Microsoft SQL Server with Oracle ZFS Storage Appliance: Deployment Best Practices

ORACLE WHITE PAPER | APRIL 2017

Table of Contents

Introduction

The products in the Oracle ZFS Storage Appliance family integrate advanced hardware and software architectures to offer agile, multiprotocol storage systems capable of running a demanding workload, including a variety of simultaneously operating applications and data services. Test results for industry-standard benchmarks—such as SPC-1, SPC-2, and SPECsfs—for Oracle ZFS Storage Appliance systems consistently demonstrate first-class performance characteristics.

Oracle ZFS Storage Appliance systems are based on an innovative, memory-centric storage architecture and a symmetric multiprocessing (SMP) operating system (OS) that delivers unparalleled NAS performance, efficiency, and value. They are coengineered with Oracle’s converged infrastructure stack to accelerate Oracle Database production and development test environments. They incorporate a hybrid disk storage technology that tailors the usage of data disks, log cache solid-state drives (SSDs), and read cache SSDs to the specific needs of the data share. Additional features include variable storage record size, data compression, and multiple connectivity options.

As a unified storage solution, Oracle ZFS Storage Appliance systems provide an excellent base for Microsoft Windows Server deployments by providing iSCSI or Fibre Channel block storage for Microsoft SQL Server, and network file system (NFS) or server message block (SMB) file storage for Microsoft Windows client access.

This paper discusses the best practices to follow when deploying Microsoft SQL Server on an Oracle ZFS Storage Appliance system. The intended audience is storage administrators and Microsoft SQL Server database administrators.

Maintaining High Availability

As with any business-critical application, high availability is a key design criterion to be considered when deploying a Microsoft SQL Server installation. Microsoft SQL Server 2016 can be installed on local and/or shared file systems, and Oracle ZFS Storage Appliance systems can satisfy both of these options. Local file systems (from the Microsoft Windows Server perspective) are hosted as block volumes—iSCSI and/or Fibre-Channel-connected LUNs and file systems as SMB and/or NFS volumes.

High availability starts with the network connectivity supporting the storage and server interconnectivity. Any design for the storage infrastructure should avoid single points of failure. Because there are many white papers and publications covering storage-area networking and network-attached storage resilience, those topics are not covered in detail in this paper.

Storing Data: Fundamental Units

Microsoft SQL Server datastores consist of two fundamental data structure containers: data files and log files.

The data files contain the information that would normally be considered the data. A database can consist of one or multiple data files. In Microsoft SQL Server, data files are identified by the filename extension `.mdf`.

Log files are used to store the update, addition, and deletion transactions in a replayable format, and they are stored to allow the database to be restored to a particular point in time. In Microsoft SQL Server, log files have the file extension `.ldf`.

A third file type is also created during the running and administration of Microsoft SQL Server: the backup files, which contain restorable copies of the database files to recover from mishaps, corruption, or maladministration.

The fourth type that might potentially be associated with a database is a `.ndf` file, which is a secondary file that Microsoft SQL Server might use. These are known as “next files.” For the purposes of this document, next files are considered the same as data files.

Microsoft SQL Server does not enforce this naming scheme, but it is a good practice to maintain the files according to this standard to avoid ambiguity and confusion.

A database consists of one or more data files and one or more log files, with potentially a number of next files, as shown in Figure 1.

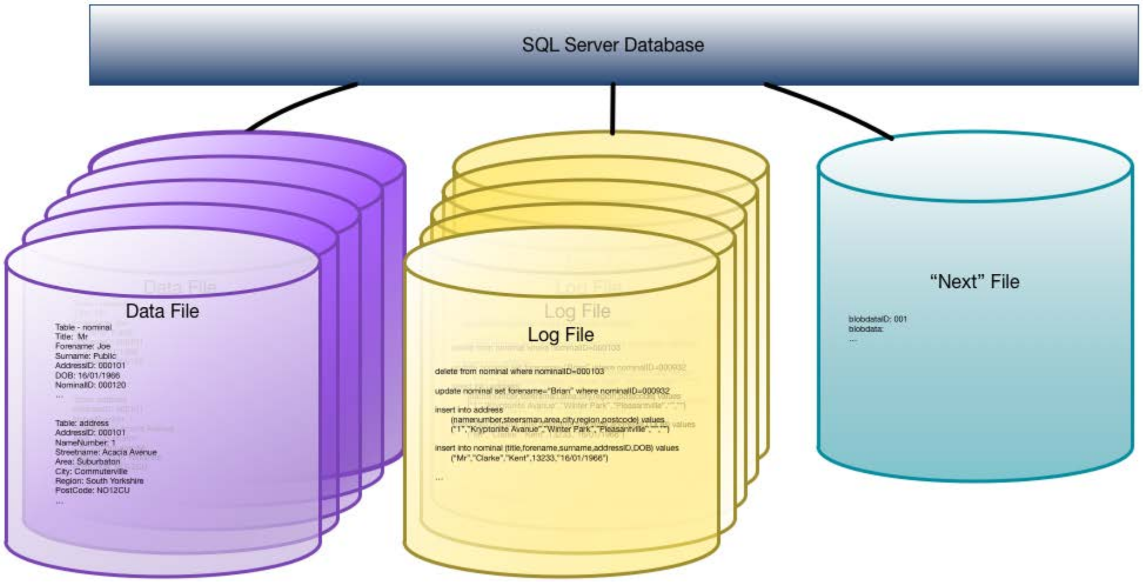
SQL Server Database
Data File
Table - nominal
Title: Mr
Forename: Joe
Surname: Public
AddressID: 000101
DOB: 16/01/1966
NominalID: 000120
...
Table: address
AddressID: 000101
NameNumber: 1
Streetname: Acacia Avenue
Area: Suburbaton
City: Commuterville
Region: South Yorkshire
PostCode: NO12CU
...
Log File
delete from nominal where nominalID=000103
update nominal set forename="Brian" where nominalID=000932
insert into address
(namenumber,steersman,area,city,region,postcode) values
("1","Kryptonite Avanue","Winter Park","Pleasantville","","")
insert into nominal (title,forename,surname,addressID,DOB) values
("Mr","Clarke","Kent",13233,"16/01/1966")
...
"Next" File
blobdataID: 001
blobdata:
...

