify disk is online.

```
Get-disk -FriendlyName XtremIO* | Where-Object IsOffline -Eq $True
Set-Disk -Number 6 -IsOffline $False
Get-disk -FriendlyName XtremIO*
```
Step 5. Attach the database

Execute the "create database ... on <filespec> for attach" SQL statement to attach the database, for example:

```
USE [master]
GO
CREATE DATABASE [tpce] ON
    ( FILENAME = N'J:\MSSQL_tpce_root.mdf' ),
    ( FILENAME = N'J:\TPCE_Log.ldf' ),
    ( FILENAME = N'J:\Fixed_1.ndf' ),
    ( FILENAME = N'J:\Scaling_1.ndf' ),
    ( FILENAME = N'J:\Scaling_2.ndf' ),
    ( FILENAME = N'J:\Scaling_3.ndf' ),
    ( FILENAME = N'J:\Growing_1.ndf' ),
    ( FILENAME = N'J:\Growing_2.ndf' ),
    ( FILENAME = N'J:\Growing_3.ndf' )
FOR ATTACH
GO
```

DATA MASKING

Many organizations have requirements to sanitize production data before presenting data to non-production environments. There are numerous laws and regulations such as Sarbanes-Oxley, PCI DSS, and HIPAA that specify required privacy data protection measures. The process of sanitizing the data and protecting sensitive data from unauthorized access is called data masking.

The methods to mask or obfuscate data can range from simple substitution of sensitive data with generic blank values, randomizing the existing values using character scrambling, artificial data generation, and averaging, to the more complex methods that maintain referential integrity and data distribution.

Depending on the business requirements, different masking methods may be used for different data. No matter which methods are used to obfuscate the data, the process of data masking typically involves the following stages.

1) Identifying data that needs to be masked
2) Defining the masking functions to use for each data field
3) Transforming the data using the defined masking functions
4) Testing and verifying results of the data masking

Depending on the complexity of the data structures and masking requirements, DBAs may develop custom T-SQL scripts to achieve the data masking, or look to third-party data masking tools for assistance with the process. Many third-party data masking tools do automatic initial discovery of sensitive data, provide masking suggestions, allow customization, and generate the final T-SQL scripts for data masking.

With XtremIO iCDM, you can integrate any of your existing data masking processes as one stage in the copy data management workflow. With the support of multi-level copies, data masking can be performed and tested on a first generation copy. Multiple secondary generation copies can be created based on the first generation copy that has already been sanitized. Figure 11 depicts a multi-level copy workflow with data masking.
Figure 11. Workflow for repurposing with data masking

IO RESOURCE GOVERNING

XtremIO iCDM significantly simplifies the process and cost for creating and managing copies of databases. When copies are being used for different purposes and stages of the database lifecycle, IOPS throughputs for certain repurposed copies must be controlled (for example, resource usages for test and development copies).

SQL SERVER RESOURCE GOVERNOR

The SQL Server 2014 Resource Governor IO control feature provides the ability to control the maximum outstanding IOs per volume. With the configurable settings MIN_IOPS_PER_VOLUME and MAX_IOPS_PER_VOLUME, you can set the lower bound and upper bound IOPS limits per volume. On shared storage, setting MIN_IOPS_PER_VOLUME on the production volumes will guarantee the production database gets all of the IO resource it needs to perform well. For test or development copies, setting the MAX_IOPS_PER_VOLUME parameter will allow effective throttle for the IO bandwidth used for the less critical workloads.

Setting up Resource Governor for IO Governance

By default, Resource Governor is disabled and needs to be enabled prior to use. It can also be disabled at any time and the configuration will be saved. This will mean the classifier function is no longer throttling IO.

Once enabled, four short steps are required to configure its use.

1. Create a resource pool which defines either or both;
   - MIN_IOPS_PER_VOLUME
   - MAX_IOPS_PER_VOLUME

   ```sql
   USE master
   GO
   CREATE RESOURCE POOL RestrictedIOPool_OLTP_1_SNAPSHOT WITH
   (    MAX_IOPS_PER_VOLUME = 1000,
       MIN_IOPS_PER_VOLUME = 1
   );
   GO
   ```

2. Create a workload group that will use the resource constraints defined by the resource pool.

3. Create a classifier function to route the processes to the respective workload groups based on a condition.

4. Alter Resource Governor to use the classifier function for incoming connections.
The classifier function is the most critical configuration step as it governs the routing of IO. The simpler you can make the logic within the function, the better.

