Violin MEMORY

Best Practices

Best Practices Guide for Misaligned I/O

Document Number: 535-0267-00 Rev 02
May 2016

LEGAL NOTICE

Violin Memory, Inc.
4555 Great America Parkway
Santa Clara, CA 95054
USA

Table of Contents

1. What is Misaligned I/O

When I/O begins at a logical block that is not at the start of a physical block, the I/O is said to be misaligned. I/O misalignment is very common in virtualized environments because while the hypervisor (like VMware) may natively be aligned, the Virtual Machine guests may not be. However, certain versions of Linux, UNIX and Windows 2008 Server and earlier will more than likely have misaligned I/O issues if this issue wasn't addressed when the OS was installed.

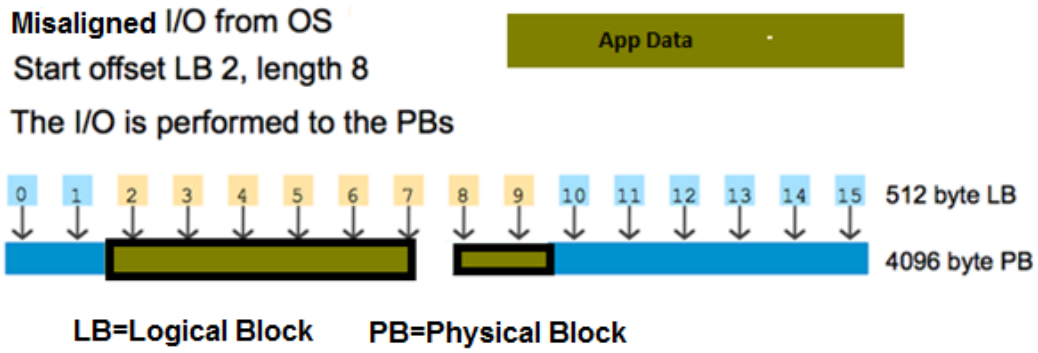

Figure 1: Misaligned I/O

The aligned I/Os are starting at the first offset zero of the physical block and the following file system I/Os are aligned with the physical block.

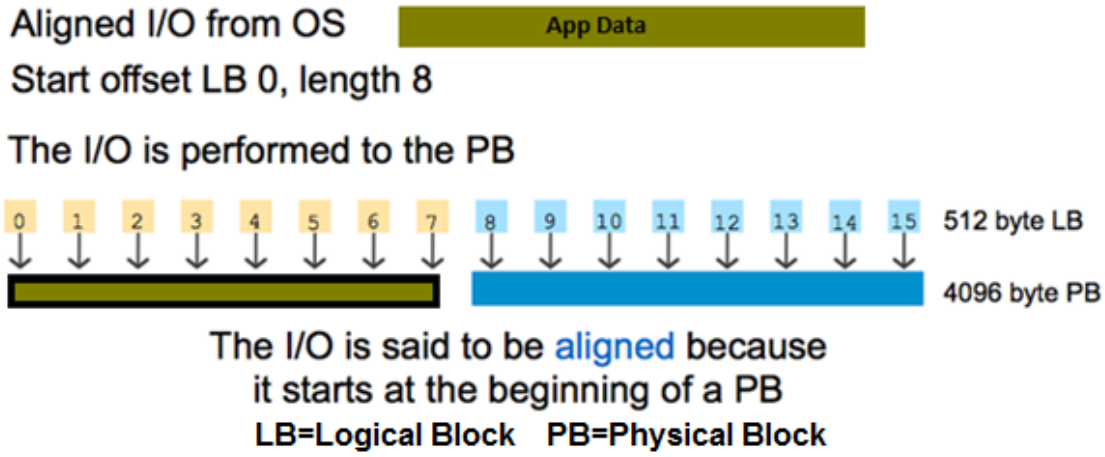

Figure 2: Aligned I/O

I/O Types	Aligned on 4k Boundary	4k Multiple Size
Aligned Wr I/Os	Yes	Yes
Partial Wr I/Os	No	No (<4k)
Misaligned Wr I/Os	No	Yes/No both (>4k)
Odd-size Wr I/Os	Yes	No

2. How to Detect the Concerto Misaligned I/O

Concerto OS detects the misaligned I/Os using the "misaligned.sh" script.