Figure 1. Microsoft SQL Server file types

Segregating Different Datatypes

When they are created with Microsoft SQL Server Management Studio, the database log and data files are, by default, placed in the same directory. There are a number of good arguments for not placing them on the same logical drive—never mind the same directory—including the following fundamental points. There is a performance impact when reading from and writing to the same drive when applying log transactions to the data files. Also, the contention for the logical drive's queue resources could be potentially stretched too far in a highly volatile environment.

Similarly, backup files should be stored on yet more logical drive(s) and, preferably, on a different server or storage platform located in a different site.

Separating the differing datatypes allows for input/output (I/O) activity to occur, and it allows for more administrative control.

Structuring Pool Layout

Oracle ZFS Storage Appliance systems use the concept of a storage pool, which is a collection of storage and cache devices exclusively assigned to the pool. Storage is provisioned in shared file systems or block storage, and it is backed by intent-log and write-cache devices. Figure 2 illustrates an Oracle ZFS Storage Appliance system's Hybrid Storage Pool technology.

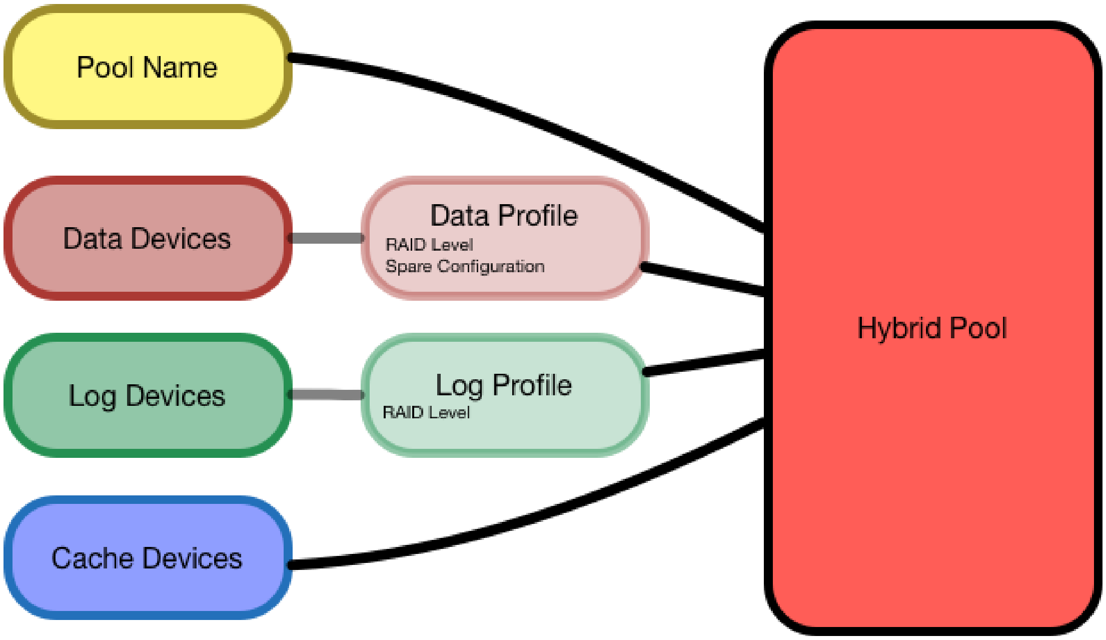

Figure 2. Hybrid Storage Pool technology for an Oracle ZFS Storage Appliance system

To provide the most efficient service to applications, the number of pools should be kept to a minimum, with similar or identical configurations combined. For example, RAID-1 pools should be configured as a single RAID-1 pool, thus allowing the maximum number of cache devices to be configured for that pool. Remember that cache and log devices are exclusively assigned to a pool.

