Disk Partition Alignment Best Practices for SQL Server

Writers: Jimmy May, Denny Lee

Contributors: Mike Ruthruff, Robert Smith, Bruce Worthington, Jeff Goldner, Mark Licata, Deborah Jones, Michael Thomassy, Michael Epprecht, Frank McBath, Joseph Sack, Matt Landers, Jason McKittrick, Linchi Shea, Juergen Thomas, Emily Wilson, John Otto, Brent Dowling

Technical Reviewers: Mike Ruthruff, Robert Smith, Bruce Worthington, Emily Wilson, Lindsey Allen, Stuart Ozer, Thomas Kejser, Kun Cheng, Nicholas Dritsas, Paul Mestemaker, Alexei Khalyako, Mike Anderson, Bong Kang

Published: May 2009

Applies to: SQL Server 2008

Summary: Disk partition alignment is a powerful tool for improving SQL Server performance. Configuring optimal disk performance is often viewed as much art as science. A best practice that is essential yet often overlooked is disk partition alignment. Windows Server 2008 attempts to align new partitions out-of-the-box, yet disk partition alignment remains a relevant technology for partitions created on prior versions of Windows.

This paper documents performance for aligned and nonaligned storage and why nonaligned partitions can negatively impact I/O performance; it explains disk partition alignment for storage configured on Windows Server 2003, including analysis, diagnosis, and remediation; and it describes how Windows Server 2008 attempts to remedy challenges related to partition alignment for new partitions yet does not correct the configuration of preexisting partitions.

The following topics are also included: background information, implementation, vendor considerations, two essential correlations, valid starting partition offsets, and the simple protocol to align partitions, define file allocation unit size, and assign drive letters. It includes results from tests that show how partition alignment can affect performance for SQL Server 2008.

You can also download a Microsoft Word version of this article [http://download.microsoft.com/download/C/E/7/CE7DA506-CEDF-43DB-8179-D73DA13668C5/DiskPartitionAlignment.docx] .

Introduction

Configuring optimal disk performance is often viewed as much art as science. Yet an understanding of disk performance best practices can result in significant improvements. Some of the many factors that affect disk I/O performance include the number, size, and speed of disks; file allocation unit size; configuration of host bus adapters (HBAs) and fabric switches; network bandwidth; cache on disks, controllers, and storage area networks (SANs); whether disks are dedicated, shared, or virtualized; RAID levels; bus speed; number of paths from disk I/O subsystem to servers; driver versions for all components; factors related to RAID stripes sizes; and workload.

A best practice that is essential yet often overlooked is disk partition alignment.

The Windows Server® 2008 operating system attempts to align new partitions out of the box, yet disk partition alignment remains a relevant technology for partitions created on prior versions of the Windows® operating system.

Noncompliance with storage configuration best practices for the Microsoft® SQL Server® database software is a common root cause of support cases. The reason is often shown to be misalignment between Windows, storage, disk controllers, and cache segment lines.

Failure to perform partition alignment may result in significant performance degradation. Disk partition alignment

is a requirement for partitions from which high performance is demanded and that were created on RAID disk devices on versions of Windows prior to Windows Server 2008. As we will explain, new partitions on Windows Server 2008 are created at a starting offset which (with high probability) aligns with underlying RAID stripe units.

For systems from which high performance is required, it is essential to experiment with representative workloads and determine the validity of disk partition alignment for your environment.

Unless performed at the time of partition creation, the default alignment offset will result in unaligned partitions on versions of Windows up to and including Windows Server 2003. These versions of Windows create disk partitions by default to boundaries based on the Cylinder/Head/Sector (CHS) addressing scheme used by previous generation disk controllers. Preexisting partitions attached to Windows Server 2008 maintain the original, flawed alignment under which they were created.

This paper documents performance for aligned and nonaligned storage and why nonaligned partitions can negatively impact I/O performance; it explains disk partition alignment for storage configured on Windows Server 2003, including analysis, diagnosis, and remediation; and it describes how Windows Server 2008 attempts to remedy challenges related to partition alignment for new partitions yet does not correct the configuration of preexisting partitions.

The following topics are also included:

- Background information
- Implementation
- Vendor considerations
- Two essential correlations
- Valid starting partition offsets

The paper also covers the simple protocol to align partitions, define file allocation unit size, and assign drive letters.

Scope

The information presented here applies to Windows basic and dynamic disks with master boot record (MBR) partitions.

Details related to GUID partition table (GPT) disks are not addressed. However, disk partition alignment is a best practice, and it is required for optimal performance for each of these hard drive configurations:

- MBR basic
- MBR dynamic
- GPT basic
- GPT dynamic

Performance characteristics of dynamic disks and GPT disks will be addressed in subsequent publications on the SQL Server Customer Advisory Team Web site [http://www.sqlcat.com/default.aspx] (www.sqlcat.com).

Background Information

You may hear the terms *partition alignment, disk alignment, volume alignment*, and *sector alignment* used synonymously. This paper uses the term *partition alignment*.

Many customers are unaware of partition alignment. Even experienced disk administrators may be unfamiliar with it. Explanations are often initially met with disbelief.

Engineers familiar with the topic may underestimate its importance. For example, some customers think it is useful only for Microsoft Exchange Server. In fact, partition alignment is important for all servers from which high performance is expected—especially SQL Server.