SQL Server Management Server (SSMS) can also be used to create the appropriate objects if required and Figure 1 shows the Resource Pool and Workload Group that were created for testing visible within the SSMS Object Explorer.

![Resource Governor View]

**Figure 12. SSMS Resource Governor view**


**VMWARE VSPHERE STORAGE TO CONTROL**

VMware vSphere storage IO control (SIOC) was initially introduced in vSphere 4.1 to address the "noisy neighbor" problem, preventing a low priority virtual machine consuming too much IO resource that impacts higher priority virtual machines. It was further extended in later vSphere versions to provide more sophisticated capabilities, including setting limits and shares on a VMDK and setting IOPS reservation for a VMDK.

When using XtremIO iCDM to consolidate production and non-production environments, an effective strategy for IO resource governance is to limit the bandwidth available to the non-production copies. You can limit IOPS for VMDKs in the non-production virtual machine via virtual machine "edit settings." See Figure 13 for operation within vCenter web client.
IMPACTS TO LOG TRUNCATION

Many of the SQL Server operations—for example, replication, change data capture, log only backup, and others—are dependent on the transaction log. Some people have concerns about XtremIO iCDM potentially impacting the transaction log-dependent operations. Below we examine application-consistent copies and crash-consistent copies that are supported by XtremIO iCDM, and their respective impacts to SQL Server transaction log-related operations.

APPLICATION-CONSISTENT COPY

Making an application-consistent copy of the database is equivalent to performing a full backup of the database. While there may still be misconceptions out there, it is important to understand that SQL Server only truncates the transaction log when there is a transaction log back or during checkpoint for a database running in SIMPLE recovery mode. For a database running in FULL or BULK_LOGGED recovery mode, a full database backup, with or without the copy_only option, never triggers a log truncation.

With the database in FULL recovery mode, you may perform log-only backups after taking the application-consistent copy with full backup option using XtremIO iCDM and restore the log backups after mounting and recovering from the copy to achieve point-in-time database recovery.

Application-consistent copies created in full backup mode maintain full log sequence number (LSN) information. The copy can be used to seed an AlwaysOn AG secondary replica.
Figure 14. Application-consistency copy using full backup option

CRASH-CONSISTENT COPY

A crash-consistent copy is made without SQL Server being notified, hence there is no impact to regular log operations on the production database. The transaction log file is guaranteed to be in a write order consistent state if the crash-consistent copy contains data that has been hardened on disk. When a crash-consistent copy is restored, SQL Server will perform recovery from log entries in the transaction log. The SQL Server log sequence chain is broken. No additional log backup can be restored on the copy.

In summary, neither an application-consistent copy nor the crash-consistent copy of the database has impact on SQL Server log truncation. Log truncation is only triggered under the following conditions.

- For database under FULL or BULK_LOGGED recovery mode, log truncation is triggered when a log backup is performed without the “copy_only” option. Regular log backups should be scheduled to control transaction log growth.
- For database under SIMPLE recovery mode, log truncation is triggered when checkpoint occurs.


LICENSING

XtremIO iCDM opens up many new options that accelerate SQL Server lifecycle management. The process for creating a copy of a multi-terabyte database is instant. The storage cost is negligible. The benefit is not only increased performance of the production system, but also incredible agility that enables DBAs to do what they were not able to do before. As iCDM become increasingly popular for DBAs, it is important to discuss the potential SQL Server licensing impacts when deploying copies of databases.

Generally, when a copy of the database is created, it is deploy with one of the following options:

1. Attach to an existing SQL Server instance
2. Attach to a new SQL Server instance on an existing physical or virtual operating system environment (OSE)
3. Attach to SQL Server instance on new physical or virtual operating system environment (OSE)

Both options 2 and 3 may have licensing implications.

With the SQL Server Server+CAL (client access license) licensing model, you are allowed to run as many instances on an OSE without additional license. No additional license is needed for option 2 if you’re licensed with SQL Server Server+CAL licensing. The same is not true under SQL Server Core-Based licensing.
Option 3 generally would require additional licenses unless your organization has a volume licensing agreement with Microsoft or you are running in a virtual environment that’s licensed for maximum virtualization with software assurance.