While complying with the best practice to separate the Microsoft SQL Server datatypes and taking the foregoing into consideration, a single pool should be configured with separate LUNs provisioned for data tables, redo logs, and backup areas. Figure 3 shows an example pool and volume configuration.

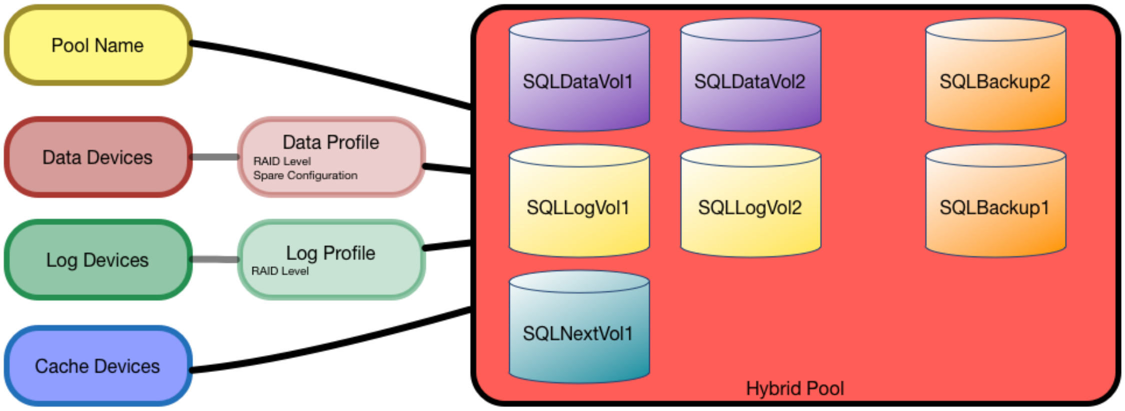

Figure 3. Pool and volume configuration

Setting NTFS Allocation Unit Sizes

When deploying Microsoft SQL Server on block volumes, it is highly advisable to choose the block size based on the typical (or highest precedence) use of the database. For example, an online transaction processing (OLTP) database might potentially require a different configuration from a data warehousing (DW) one. Similarly, a reference (or mostly read-only) database might require a different configuration from a highly volatile one.

In addition, block volumes form the basis for Microsoft new technology file systems (NTFSs). These file systems have an inherent block or allocation size specified when the file systems are formatted.

By default, the volume format operation from Microsoft Windows Server 2008 and later uses the following allocation unit sizes:

TABLE 1. NTFS DEFAULT ALLOCATION SIZES

Volume Size	Allocation Unit Size
7 MB–16 TB	4 KB
16 TB–32 TB	8 KB
32 TB–64 TB	16 K
64 TB–128 TB	32 K
128 TB–256 TB	64 KB

Per Microsoft best practices, choose an allocation unit size of 64 KB for all data and log volumes. If you choose the standard 4 KB allocation unit size, do not enable compression.

Setting Volume Block Sizes

Using the recommended 64 KB allocation size, it follows that the block size of the underlying volume should be a multiple of 64 KB.

The choice of block size is influenced by the most prevalent or highest-priority mode of operation—OLTP or DW—because databases can be a mixture of both types of access during phases of operation, for example, differing during the day or by the date.

OLTP access is characterized by random reads and writes, which update many small areas of the database and rely very heavily on indices, with potentially many users requiring concurrent access. The database can be described as volatile, and the overriding priority in this case is input/output operations per second (IOPS).

Contrasting OLTP with DW workloads, the latter shows predominant access by sequential reads and writes: data is searched in full-table scans, or tables are appended with bulk loads. Index operations tend to be lighter than OLTP operations, with index updates being the most common. These workloads tend to be low-concurrency, and data can be considered nonvolatile, relative to OLTP workloads. DW workloads require more bandwidth (megabytes per second) than IOPS.

Oracle ZFS Storage Appliance systems provide intelligent caching systems and algorithms that allow random I/O requests to be combined, where appropriate, in terms of locality and time. This feature effectively streamlines random I/O requests into as sequential an order as possible within the constraints. When combined with dynamic block size I/O operations, Oracle ZFS Storage Appliance systems mitigate the performance impact of running different forms of workloads on a given storage configuration.