This section presents essential terms and history.

Terms

Discussion of disk I/O includes terms that are often used ambiguously and interchangeably—sometimes reflecting accepted definitions, sometimes not. A common framework for communication is important. Hard drives contain

many components; the important ones for this discussion are labeled in Figure 2 in the "Appendix". Figures 3 and 4 in the same section provide graphical documentation.

Hard drives house *platters*, one or more thin, circular disks, on the surfaces of which are the electronic media that store information.

Each side of each platter has thousands of tracks. The set of tracks with the same diameter from all platter surfaces comprises a *cylinder*. (For modern drives, the cylinder concept is no longer relevant, and tracks are no longer arranged in concentric circles, yet it is useful to understand the origin of the terms.) Each platter surface has a dedicated read/write head. Tracks are divided into sectors. A sector is the minimum chunk of data that can be read or written to a hard drive. Historically, sector size has been fixed at 512 bytes. Newer drives may offer 1 KB, 2 KB, or 4 KB sectors.

Many engineers will recognize file allocation unit by another name: cluster. Cluster size is determined when the partition is formatted by the operating system (or the user). For example, if the sectors of a hard drive are 512 bytes, a 4 KB cluster has 8 sectors, and a 64 KB cluster has 128 sectors.

Stripe size is the size of one entire stripe spread across all the disks in a RAID-0, RAID-10, RAID-5, or RAID-6 disk group. Stripe unit size is the size of each element of the stripe, as stored on each member of the disk group. Stripe size is a product of the stripe unit size and the number of disks in a RAID group. Stripe unit size is the attribute of a RAID disk group that can be configured by administrators. A stripe unit is the collection of bits on each disk exactly equal to the stripe unit size.

Description

Understanding the nuances of partition alignment is not necessary to follow the simple protocol required for optimal alignment. For more information about how to execute partition alignment, see the "Implementation" section later in this paper.

The following is a simplified characterization of partition misalignment: Disk array hardware reports 63 reserved (hidden) sectors, and Windows obediently implements this information, reserving these sectors at the beginning of the first partition of a disk. The master boot record (MBR) resides within these hidden sectors. The compliance by Windows with 63 hidden sectors reported by disk hardware results in misalignment with stripe units, disk controllers, and cache segment lines. In all versions of Windows earlier than and including Windows XP and Windows Server 2003, these reserved sectors do not coincide with fundamental physical boundaries. The result is that single clusters of user data are written across multiple stripe units of the RAID. Every nth operation is affected (n depends on file allocation unit (cluster) size and stripe unit size). Fundamental physical boundaries in disk controllers and other hardware are also violated.

Across a striped array, a single I/O coming into the array controller turns into multiple I/Os if the request crosses one or more stripe unit boundaries. The cumulative effect can contribute to substantial performance degradation. For more information about the effect in experimental and production environments, see the "Performance Impact" section later in this paper.

In all cases, similar principals are at work: Due to misalignment, clusters of data are written across physical boundaries, requiring unintended, unnecessary I/Os that result in performance degradation.

In the absence of an explicit vendor recommendation, use a partition offset that complies with the correlation discussed in the section "Essential Correlations: Partition Offset, File Allocation Unit Size, and Stripe Unit Size". 64 KB is a common, valid starting partition offset because it correlates well with fundamental physical boundaries in disks, controllers, and cache. Other valid starting partition offsets exist. The Windows Server 2008 default is 1024 KB. For more information, see the "Valid Starting Partition Offsets" section later in this paper.

Partition Alignment in Windows Operating Systems

The way partition alignment works depends on the version of Windows being used and the version in which the partition alignment was created. The following sections describe how partition alignment works in Windows Server 2008, the Windows Vista® operating system, and Windows Server 2003 and earlier.

Windows Server 2008 and Windows Vista: New Partitions

In Windows Vista as well as Windows Server 2008, partition alignment is usually performed by default. The default for disks larger than 4 GB is 1 MB; the setting is configurable and is found in the registry at the following location:

HKLM\SYSTEM\CurrentControlSet\Services\VDS\Alignment

However, if OEM setups are delivered (for example, with recovery partitions), even fresh installations of Windows Server 2008 having partitions with undesirable partition starting offsets have been observed.

Whatever the operating system, confirm that new partitions are properly aligned.

Windows Server 2008: Pre-existing Partitions

New partitions on Windows Server 2008 are likely to be aligned. Yet partitions created on earlier versions of Windows and become associated with Windows Server 2008 maintain the properties under which they were created. That is, in the absence of partition alignment being explicitly performed, these partitions are not aligned.

Windows Server 2003 and Earlier

Partitions created on versions of Windows up to and including Windows Server 2003 by default are not aligned. Partition alignment must be explicitly performed.

System Drives

System drives in versions of Windows prior to Windows Server 2008 cannot be aligned. Fortunately, workloads associated with system partitions of dedicated SQL Server computers are typically not as sensitive to partition misalignment as disks dedicated to I/O intensive uses, for example, SQL Server database files from which high-performance is demanded. As described in the "Appendix", modern disks are proprietary "black boxes" and disk partition alignment does not enhance performance for individual disks; however, cache line misalignment may still apply. Make certain that performance thresholds, particularly disk latency, are not exceeded for all disks, including those on which the operating system and SQL Server data and log files reside.