ROLE-BASED CONTROL

Typically, organizations have DBAs for managing the SQL Server database and storage administrators for managing storage. While DBAs can decide when to create iCDM copies of databases and how to consume the copies, creating the copy is a storage operation. In order to enable self-service copy data management for DBAs while maintaining storage and database security, it is critical to have a flexible role-based control user management system.

AppSync provides role-based authentication, allowing, for example, a storage administrator to add and configure the storage, manage licensing, and add hosts. A system administrator can, in turn, configure and run Service Plans. And yet another user, such as a DBA, can manage his or her own application replication environment, refreshing their UAT/QA environments, for example.

Roles determine the operations that a user can perform. A user can have multiple roles. Refer to table below for detailed permissions defined for the roles. Roles are accumulative, not hierarchical.

<table>
<thead>
<tr>
<th>Role</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Administrator</td>
<td>• Configure LDAP servers</td>
</tr>
<tr>
<td></td>
<td>• Add / modify / remove users</td>
</tr>
<tr>
<td></td>
<td>• User roles management</td>
</tr>
<tr>
<td>Resource Administrator</td>
<td>• Register storage array, host, hypervisor</td>
</tr>
<tr>
<td></td>
<td>• License management</td>
</tr>
<tr>
<td></td>
<td>• Manage server settings</td>
</tr>
<tr>
<td>Service Plan Administrator</td>
<td>• Add / modify / delete service plan</td>
</tr>
<tr>
<td></td>
<td>• Run a service plan</td>
</tr>
<tr>
<td>Data Administrator</td>
<td>• Discover SQL Server</td>
</tr>
<tr>
<td></td>
<td>• Manage database to service plan subscription</td>
</tr>
<tr>
<td></td>
<td>• Copy management operations</td>
</tr>
</tbody>
</table>

For additional information on AppSync security and user management, refer to https://support.emc.com/docu61179_AppSync-2.2.2-Security-Configuration-Guide.pdf?language=en_US.

MONITORING

IT administrators used to equate the disk space usage on a server to disk space usage on storage. While that might be true for traditional storage, it is no longer the case for a modern storage array with advantage data services. As explained in earlier sections, on a modern storage array like XtremIO with thin provisioning and in-line data services for deduplication and compression, a data block is stored once and referenced many times. A data block is compacted before it is stored onto the physical media. As a result, the actual physical capacity used on the storage can be much smaller than the disk space use observed on the server.

Figure 15 shows the effects of XtremIO iCDM storage efficiency after initial creation of 1, 2, 4, and 8 copies of a database. While the provisioned capacity, space reserved by SQL Server, and space used by SQL Server multiplies as the number of copies increases, the physical capacity used on the storage remains the size of a single database.
Figure 15. Storage efficiency with database copies

If you are a storage administrator trying to project future storage growth needs, the most acute way of looking at storage consumption is by monitoring the physical capacity usage.

Storage

<table>
<thead>
<tr>
<th>Overall Efficiency</th>
<th>42.2:1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Reduction Ratio</td>
<td>3.8:1</td>
</tr>
<tr>
<td>Deduplication</td>
<td>1.0:1</td>
</tr>
<tr>
<td>Compression</td>
<td>3.6:1</td>
</tr>
</tbody>
</table>

Thin Provisioning Saving | 91%

Volume Capacity | 26.789TB

- 650.014GB Physical Space Used
- 2.49TB (9%) used
- 24.28TB (91%) free

Physical Capacity | 7.587TB

- 650.014GB (9%) used
- 6.952TB (92%) free

Figure 16. Physical capacity usage dashboard view

As a database administrator, you can also monitor available disk space from within SQL Server. SQL Server supports a dynamic management function (DMF) "sys.dm_os_volume_stats()" as of SQL Server 2008 R2 SP1 for discovery of database disk usages. The following example returns the total space and available space in bytes for the database files in the current database.
SELECT database_id, f.file_id, volume_mount_point, total_bytes, available_bytes
FROM sys.database_files AS f
CROSS APPLY sys.dm_os_volume_stats(DB_ID(f.name), f.file_id);

SUMMARY
With its unique combination of consistent performance, scale-out on demand, in memory XVC, powerful application integrations, and self-service tools, XtremIO iCDM provides unlimited potential to simplify and accelerate all phases of SQL Server database application lifecycle management. XtremIO iCDM enables you to:

- Reduce risks of rolling out new features by developing and testing on a production-like environment
- Deliver near real-time reporting without impacting production SLA
- Offload heavy lifting backup and maintenance tasks from your production database
- Reduce downtime for service pack and version upgrades
- Deploy AAG secondary replicas in minutes with storage efficiency

XtremIO iCDM consolidates copy services for all application, database, development, and data protection teams, boosting business agility to transform customers’ business processes, operations, and infrastructure.