The recommendation for volume block size per workload type is as follows:

TABLE 2. RECOMMENDED VOLUME BLOCK SIZE BY WORKLOAD TYPE

Workload Type	Workload I/O Request Size	Volume Block Size
Online Transaction Processing	8 K–16 K	128 KB/256 KB
Decision Support/Data Warehousing	64 KB–512 KB	1024 KB

Applying Data Compression

Oracle ZFS Storage Appliance systems allow data on disk to be compressed automatically and transparently to the application. Various levels of compression are available, with increasing performance impact as the data compression level increases.

Note: Oracle ZFS Storage Appliance compression should not be used when Microsoft SQL Server row and page compression is deployed; these two compression modes should be considered mutually exclusive.

In practice, the lowest compression level, LZJB, can provide better performance than no compression, thanks to the performance features of Oracle ZFS Storage Appliance systems and the decreased amount of data to be read or written to disk.

The suitability of Oracle ZFS Storage Appliance compression depends on the type of data being stored and the data's compressibility. BLOB storage might not be a candidate for compression, for example, if it is used to store JPEG images, because these are already compressed.

Oracle ZFS Storage Appliance systems apply compression to blocks only if the data blocks compress well, because compressing already-compressed data results in a larger block than the singly-compressed data. In pure textual databases, compression can save a large amount of space.

The best advice for determining compression viability is to run performance tests using Microsoft's SQL Server simulation utility, SQLIOSim or its newer counterpart, DiskSpd, both of which are available for download from Microsoft.

Ensuring LUN Partition Alignment

With larger block sizes, partition alignment becomes important. Partition alignment refers to the degree to which the NTFS data blocks "line up" with the underlying volume block boundaries, as shown in Figure 4.

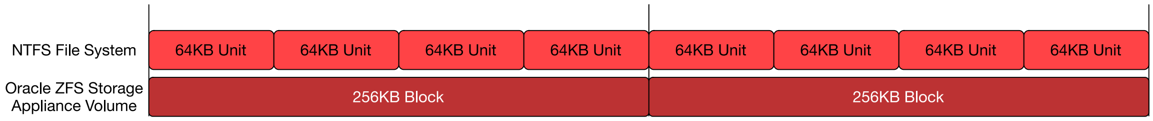

Figure 4. Aligned blocks and units

Misalignment can occur at the logical start of the volume where volume labels are written during the initialization process. For example, misalignment could occur due to the placement of the master boot record (MBR), which contains bootstrap code, a disk signature, the partition table information, an MBR signature, and some padding bytes. This structure is 512 bytes in total length.

When a globally unique identifier partition table (GPT) header is used, as is the case with larger disks, the structure contains a protective MBR, a primary GPT header, and four partition definitions per logical block address (LBA). The partition data resides in LBA34 onward, and a duplicate GPT entry table and a secondary GPT header follow that.

GPT structures are based on LBAs, each consisting of blocks that are the underlying physical sector size of the supporting drive. Typically this is 512 bytes, which is the traditional block size, or 4,096 bytes. The drive block size is different from the volume block size and represents the smallest I/O unit of the physical drive. For block-based LUNs presented by an Oracle ZFS Storage Appliance system, drive block size is set at 512 bytes. The structures are illustrated in Figure 5.