System drives on fresh installations of Windows Server 2008 should be aligned by default.

Virtual Drives

Virtual drives *and* the host drives on which they reside must be aligned for optimal performance. The guidelines described here apply to the respective guest & host operating systems.

Description

Understanding the nuances of partition alignment is not necessary to follow the simple protocol required for optimal alignment. For more information about how to execute partition alignment, see the "Implementation" section later in this paper.

Performance Impact

In the detailed experiment below, alignment reduced both disk latency and query duration by approximately 30 percent. The performance of six aligned disks was comparable to or better than eight nonaligned disks.

This work was done on a DELL PowerEdge 2950 with two dual core 3.00 GHz Intel Xeon processors, a PERC 5/E controller, and 8 GB of physical RAM. Six or eight disks, SAS DAS, 73 GB 15K RPM, were configured in RAID 10 with a cluster size of 64 KB and a stripe unit size of 64 KB. Windows Server 2003 and SQL Server 2005 were installed. A query was executed that the customer used to benchmark performance. Prior to each run, DBCC DROPCLEANBUFFERS was executed to clear the SQL Server buffer cache, which ensured that all data required to satisfy the workload was loaded from disk.

Data was collected for disk latency, duration, and other relevant metrics. The Avg. Disk Transfers/sec counters of the **PhysicalDisk** and **LogicalDisk** performance objects were used to measure disk latency. Disk latency is a fundamental measure of disk performance.

The experiment was simple, yet convincing. The results were consistent and significant. Figure 1 documents the outcome.

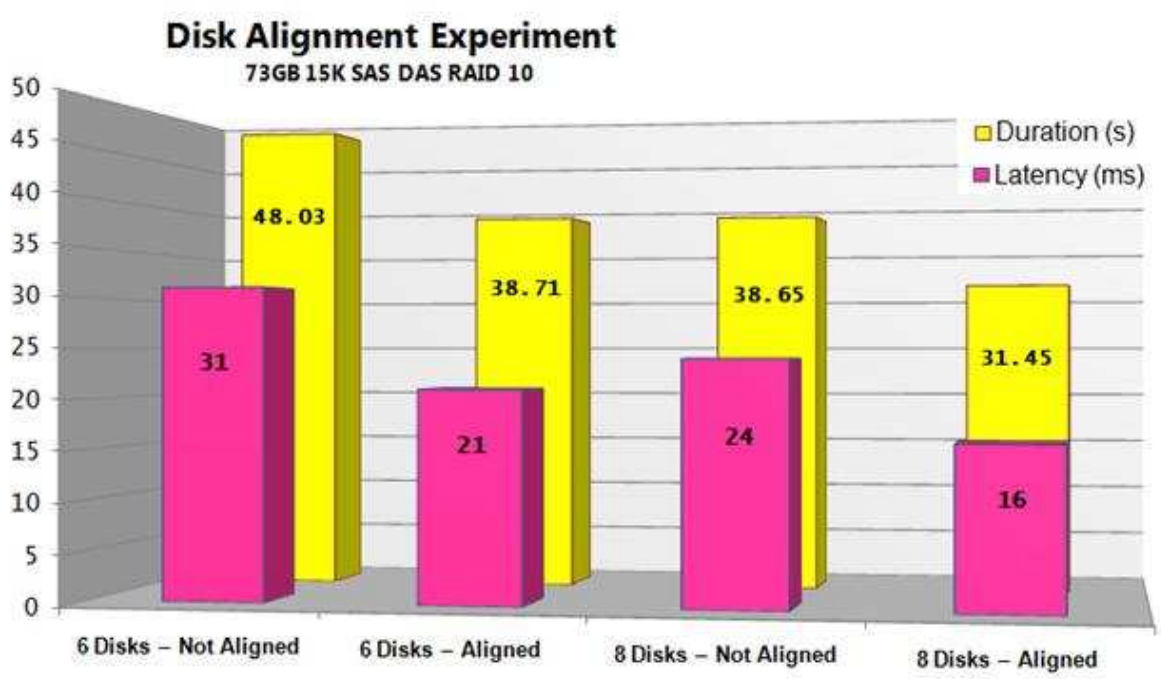

Figure 1: Results of an experiment documenting the benefits of disk partition alignment

Analysis resulted in the following conclusions:

- Disk alignment produced significant improvement compared to nonaligned disks. The measurements document enhancements in excess of 30% for disk latency and duration.
- The performance of six aligned disks was comparable to or better than eight nonaligned disks.

Not displayed is that partition alignment also increased throughput (bytes/sec) and reduced disk queues. CPU differences were insignificant.

Other Examples

The following table summarizes experimental efforts as well as results at other customer sites. While these results are not comprehensive, their purpose is to provide evidence that disk partition alignment enhances performance under a wide variety of workloads and hardware configurations. Based on the work described here, customers had reconfigured substantial amounts of data. For example, a major financial firm rebuilt the SQL Server disk infrastructure that hosted their SharePoint® services, and a major telecom firm rebuilt multiple terabytes of data across their enterprise.