APPENDIX I: SAMPLE DEPLOYMENT – REPURPOSING FOR TEST/DEV
This section describes the sample workflow for repurposing deployment test / development environment using PowerShell scripts. In this workflow, you will see how we use XtremIO multi-level crash consistent copies to create first generation copy, apply simple data masking, and deploy test / development environment off the sanitized first generation copy.

The repurposing for test / development workflow can be broken down into 6 steps.

1) Create first generation crash consistent copy from production source database prod
2) Mount and recover first generate copy as stageDB
3) Execute data masking scripts "c:\scripts\dataMask.sql" to mask customer SSN data on stageDB
4) Create second generation copy from the first generation copy
5) Mount and recover second generate copy as testDB

The followings are detail scripts.

```powershell
# Adding exception to accept a self - signed certificate or accepting an X509Certificate that previously didn't exist
add-type @"using System.Net;
public class TrustAllCertsPolicy : ICertificatePolicy {
    public bool CheckValidationResult(
        ServicePoint srvPoint, X509Certificate certificate,
        WebRequest request, int certificateProblem){
        return true;
    }
}
""
```
## A generic wrapper function that queries XtremIO REST API using HTTP/GET

# Retrieves and lists existing configuration of an object or multiple objects

```powershell
function ExecuteGetRestQuery ($xmsip,$cfgOption,$headers)
{
  try
  {
    $baseUrl = "https://"+$xmsip
    $resUrl = '/api/json/v2/types/
    $url = $baseUrl + $resUrl + $cfgOption
    Write-Host "ExecuteGetRestQuery()::"$url
    $jsonserial.MaxJsonLength = [int]::MaxValue
    $result = $jsonserial.DeserializeObject((Invoke-WebRequest -Method GET -Uri $url -Headers $headers))
    return $result
  }
  catch{
    return $false
  }
}
```

## A generic wrapper function that queries XtremIO REST API using HTTP/POST

# Creates a new object with specified properties

```powershell
function ExecutePostRestQuery ($xmsip,$cfgOption,$data,$headers)
{
  $baseUrl = "https://"+$xmsip
  $resUrl = '/api/json/v2/types/
  $url = $baseUrl + $resUrl + $cfgOption
  Write-Host $url
  Write-Host $data
  $jsonserialInput.MaxJsonLength = [int]::MaxValue
  $jsonSerialOutput.MaxJsonLength = [int]::MaxValue
  $result = (Invoke-RestMethod -Method POST -Uri $url -Body $data -Headers $headers)
  return $result
}
```

## A function to create XtremIO snapshot on a single volume or an existing snapshot
# It accepts name of the object (volume or a snapshot) name and snapshot suffix as arguments and invokes

```powershell
function createXtremSnapshot ($xmsip,$parentVolumeName,$snapSuffix,$ssName,$headers)
{
  Write-Host "+++++++++Creating snapshot from "$parentVolumeName
  $cfgOption = 'snapshots'
  $parentVolumes = ['"$parentVolumeName"]
  $parentVolumes = ($parentVolumes|ConvertTo-Json).ToString()
  $snapSuffix = ($snapSuffix|ConvertTo-Json).ToString()
  $ssName = ($ssName|ConvertTo-Json).ToString()
  $data = @"{
    "volume-list":$parentVolumes,
    "snap-suffix":$snapSuffix,
    "snapshot-set-name":$ssName
  }"
  return ExecutePostRestQuery $xmsip $cfgOption $data $headers
}
```
# A function to create XtremIO snapshot from a consistency group
# It accepts name of the consistency group, target snapshot suffix, and resulting snapshot set name as arguments and invokes ExecutePostRestQuery
###

```powershell
function createXtremSnapshotByCG ($xmsip,$cgName,$snapSuffix,$ssName,$headers)
{
    Write-Host "+++++++++Creating snapshot from" $parentVolumeName
    $cfgOption = 'snapshots'
    $cgName = ($cgName|ConvertTo-Json).ToString()
    $snapSuffix = ($snapSuffix|ConvertTo-Json).ToString()
    $ssName = ($ssName|ConvertTo-Json).ToString()
    $data = @"{
        "consistency-group-id":$cgName,
        "snap-suffix":$snapSuffix,
        "snapshot-set-name":$ssName
    }"
    return ExecutePostRestQuery $xmsip $cfgOption $data $headers
}
```