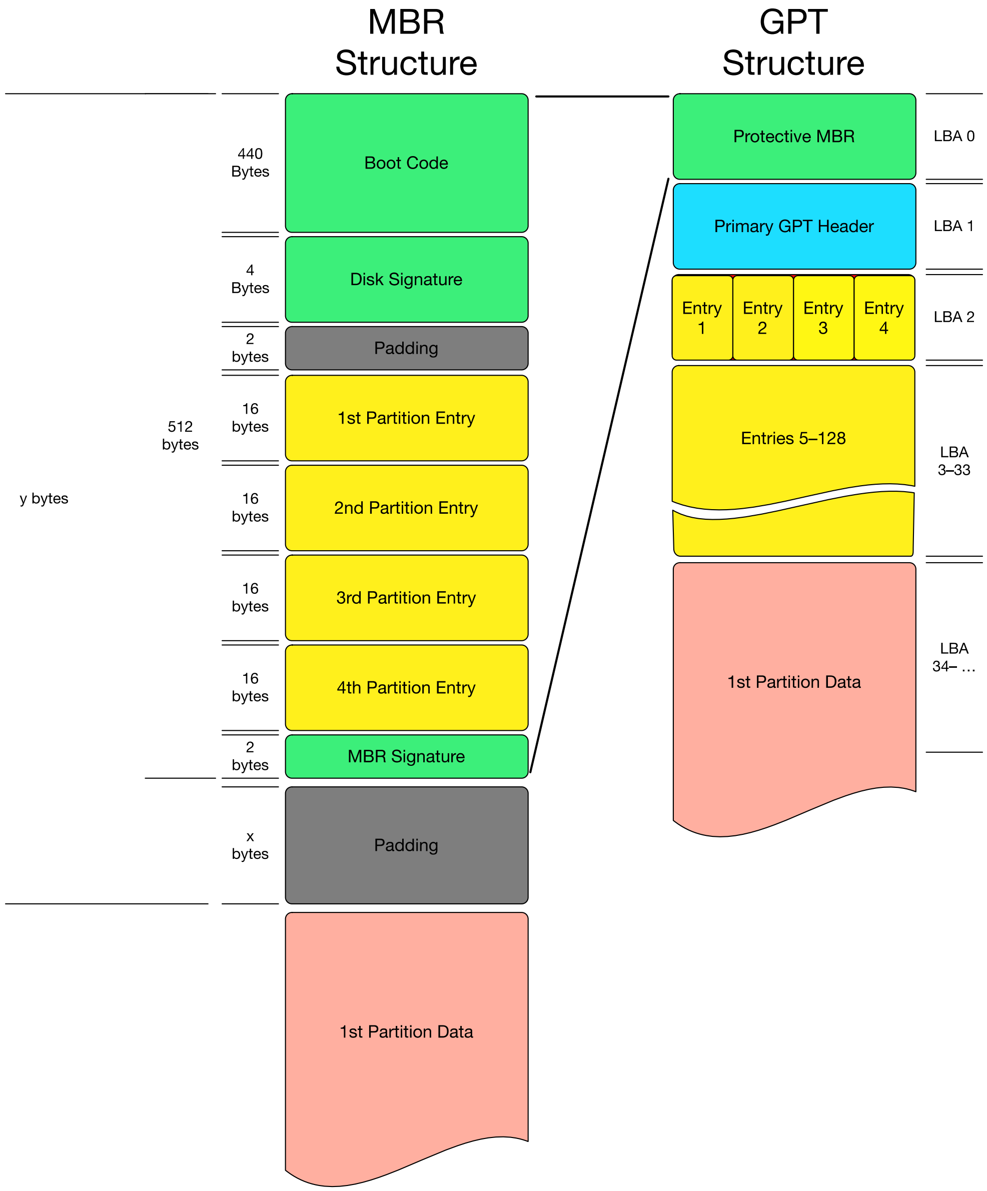

Figure 5. MBR and GPT structures

A native file system placement is at the end of the MBR or GPT structure, which leads to misalignment of the file system allocation unit size against the volume block size. A similar situation could occur with GPT volumes, as shown in Figure 6.

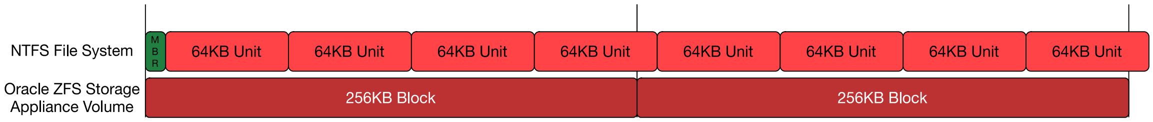

Figure 6. Misaligned partition

To align partitions to underlying blocks, padding bytes are allocated after both the MBR and GPT structures. With Microsoft Windows Server 2008 and later, the first partition starts at 1 MB from the start of the volume.

However, it is possible to see misaligned partitions in Microsoft Windows Server 2008 and later, particularly if the data volumes have been created in an earlier version of Microsoft Windows Server. The performance impact of misaligned partitions can be as much as 30 percent when compared to a properly aligned partition. Therefore, it is worth checking that any created partitions are aligned to the 1 MB mark. This configuration is shown in Figure 7.

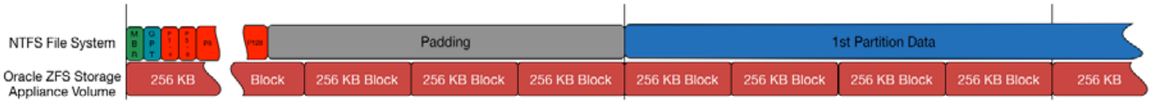

Figure 7. Partition aligned to 1 MB mark

To check the alignment of a partition, use the Windows command-line interface (CLI) utility `wmic` to determine the partition offset from the start of the disk, as shown in the following CLI example.

```
PS C:\Users\Administrator> wmic

wmic:root\cli> partition get Name,BlockSize,StartingOffset

BlockSize  Name                   StartingOffset
512        Disk #0, Partition #0  1048576
512        Disk #0, Partition #1  368050176
512        Disk #2, Partition #0  1048576
512        Disk #3, Partition #0  1048576
512        Disk #4, Partition #0  1048576
```

If the offset is divisible by 1,048,576, the partition is aligned to a megabyte (MB) boundary. In the previous example, the first partition and last three are divisible by 1,048,576 but the second partition is not. Therefore, the first and last three partitions are properly aligned and the second is not.