Scenario	Performance Enhancement
Workload on new OLTP production system involving table scans, and UPDATE and SELECT statements.	30% in terms of latency and duration as noted in Figure 1.
Enterprise Hitachi AMS SAN storage array.	8% - 12% for average workloads.
SQLIO on a SAP database server: Using 8-core server with 16GB memory on an EMC Symmetrix SAN using Veritas managed dynamic volumes.	Random 8 KB writes: 18% Sequential 64 KB writes: 26% Sequential 256 KB writes: 23%.

Financial business intelligence (BI) data warehouse workload: Using a 4-core server with 16 GB of memory on a CX380 SAN.	34% for random reads.
Financial BI data warehouse workload: Using a 4-core server with 16 GB of memory on a CX380 SAN for a BI data warehouse workload.	30% - 40% increase in I/Os per second with 20% improvement in terms of query time. The key improvements were that upon implementing the disk partition alignment, their BI workload could handle more than three times the users and a 350% increase in requests per second.
Major telecom firm: Tested against both BI and OLTP read workloads using a 4 dual core processors with 32 GB of memory in an A/A/P environment to utilize all memory. Disk was set to RAID 5 in this scenario.	I/Os: 40% Maintenance tasks requiring four hours were reduced to 90 minutes One million row inserts into a 140-million row table: From 6:59 to 0:52 seconds. Inserts into tables with 500 million rows: 30%.

Table 1: Summary of impact of disk partition alignment in experimental and production environments

Performance penalties due to misalignment up to 90 percent on writes to RAID 5 have been observed. This varies widely and generalizations are challenging to make.

Owing to SQL Server read-ahead, the affect of misalignment on sequential I/O is not as serious as it is on random I/O.

Starting Partition Offsets

The performance benefit of disk partition alignment requires configuration of valid starting partition offsets.

Several tools report the starting partition offset. The results are reliable only in specific contexts. Valid starting partition offsets and tools used to report them are discussed in this section.

Valid Starting Partition Offsets

Because versions of Windows earlier than and including Windows Server 2003 comply with the 63 hidden sectors reported by disk hardware, and because the most common sector size 512-byte sectors, the default (and suboptimal) starting partition offset is 32,256 bytes, exactly 31.5 KB.

Explicitly defining the starting offset from 31.5 KB to exactly 32 KB might seem like a legitimate approach. In fact, as mentioned earlier, 64 KB is typically the minimum (and a common) valid starting partition offset for SQL Server because of the correlations described later.

When choosing a valid partition starting offset, refer first to your storage vendor best practices. Make certain their recommendations correlate with the stripe unit size and file allocation unit size configured for SQL Server. In the absence of definitive vendor information, choose the Windows Server 2008 default.

Windows Server 2008 partition alignment defaults to 1024 KB (that is, 1,048,576 bytes). This value provides a durable solution. It correlates well (as described later) with common stripe unit sizes such as 64 KB, 128 KB, and 256 KB as well as the less frequently used values of 512 KB and 1024 KB. Also, the value virtually guarantees hidden structures allocated by storage hardware are skipped.

Basic Disk Partition Offsets: wmic.exe

Windows can be interrogated for disk-related information via Windows Management Instrumentation (WMI). A straightforward method for obtaining partition starting offsets of Windows basic disks is this Wmic.exe command.

Command Line Syntax Copy Code

```
wmic partition get BlockSize, StartingOffset, Name, Index
```

The value for **Index** is the same as disk number in the Disk Management Microsoft Management Console (MMC) snap-in (Diskmgmt.msc); **wmic volume** can also be used to map disk indexes and drive letters.

Dynamic Disk Partition Offsets: dmddiag.exe -v

The command-line utility Dmdiag.exe is used to determine the partition offsets of Windows *dynamic volumes*.

Important: Neither the output of the **wmic** command listed earlier nor any other tool designed only for basic disks reliably reports starting partition offsets of Windows dynamic disks.

The tool is available in the support tools folder of Windows Server 2003. In Windows Server 2008, the tool has been renamed diskdiag.exe.

To determine the starting partition offset of dynamic disks, execute the following command.

Command Line Syntax Copy Code

```
dmdiag -v
```

The output has several sections; the sections that are cogent to analyzing the starting partition offset of existing volumes are generated only if the -v switch is used. Those sections (and relevant columns) are:

Dynamic Disk Information (Rel Sec)

LDM Volume Information (Rel Sectors)

Focus on these sections and columns. Subsequent sections of the output that report offsets can be misleading, and they are unlikely to be reliable for interpreting starting partition offsets of dynamic volumes.

Note that Microsoft tools, including even dmdiag, may be unreliable for reporting starting partition offsets of dynamic volumes created by third-party vendors. For example, dmdiag does not report correct starting partition offsets of dynamic volumes created by Veritas Enterprise Administrator. In these cases, consult your vendor for the tools and techniques required for proper analysis.

Command-Line Partition Alignment Tools: Diskpart.exe and Diskpar.exe

Disk partition alignment is not available from the Disk Management snap-in (diskmgmt.msc).

Windows provides two tools to implement disk partition alignment: diskpart.exe and diskpar.exe.

The Windows 2000 Resource Kit introduced the command-line utility diskpar.exe. Its successor, diskpart.exe, was introduced in Windows Server 2003. Note the presence or absence of a "t" in their names. The **/align** option debuted in Windows Server 2003 Service Pack 1 (SP1). Both utilities are powerful and should be exercised with caution.