###
# A function to create XtremIO snapshot from a snapshot set
# It accepts name of the snapshot set, target snapshot suffix, and resulting snapshot set name as arguments and invokes ExecutePostRestQuery
###

```powershell
function createXtremSnapshotBySS ($xmsip,$srcSSName,$snapSuffix,$targetSSName,$headers)
{
    Write-Host "+++++++++Creating snapshot from" $parentVolumeName
    $cfgOption = 'snapshots'
    $srcSSName = ($srcSSName|ConvertTo-Json).ToString()
    $snapSuffix = ($snapSuffix|ConvertTo-Json).ToString()
    $targetSSName = ($targetSSName|ConvertTo-Json).ToString()
    $data = @"{
        "snapshot-set-id":$srcSSName,
        "snap-suffix":$snapSuffix,
        "snapshot-set-name":$targetSSName
    }"
    return ExecutePostRestQuery $xmsip $cfgOption $data $headers
}
```

###
# A function to check if an XtremIO volume copy exists
# It accepts the name of the copy volume and invokes ExecuteGetRestQuery to retrieve properties of the copy volume object
###

```powershell
function checkSnapshot ($xmsip,$snapshotName,$headers)
{
    $cfgOption = 'snapshots?name='+$snapshotName
    $snapshot = ExecuteGetRestQuery $xmsip $cfgOption $headers
    if ( $snapshot -ne $null)
    {
        return $snapshot['content']
    }
    else
    {
        return $false
    }
}
```
A function returns the device NAA name of a given volume

```powershell
function getVolDeviceID ($xmsip,$volName,$headers)
{
    $cfgOption = 'volumes?name='+$volName
    $vol = ExecuteGetRestQuery $xmsip $cfgOption $headers
    if ( $vol )
    {
        $volContent = $vol['content']
        return $volContent['naa-name']
    }
    else
    {
        return $false
    }
}
```

This function is used to create lun mapping between a set of initiators and volume

```powershell
# This function is used to create lun mapping between a set of initiators and volume
# It accepts a list of initiators, the volume name and invokes ExecutePostRestQuery to create a
# new lun map between every initiator in the list and the volume

function createLunMap($xmsip, $initiatorList, $volName, $headers)
{
    $cfgOption = 'lun-maps'
    $volName = ($volName|ConvertTo-Json).ToString()
    foreach ($initiator in $initiatorList)
    {
        $initiator = ($initiator|ConvertTo-Json).ToString()
        $data = @"{
            "vol-id":$volName,
            "ig-id":$initiator
        }"
        ExecutePostRestQuery $xmsip $cfgOption $data $headers
    }
}
```