Partitions can be manually aligned during creation using the `diskpart` utility, and `diskpart` may also be used to verify alignment by using the `list partition` command, as illustrated in the following CLI example.

```
PS C:\Users\Administrator> diskpart

Microsoft DiskPart version 6.3.9600

Copyright (C) 1999-2013 Microsoft Corporation.

On computer: W2K12-R2

DISKPART> list disk

  Disk ###  Status         Size     Free     Dyn  Gpt
  --------  -------------  -------  -------  ---  ---
  Disk 0    Online           84 GB      0 B
  Disk 1    Online           25 GB    25 GB
  Disk 2    Online         1024 MB  1920 KB        *
  Disk 3    Online         2048 MB  1920 KB        *
  Disk 4    Online         4096 MB  4094 MB        *

DISKPART> select disk 4

Disk 4 is now the selected disk.

DISKPART> create partition primary align=1024

DiskPart succeeded in creating the specified partition.

DISKPART> list part

  Partition ###  Type              Size     Offset
  -------------  ----------------  -------  -------
* Partition 1    Primary           4094 MB  1024 KB

DISKPART> format fs=ntfs unit=64k quick label="SQLData1"

  100 percent completed

DiskPart successfully formatted the volume.

DISKPART>
```

More information regarding partition alignment and the performance implications can be found in the Oracle white paper “Aligning Partitions to Maximize Storage Performance.”

General Recommendations

Use the following general configuration guidelines based on Oracle ZFS Storage Appliance best practices and Microsoft recommendations.

Oracle ZFS Storage Appliance DRAM

To provide the best caching facilities to Microsoft SQL Server I/O requests, an Oracle ZFS Storage Appliance system should be configured with the maximum amount of dynamic random access memory (DRAM) possible. Oracle ZFS Storage Appliance systems use advanced cache-management features to satisfy as many requests as possible from DRAM, flash cache or, if necessary, from hard drives under control of a proven operating system.

With a large amount of DRAM installed, an Oracle ZFS Storage Appliance system can satisfy 70 percent to 90 percent of I/O requests from DRAM, thus minimizing I/O latency while still providing the resilience and persistence of a flexible RAID device.

tempdb Database

Per Microsoft’s recommendation, Microsoft's tempdb database should be considered a top-priority database when allocating storage resources, because Microsoft SQL Server uses it internally for many operations. tempdb is a system database that is used as a global resource for temporary user objects, tables, stored procedures, variables, and cursors, among many other uses. tempdb performance impacts the performance of all other databases to which tempdb provides services.

Microsoft recommends that tempdb be situated on its own volumes (data and log) and not shared with any other database files. These tempdb structures should, however, be placed in the common hybrid pool to maximize the available cache resources.

Volume Space Management

Databases should not be set to automatically grow or shrink, but should have their volume space allocated at creation. Allowing dynamic growth and shrinkage of the database can lead to unpredictable performance, because the file modification might take place at an unsuitable time for the database workload. Similarly, underlying Oracle ZFS Storage Appliance LUNs should not be configured as thinly provisioned.

To provide the best performance, file systems built on Oracle ZFS Storage Appliance volumes should not exceed 90 percent capacity. When very large (multi-terabyte) volumes are created, this figure can be increased to 95 percent.

Sample Configurations

To demonstrate the best practices in action, this section describes the configuration suggested for two differing workload types: DW and OLTP.

As always, these configurations should be tested with a benchmarking tool, such as Microsoft SQLIOSim, to ensure that the configuration meets or exceeds the performance requirements for the required priority workload type.

The following tables provide sample configurations for DW workloads and OLTP workloads.

TABLE 3. SAMPLE CONFIGURATION FOR DW WORKLOADS

Parameter	DW Data	DW Log
Pool Data Profile	Mirrored	Mirrored
Pool Log Profile	Striped	Striped
Compression	LZJB	LZJB
Volume Size	As required	As required
Thin Provision	No	No
Block Size	1 MB	1 MB

Parameter	DW Data	DW Log
Encryption	Off	Off
Cache Device Use	Data + metadata	Data + metadata
Synchronous Write Bias	Throughput	Latency
NTFS Allocation Unit	64 K	64 K

TABLE 4. SAMPLE CONFIGURATION FOR OLTP WORKLOADS

Parameter	OLTP Data	OLTP Log
Pool Data Profile	Mirrored	Mirrored
Pool Log Profile	Striped	Striped
Compression	LZJB	LZJB
Volume Size	As required	As required
Thin Provision	No	No
Block Size	512 KB	512 KB
Encryption	Off	Off
Cache Device Use	Data + metadata	Data + metadata
Synchronous Write Bias	Latency	Latency
NTFS Allocation Unit	64 K	64 K

Microsoft SQL Server Backups

The backup utility built into Microsoft SQL Server can be initiated from Microsoft SQL Server Management Studio or from a T-SQL session.