Diskpar.exe reliably reports partition alignment in terms of bytes. However, results are valid only for MBR basic disks, and this tool is no longer supported by Microsoft.

Diskpart.exe reports alignment for basic disks in terms of kilobytes. As noted, the Windows Server 2003 (and earlier) default alignment is 32,256 bytes, exactly 31.5 KB; unfortunately DiskPart rounds this up to 32 KB. Though DiskPart is the tool of choice to implement partition alignment, the value it reports for partition offset is not sufficiently granular. Therefore, use the **wmic** command to report partition offsets of basic disks; use **dmdiag –v** for Windows dynamic disks.

Implementation

This section provides information about using specific measurements and tools to implement the recommendations presented earlier in this white paper. It also addresses issues to consider regarding vendors.

Essential Correlations: Partition Offset, File Allocation Unit Size, and Stripe Unit Size

Use the information in this section to confirm the integrity of disk partition alignment configuration for existing partitions and new implementations.

There are two correlations which when satisfied are a fundamental precondition for optimal disk I/O performance. The results of the following calculations must result in an integer value:

Partition_Offset ÷ Stripe_Unit_Size

Stripe_Unit_Size ÷ File_Allocation_Unit_Size

Of the two, the first is by far the most important for optimal performance. The following demonstrates a common misalignment scenario: Given a starting partition offset for 32,256 bytes (31.5 KB) and stripe unit size of 65,536 bytes (64 KB), the result is 0.4921875. This is not an integer; therefore the offset & strip unit size are not correlated. This is consistent with *mis*alignment.

However, a starting partition offset of 1,048,576 bytes (1 MB) and a stripe unit size of 65,536 bytes produces a result of exactly 8, an exact integer, which is consistent with alignment.

Note that file allocation unit (cluster) size commonly correlates with common stripe unit sizes. The performance question here is usually not one of correlation per the formula, but whether the cluster size is the NTFS default of 4,096 bytes or has been explicitly defined at 64 KB, which is a best practice for SQL Server.

Stripe Unit Size

Windows does not have a reliable way to determine stripe unit sizes. These values are obtained from vendor disk management software or from your SAN administrator.

File Allocation Unit Size

Run this command for each drive to see the file allocation unit size reported in bytes per cluster.

Command Line Syntax Copy Code

```
fsutil fsinfo ntfsinfo c:
fsutil fsinfo ntfsinfo d:
etc...
```

The Bytes Per Cluster value, which contains the file allocation unit size, is highlighted here.

Command Line Syntax Copy Code

```
D:\>fsutil fsinfo ntfsinfo b:
NTFS Volume Serial Number :       0xa2060a7f060a54a7
Version :                         3.1
Number Sectors :                  0x00000000043c3f5f
Total Clusters :                  0x000000000008787e
Free Clusters  :                  0x000000000008746e
Total Reserved :                  0x0000000000000000
Bytes Per Sector  :               512
Bytes Per Cluster :               65536
Bytes Per FileRecord Segment    : 1024
Clusters Per FileRecord Segment : 0
Mft Valid Data Length :           0x0000000000010000
Mft Start Lcn  :                  0x000000000000c000
Mft2 Start Lcn :                  0x0000000000043c3f
Mft Zone Start :                  0x000000000000c000
Mft Zone End   :                  0x000000000001cf20
```

An appropriate value for most installations should be 65,536 bytes (that is, 64 KB) for partitions on which SQL Server data or log files reside. In many cases, this is the same size for Analysis Services data or log files, but there are times where 32 KB provides better performance. To determine the right size, you will need to do

performance testing with your workload comparing the two different block sizes.

Using Diskpart.exe to Perform Partition Alignment, Assign Drive Letters, and Assign File Allocation Unit Size

The good news is that partition alignment is simple to perform; the bad news is that partition alignment must be done at partition creation time, prior to partitions being formatted. This is great if you have a new SAN, but it might be painful to convert large amounts of existing data on misaligned partitions.

Caution: Realigning partitions using any tool is a destructive operation, because it wipes out existing data.

Here is a template for using diskpart.exe to perform partition alignment, assign a drive letter, and format a partition.

Command Line Syntax Copy Code

```
Diskpart
list disk
select disk <DiskNumber>
create partition primary align=<Offset_in_KB>
assign letter=<DriveLetter>
format fs=ntfs unit=64K label="<label>" nowait
```

Here is an example in which the F: drive is created on disk 3, aligned with an offset of 1,024 KB, and formatted with a file allocation unit (cluster) size of 64 KB.

Command Line Syntax Copy Code

```
C:\>diskpart
Microsoft DiskPart version 6.0.6001
Copyright (C) 1999-2007 Microsoft Corporation.
On computer: ASPIRINGGEEK
DISKPART> list disk
  Disk ###  Status      Size     Free     Dyn  GPT
  --------  ----------  -------  -------  ---  ---
  Disk 0    Online       186 GB      0 B
  Disk 1    Online       100 GB      0 B
  Disk 2    Online       120 GB      0 B
  Disk 3    Online       150 GB   150 GB
DISKPART> select disk 3
Disk 3 is now the selected disk.
DISKPART> create partition primary align=1024
DiskPart succeeded in creating the specified partition.
DISKPART> assign letter=F
DiskPart successfully assigned the drive letter or mount point.
DISKPART> format fs=ntfs unit=64K label="MyFastDisk" nowait
```

Note that **nowait** initiates asynchronous formatting, so that the formatting of multiple partitions can begin one after another, allowing the operations to proceed in parallel. The format option is not available in Windows Server 2003.