This function does a SCSI rescan to discover the new snapshot volume, mounts the volume to
the file system, and restore the SQL Server database from mdf & ldf file via the attach method.

```powershell
function mountAndRecover ($dataDeviceID, $logDeviceID, $dbName, $dataDrive, $volLabelData, $logDrive,
```
```powershell
$volLabelLog, $dataPath, $logPath) {

    ## Rescan and bring new disk online
    Update-HostStorageCache

    Start-Sleep -Seconds 1
    ## Mount disks
    mountDisk $dataDeviceID $dataDrive $volLabelData
    mountDisk $logDeviceID $logDrive $volLabelLog

    ## Attach database
    $sqlStmt = "create database " + $dbName + " on (filename = N'"+$dataPath+"'), (filename = N'"+$logPath+"') for attach"
    if ((Test-Path -Path $datapath) -and (Test-path -Path $logpath)){
        # Attach Database
        Invoke-SqlCmd -Query $sqlStmt
    }
}

# Importing SQL powershell module. This module is essential to perform management operations on SQL server
Import-Module sqlps -DisableNameChecking

### Creating authentication header object for XMS. This header object is passed to every function call defined above
$xmsip = "xmsip"
$xmsuser = "username"
$xmspwd = ConvertTo-SecureString "password" -AsPlainText -Force
$BSTR = [System.Runtime.InteropServices.Marshal]::SecureStringToBSTR($xmspwd)
$basicAuth = ("{0}:{1}" -f $xmsuser,$secPwd)
$EncodeAuth = [System.Text.Encoding]::UTF8.GetBytes($basicAuth)
$EncodeBase64Auth = [System.Convert]::ToBase64String($EncodeAuth)
$headers = @{Authorization=('Basic {0}' -f $EncodeBase64Auth)}

### Creation of header object is complete

## Step 1: Create generation 1 snapshot for stage env
createXtremSnapshotByCG $xmsip "cg-sql-prod" "g1" "ss-sql-prod.g1" $headers

## Step 2: Mount snapshot volume to stage host, create stage DB
createLunMap $xmsip "lgsc040" "sql-prod.g1" $headers
createLunMap $xmsip "lgsc040" "sql-prod-log.g1" $headers
$dataDeviceID = getVolDeviceID $xmsip "sql-prod.g1" $headers
$logDeviceID = getVolDeviceID $xmsip "sql-prod-log.g1" $headers
mountAndRecover $dataDeviceID $logDeviceID "stageDB" "M" "sql-stage" "N" "sql-stage-log" "M:\prod.mdf" "N:\prod_log.ldf"

## Step 3: Run data scubbing scripts to perform data masking
Invoke-SqlCmd -InputFile "C:\scripts\dataMasking.sql"

## Step 4: Create generation 2 snapshot from stage
createXtremSnapshotBySS $xmsip "ss-sql-prod.g1" "g2" "ss-sql-prod.g1.g2" $headers

## Step 5: Mount snapshot volume to test host, create test DB
createLunMap $xmsip "lgsc040" "sql-prod.g1.g2" $headers
createLunMap $xmsip "lgsc040" "sql-prod-log.g1.g2" $headers
$dataDeviceID = getVolDeviceID $xmsip "sql-prod.g1.g2" $headers
$logDeviceID = getVolDeviceID $xmsip "sql-prod-log.g1.g2" $headers
mountAndRecover $dataDeviceID $logDeviceID "testDB" "S" "sql-test" "T" "sql-test-log" "S:\prod.mdf" "T:\prod_log.ldf"
```

APPENDIX II: SAMPLE DEPLOYMENT – AAG Secondary Replica

This section describes the process to deploy AlwaysOn Availability Groups Secondary Replica using XtremIO iCDM AppSync self-service tool.

The processes are as followed.

1) Define Service Plan

Navigate to

![EMC AppSync > Service Plans > Microsoft SQL Server](image)

Create an on-demand Service Plan “AAG” using the Bronze template

![Create AAG Service Plan](image)

2) Create application-consistent copy of the production “tpce” database on “sql1\tpce” instance. Taking a full backup of the database will reset the differential bitmap and affect any existing differential backup process. Care needs to be taken to coordinate AlwaysOn AG deployment with existing differential backup process.

Navigate to

![EMC AppSync > Copy Management > Microsoft SQL Server > SQL1\TPCE > User Databases > tpce](image)

Select Copies Tab → Create copy using plan → AAG

Details of dataMasking.sql

Use stageDB
Go

Update Customer set SSN = ‘XXX-XX-XXXX’
Go
3) Backup the active transaction log on production database

Select Log Backups Tab → Create copy using plan → AAG

4) Mount and recover the copy onto secondary replica instance "sql2\tpce" as NO RECOVERY

Select the copy → Mount
31

Figure 20. Mount and recover copy to replica instance

5) Restore initial log backup onto secondary replica

On SQL Server Management Studio, run query to restore production log backup onto replica instance sql2\tpce

```
RESTORE LOG tpce FROM DISK = '\sql1\sqlbackup\TPC_AppSync_tpce_2016_01_06_00_12_22.trn' WITH NORECOVERY
```

Figure 21. Restore log on replica

6) Create AlwaysOn Availability Groups relationship

On SQL Server Management Studio, connect to sql1\tpce → AlwaysOn High Availability → Create New Availability Group Wizard ...
Figure 22. Create AAG using AAG Wizard in SSMS

REFERENCE