Microsoft SQL Server backups allow tablespaces and indices to be backed up without shutting down the database. When the individual database files are very large and there are multiples of these very large files, Microsoft SQL Server offers the ability to back up these files independently, without the need to back up all files at the same time. In practice, very few implementations use this option, so it will not be covered in this document.

With this in mind, there are still several choices to be made regarding the backup options and the backup destination options. Assuming the backup is destined for storage within an Oracle ZFS Storage Appliance system, the following items should be considered:

- SQL Server `BLOCKSIZE`
- SQL Server `MAXTRANSFERSIZE`
- SQL Server `BUFFERCOUNT`
- SQL Server compression versus Oracle ZFS Storage Appliance compression
- Oracle ZFS Storage Appliance access protocol
- Oracle ZFS Storage Appliance pool data profile

- Oracle ZFS Storage Appliance pool log profile
- Oracle ZFS Storage Appliance data deduplication
- Network infrastructure

Microsoft SQL Server `BLOCKSIZE`

The Microsoft SQL Server `BLOCKSIZE` parameter defines the block size of the output stream from the backup utility. This is distinct from any database block size. This is a highly relevant parameter when the destination of the backup utility is a block device, such as a tape drive, that can also affect the throughput of the backup application and, thus, affects the performance of backups for normal database operation.

The supported block sizes, in bytes, are 512; 1,024; 2,048; 4,096; 8,192; 16,384; 32,768; and 65,536. When the destination is an Oracle ZFS Storage Appliance system, the best backup performance, in general, is achieved with a 64 K (65,536 bytes) `BLOCKSIZE` setting.

Microsoft SQL Server `MAXTRANSFERSIZE`

The `MAXTRANSFERSIZE` parameter defines the maximum chunk of data to be transferred to the backup medium.

Its permitted values are in multiples of 64 K, up to 4,096 K (or 4 MB), with a default of 1 MB. When the backup destination is an Oracle ZFS Storage Appliance system, 2 MB (2,097,152 bytes), in practice, is the most efficient `MAXTRANSFERSIZE` setting.

Microsoft SQL Server `BUFFERCOUNT`

The Microsoft SQL Server `BUFFERCOUNT` parameter defines the total number of buffer slots to be used for the backup operation, where the total size of memory allocated is `BUFFERCOUNT * MAXTRANSFERSIZE`. This is mainly relevant only in small-memory servers, and it should be set such that it does not cause memory starvation in these servers, while also allowing for all running processes. When server memory is not an issue and the destination is an Oracle ZFS Storage Appliance system, 512 buffers are more than sufficient.

Microsoft SQL Server Compression Versus Oracle ZFS Storage Appliance Compression

Microsoft SQL Server offers the ability to compress the data stream from the backup prior to transmitting it to any backup medium. Oracle ZFS Storage Appliance systems also offer the opportunity to compress the data on disk—transparent to the application.

Compression has an impact on CPU utilization on whichever platform undertakes the operation: This might be relevant when CPU resources are at a premium on the server.

In practice, when compression is applied at Microsoft SQL Server, the backup throughput can double, and the backup image can be compressed by a factor of 3.83.

However, when LZ4 compression is applied at the Oracle ZFS Storage Appliance file system level, the backup image is compressed by a factor of 5.83.

When a pattern of full and incremental backups is taken (for example, a weekly full backup and a daily incremental backup), this compression can increase further with the implementation of data deduplication within an Oracle ZFS Storage Appliance system. This can be over 1.5 times the data reduction offered by compression alone, resulting in a total data reduction ratio of 10, but without the added advantage of reducing the backup time.

The choice is between Microsoft SQL Server with low compression and using more disk space for faster backups versus an Oracle ZFS Storage Appliance system with high compression and using much less disk space for slower backups.

Oracle ZFS Storage Appliance Access Protocols

Oracle ZFS Storage Appliance systems offer a number of access protocols to clients: Fibre Channel; iSCSI; SMB versions 1, 2, and 3; NFS versions 2, 3, 4, and 4.1; and InfiniBand.

Predominantly, iSCSI, a block protocol, and SMB, a file protocol, are used for Microsoft SQL Server backups.

iSCSI offers very little performance difference over SMB3.0, and, in a number of circumstances, iSCSI is outperformed by SMB3.0.

One of the major differences between SMB2 and SMB3 is the multichannel SMB feature, which is negotiated at client connection time. Just like iSCSI, it is possible to access volumes (LUNs for iSCSI) for multiple Ethernet connections over multiple common networks. Unlike iSCSI, this is entirely and automatically configured by the client and server at connection time.

For iSCSI, these settings must be manually configured in the iSCSI initiator control panel (for details, see "Using Sun ZFS Storage Appliance iSCSI LUNs in a Microsoft Windows Server 2008 R2 Environment").