Diskpart can be executed directly from the command line or scripted. The /s switch specifies an input script file.

Vendor Considerations

For many vendors, disk partition alignment under Windows Server 2003 is a documented best practice.

Other vendors claim that partition alignment is not a required optimization. For example, one vendor states that partition alignment is not necessary, adding that it "neither enhances nor detracts from [SAN] sequential performance". The claims are intriguing, but corroborating data is lacking or in dispute. The statement explicitly cites sequential I/Os, but it fails to address random I/Os, optimal performance of which is important for OLTP databases and Analysis Services databases.

Consult your storage vendor for their recommendation. In the absence of definitive information, consider implementing disk partition offset with the Windows Server 2008 default of 1024 KB.

For systems from which high performance is required, it is essential to experiment with representative workloads and determine the validity of disk partition alignment for your environment.

Conclusion

Many factors contribute to optimal disk I/O performance. For disk partitions created with Windows Server 2003, partition alignment properly correlated with stripe unit size and file allocation unit size is a best practice, and it provides a fundamental foundation for optimal performance.

Windows Server 2008 aligns partitions by default. When servers are upgraded to Windows Server 2008, preexisting partitions are not automatically aligned and must be rebuilt for optimal performance. Thus, until existing misaligned partitions created using Windows Server 2003 or Windows 2000 Server are rebuilt properly, disk partition alignment will remain a relevant technology.

Check existing disks, be mindful of the differences in analysis of basic partitions and dynamic volumes, rebuild where possible and appropriate, and create all new partitions using the best practices described here.

For more information:

http://www.microsoft.com/sqlserver/ [http://www.microsoft.com/sqlserver/default.aspx] : SQL Server Web site

http://technet.microsoft.com/en-us/sqlserver/ [http://technet.microsoft.com/en-us/sqlserver/default.aspx] : SQL Server TechCenter

http://msdn.microsoft.com/en-us/sqlserver/ [http://msdn.microsoft.com/en-us/sqlserver/default.aspx] : SQL Server DevCenter

http://support.microsoft.com/default.aspx?scid=kb;en-us;923076&sd=rss&spid=3198
[http://support.microsoft.com/default.aspx?scid=kb;en-us;923076&sd=rss&spid=3198] : "An updated version of the Disk Partition tool for Windows Server 2003 is available" (Microsoft Knowledge Base article)

http://www.microsoft.com/technet/prodtechnol/sql/bestpractice/pdpliobp.mspx
[http://www.microsoft.com/technet/prodtechnol/sql/bestpractice/pdpliobp.mspx] : "Predeployment I/O Best Practices" (Microsoft TechNet Best Practices article)

http://www.microsoft.com/technet/prodtechnol/sql/2005/physdbstor.mspx
[http://www.microsoft.com/technet/prodtechnol/sql/2005/physdbstor.mspx] : "Physical Database Storage Design" (Microsoft TechNet article)

http://www.microsoft.com/whdc/archive/subsys_perf.mspx
[http://www.microsoft.com/whdc/archive/subsys_perf.mspx] : "Disk Subsystem Performance Analysis for Windows" (Microsoft white paper)

http://blogs.msdn.com/jimmymay/archive/2009/05/08/disk-partition-alignment-sector-alignment-make-the-case-with-this-template.aspx [http://blogs.msdn.com/jimmymay/archive/2009/05/08/disk-partition-alignment-sector-alignment-make-the-case-with-this-template.aspx] : "Disk Partition Alignment (Sector Alignment): Make the Case: Save Hundreds of Thousands of Dollars" (MSDN blog post)

http://blogs.msdn.com/jimmymay/archive/2008/10/14/disk-partition-alignment-for-sql-server-slide-deck.aspx
[http://blogs.msdn.com/jimmymay/archive/2008/10/14/disk-partition-alignment-for-sql-server-slide-deck.aspx] : "Disk Partition Alignment (Sector Alignment) for SQL Server: Part 1: Slide Deck" (MSDN blog post)

http://blogs.msdn.com/jimmymay/archive/2008/12/04/disk-partition-alignment-sector-alignment-for-sql-server-part-4-essentials-cheat-sheet.aspx [http://blogs.msdn.com/jimmymay/archive/2008/12/04/disk-partition-alignment-sector-alignment-for-sql-server-part-4-essentials-cheat-sheet.aspx] : "Disk Partition Alignment (Sector Alignment) for SQL Server: Part 4: Essentials" (MSDN blog post)

http://sqlblog.com/blogs/joe_chang/archive/2008/03/04/storage-performance-for-sql-server.aspx
[http://sqlblog.com/blogs/joe_chang/archive/2008/03/04/storage-performance-for-sql-server.aspx] : "Storage Performance for SQL Server" (blog post)

http://blogs.msdn.com/mssqlisv/default.aspx [http://blogs.msdn.com/mssqlisv/default.aspx] : SQL ISV blog (Microsoft SQL ISV Program Management Team)

http://blogs.msdn.com/sqlcat [http://blogs.msdn.com/sqlcat.aspx] : SQL Server CAT Blog (Microsoft SQL Server Development Customer Advisory Team)

http://msdn.microsoft.com/en-us/sqlserver/bb671432.aspx [http://msdn.microsoft.com/en-us/sqlserver/bb671432.aspx] : SQL Server Best Practices (MSDN site)

http://www.microsoft.com/sql/alwayson/default.mspx [http://www.microsoft.com/sql/alwayson/default.mspx] : SQL Server 2005: High Availability (SQL Server AlwaysOn Partner Program, Microsoft SQL Server site)

http://technet.microsoft.com/en-us/library/cc766465.aspx [http://technet.microsoft.com/en-us/library/cc766465.aspx] : "DiskPart Command-Line Options" (TechNet article)

http://www.microsoft.com/downloads/details.aspx?familyid=BCB9100D-698F-40A3-BF53-692D793C6E4F&displaylang=en [http://www.microsoft.com/downloads/details.aspx?familyid=BCB9100D-698F-40A3-BF53-692D793C6E4F&displaylang=en] : "Microsoft IT Showcase: Volume Expansion Using Diskpart.exe" (Microsoft TechNet IT Showcase article)

Did this paper help you? Please give us your feedback. Tell us on a scale of 1 (poor) to 5 (excellent), how would you rate this paper and why have you given it this rating? For example:

- Are you rating it high due to having good examples, excellent screen shots, clear writing, or another reason?
- Are you rating it low due to poor examples, fuzzy screen shots, or unclear writing?

This feedback will help us improve the quality of white papers we release.

Send feedback [mailto://microsoft.com:25/default.aspx?subject=White%20Paper%20Feedback:%20Disk%20Partition%20Alignment%20Best%20Practices%20for%20SQL%20Server] .

Appendix: Disk Partitioning Alignment Internals

As stated previously, an understanding of the nuances of partition alignment is not necessary to follow the simple protocol required for optimal alignment. For more information, see the "Implementation" section.

Contemporary hard drives are so complex as to elude description. However, the following graphic is a simplified characterization of HDD internals. For more information, see the "Terms" section.

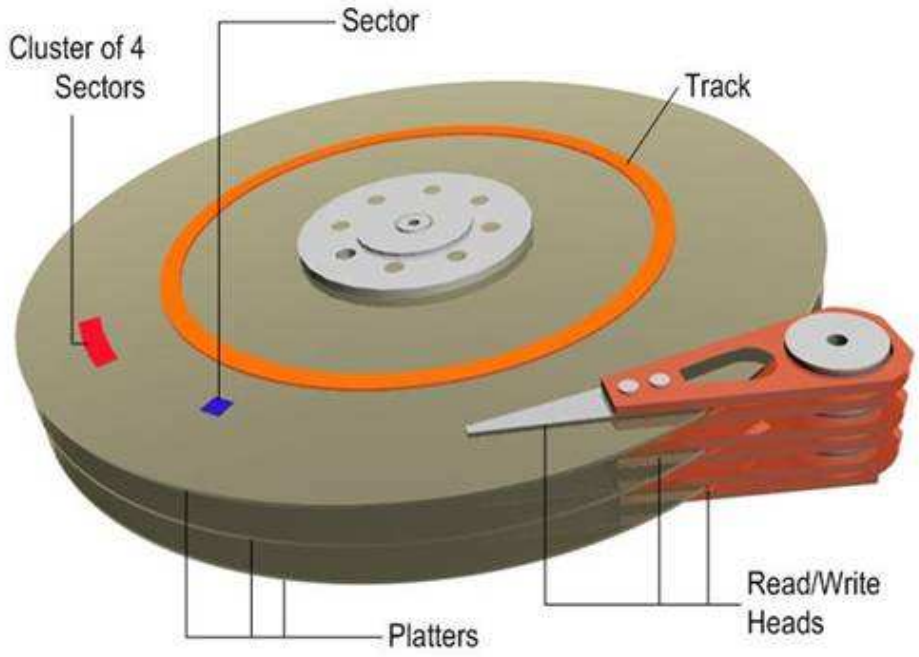

Figure 2. Simplified schematic diagram of hard drive internals

Modern disk structures are so complex even by industry experts refer to them as "black boxes". Today there is no alignment to physical disk sectors, no matter what we believe. Disks do not map sectors to physical regions in a way that we can understand from outside the box; the simplistic "geometry" reported by the device is an artifice. Technology transcended that limitation around 1984. The Cylinder/Head/Sector (C/H/S) notation reported by storage subsystems has been incorrect for years. The true number of sectors per track varies across the disk and is not exposed to the operating system. Sectors and tracks are still valid structures to think about; it is the sizes and arrangement of tracks that is no longer predictable. Disks expose the number of platters/tracks/sectors for the purpose of computing a total disk size, but that is the only useful purpose. *Yet prior to Windows Server 2008, Windows still utilized C/H/S notation in a futile attempt to align partitions.*

As a result of these limitations and assumptions, blocks of storage (for example, a file system cluster) as defined by the NTFS.SYS driver can be misaligned with blocks of storage found in the hardware (for example, a controller cache line or the stripe unit size of a disk array). In modern disk I/O subsystems, the goal is to map disk structures to cache segment lines and stripe units in the RAID array and array controller.