However, SMB3 might not be quite the advantage as it first seems, because the SMB3.0 traffic will use all common networks, whether they are designated as data or administration networks. This can be problematic when any administration network is set up as a low-throughput network due to its low incidence of intentional traffic. An example is when a gigabit Ethernet (1 GbE) network is configured for administration, and a 10 GbE network is configured for data.

Fortunately, it is possible to constrain which interfaces on the client to use for a particular server (for details, see "New-SmbMultichannelConstraint").

iSCSI LUNs cannot be shared between clients when using the default Microsoft Windows Server file system (NTFS) and, because NTFS is a block storage protocol, the size of the volume is relatively static. It is, of course, changeable if the change increases the size. However, this requires operations on both the Oracle ZFS Storage Appliance system and the Microsoft Windows Server clients to increase the LUN and file system sizes.

An Oracle ZFS Storage Appliance SMB3.0 share allows a much more efficient use of disk space, which is shared among clients and has very little "waste" compared to iSCSI or FC LUNs.

Oracle recommends using SMB3.0 shares when storing Microsoft SQL Server backups. However, this does not preclude the deployment of iSCSI or FC LUNs for the same purposes when deemed more appropriate.

Network Infrastructure

Regardless of which protocol is chosen to access an Oracle ZFS Storage Appliance system, the network infrastructure can have a major impact on the performance of Microsoft SQL Server backups. For IP-based protocols (whether file- or block-based) and for Fibre Channel protocols, there are a number of technologies available at varying costs for various performance levels.

For example, in addition to the usual 1 GbE technology, there are enhanced technologies that increase the signaling rate to 10 GbE and 40 GbE. Other technologies are available (25 GbE and 100 GbE), but those have yet to see widespread adoption.

40 GbE is effectively four bonded 10 GbE channels. However, just replacing a 10 GbE interface card with a 40 GbE card will not necessarily provide four times the bandwidth of the 10 GbE card, because single- or low-threaded applications cannot make use of the extra lanes available through 40 GbE interfaces.

In practice, 40 GbE can suffer worse performance than 10 GbE, but these are boundary conditions that, with luck, will never be seen in implementation.

Whatever form of interface is chosen, Microsoft SQL Server backups perform best with a high Ethernet maximum transmission unit (MTU) setting. Setting the MTU at 9,000 on both the client and the Oracle ZFS Storage Appliance system can result in an increase in Microsoft SQL Server backup throughput of up to 6 percent on a 10 GbE interface.

The following table provides a sample configuration for deployment of a Microsoft SQL Server and an Oracle ZFS Storage Appliance system, including the network interface settings.

TABLE 5. SAMPLE CONFIGURATION FOR BACKUP

Parameter	Backup
Pool Data Profile	RAIDZ2
Pool Log Profile	Striped
Compression	LZ4
Volume Size	As required
Thin Provision	No
Block Size	64 KB
Encryption	Off
Cache Device Use	Data + metadata
Synchronous Write Bias	Throughput
NTFS Allocation Unit	64 K
Network Interface (client)	10 GbE
Network Interface (Oracle ZFS Storage Appliance)	10 GbE (few clients) or 40 GbE (many clients)
Ethernet MTU	9,000
`BLOCKSIZE`	64 K
`MAXTRANSFERSIZE`	2 MB
`BUFFERCOUNT`	Client-memory dependent

Conclusion

Oracle ZFS Storage Appliance systems have a storage architecture that automatically optimizes their use of DRAM, flash cache, and disk accesses to provide a highly flexible platform on which businesses can host their databases, mail systems, and virtual desktop infrastructure in an efficient and cost-effective way.

By providing features such as on-disk compression, encryption, and thin provisioning, Oracle ZFS Storage Appliance systems play a key role in supporting business processes and policies so you can simplify your storage infrastructure while maintaining the same or greater level of functionality, thus reducing costs and improving efficiency. That high efficiency can be maintained by following the best practices described in this paper when you configure a Microsoft SQL Server and an Oracle ZFS Storage Appliance system for deployment.

References

See the following resources for additional product information:

- Oracle ZFS Storage Appliance white papers and subject-specific resources, including “Aligning Partitions to Maximize Storage Performance”
- Oracle ZFS Storage Appliance product information
- Oracle ZFS Storage Appliance documentation library, including installation, analytics, customer service, and administration guides

  The Oracle ZFS Storage Appliance documentation is also available through the Oracle ZFS Storage Appliance help function, which can be accessed through the browser user interface.

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

CONNECT WITH US

- blogs.oracle.com/oracle
- facebook.com/oracle
- twitter.com/oracle
- oracle.com

Integrated Cloud Applications & Platform Services

Microsoft SQL Server with Oracle ZFS Storage Appliance: Deployment Best Practices
April 2017, Version 2.0
Author: Andrew Ness