The following figures are highly stylized schematics of disk architecture showing default misaligned disk configurations as well as aligned configurations.

In both figures, objects are numbered for ease of reference. The vertical black dotted lines overlaying a red gradient correspond to the boundary between separate physical disks of a RAID group. Gradients correspond to areas with allocations that are not discretely numbered. As stated earlier, versions of Windows preceding Windows Server 2008 displayed the 63 reserved sectors reported by the disk hardware, immediately after which the remainder of the partition was exposed to the user. In these figures, the common sector size of 512 bytes is used. The stripe unit size in both examples is 64 KB. In both figures, the first row can be thought to correspond to stripe units, immediately below which is a row corresponding to disk sectors.

In Figure 3 the third row represents the default layout—the default 4 KB NTFS cluster size, appropriate for, say, a file server —in which the absence of disk alignment forces the eighth 4-KB cluster of user data to be laid across two stripe units—starting with the last empty 512-byte sector of the first stripe unit and continuing on to the second stripe unit. In this default configuration, every eighth cluster is written across two stripe units. This situation is perpetuated throughout the rest of the partition; because every nth cluster crosses stripe unit boundaries, two I/Os are required to perform a read or write.

The fourth row in Figure 3 represents the situation in which partition alignment has been done. A single sector is skipped and the first cluster of user data is written aligned with the starting boundary of the second stripe unit.

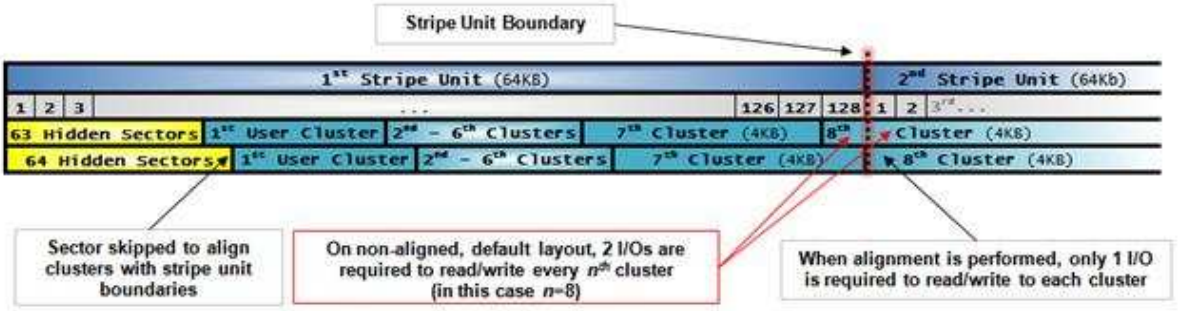

Figure 3. Stylized schematic of disk architecture corresponding to the NTFS default 4 KB cluster size showing default misaligned disk configuration as well as an aligned configuration

The file allocation unit size (cluster size) recommended for SQL Server is 64 KB; this is reflected in Figure 4.

Again, the first row corresponds to stripe units; the second row corresponds to disk sectors.

The third row in Figure 4 corresponds to the preferred 64 KB cluster size for SQL Server combined with the default partition misalignment that forces the first write of user data to be laid across two stripe units, starting with the first available sectors of the first stripe unit and continuing on to the second stripe unit. This situation is perpetuated throughout the rest of the partition.

In this configuration, every subsequent cluster is written across two tracks or two stripe units—each and every read and write is affected.

The fourth row of Figure 4 represents a situation in which partition alignment has been implemented. One full

stripe unit is skipped, and the first cluster of user data is written aligned with the boundaries between the first and second stripe units. The pathological situation characterized in the previous paragraph is remedied.

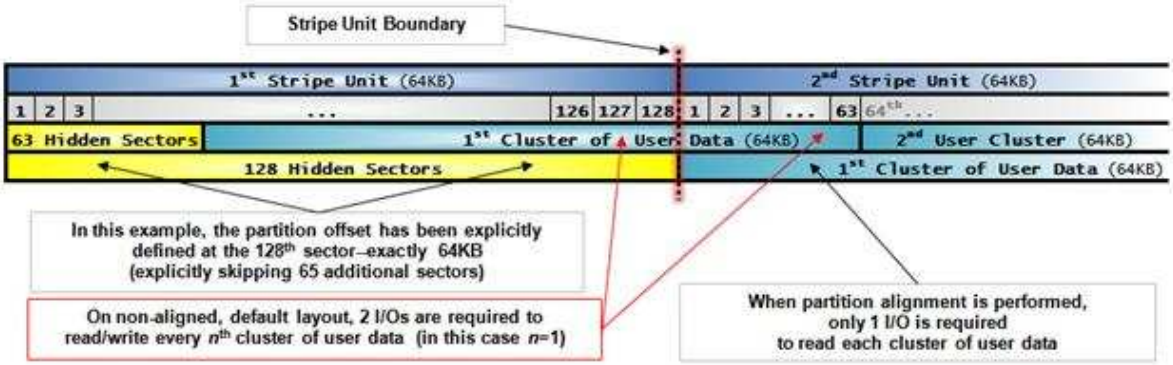

Figure 4. Stylized schematic of disk architecture showing default misaligned disk configuration as well as an aligned configuration suitable for SQL Server