The misaligned.sh script captures the live misaligned I/O data on the dedup LUNs.

```
# /usr/local/concerto/util/misaligned.sh -l
vid     TWrites (I/OPs)     UnAlWrites  UnAlWrite %       TReads(I/OPs) UnAlReads   UnAlRead %
26                0             0          0%                0           0           0%
24           447837         28106          6%           186229       38651          20%

# /usr/local/concerto/util/misaligned.sh -l -d -v 24
vid     TWrites (I/OPs)    UnAlWrites  UnAlWrite %     TReads(I/OPs)   UnAlReads   UnAlRead %
24        469570           29480          6%                  168480       40570          24%

Vid    BlockSize    TotalWrites  MisalignedWR UnAl write % TotalReads Unalleged UnAl RD
24        I/O < 4K     141057     141057          100%      266098     266097    99%
          I/O = 4K    3142418          0            0%      851543          0     0%
     4K < I/O < 8K        332        332          100%           0          0     0%
     8K <= I/O < 16K   141005          0            0%      275285          0     0%
    16K <= I/O < 32K  1530779          0            0%      105074        292     0%
```

3. Linux Best Practice to Fix the 4K Aligned I/O

Users must always take care to use properly aligned and sized I/O. This is especially important for Direct I/O access. Direct I/O should be aligned on a "logical_block_size" boundary and in multiples of the logical_block_size. With native 4K devices (logical_block_size is 4K), it is now critical that applications perform Direct I/O that is a multiple of the device's logical_block_size. This means that applications that do not perform 4K aligned I/O, but 512-byte aligned I/O, will break with native 4K devices. Applications may consult a device's "I/O Limits" to ensure they are using properly aligned and sized I/O. The "I/O Limits" are exposed through both sysfs and block device Ioctl interfaces (also see: libblkid).

sysfs interface

/sys/block/<disk>/alignment_offset
/sys/block/<disk>/<partition>/alignment_offset
/sys/block/<disk>/queue/physical_block_size
/sys/block/<disk>/queue/logical_block_size
/sys/block/<disk>/queue/minimum_I/O_size
/sys/block/<disk>/queue/optimal_I/O_size

The kernel will still export these sysfs attributes for "legacy" devices that do not provide "I/O Limits" information/On. For example:

alignment_offset: 0
physical_block_size: 512
logical_block_size: 512
minimum_I/O_size: 512
optimal_I/O_size: 0

3.1. Linux File System Format on Concerto LUNs

To specify 4K block size partitions when creating a file system, use mke2fs / mkfs.xfs with the -b option.

Example:

```
>mke2fs -b 4096 /dev/dm-x
>mkfs.xfs –b size=4096 /dev/dm-X
```

Verifying the partition alignment:

1. The start sector of each partition should be evenly divisible by 8.
2. The ending sector of each partition + 1 should be evenly divisible by 8.
3. Run perf_test against a partition to ensure your partition starts on a 4K boundary.
4. Run the “misaligned.sh” script on the Concerto host to verify that it is truly 4K aligned.

3.2. Solaris

Overview

In SPARC Solaris systems there are two types of labels available: SMI (Sun Microsystems) and EFI (Extensible Firmware Interface). The Format tool selected with defaults ensures that any partition set up using the cylinder values will be correctly aligned.

The SPARC Solaris format tool using the SMI default, in common with other format and partition tools, uses annotation of virtual sectors, tracks and cylinders to describe a disk’s geometry. This same notation is used for Solid State FLASH devices in common with conventional disk drives.

If EFI labels are used, the partition setup uses sector values. This makes setting and verifying the beginning of a partition easy; the sector size multiplied by the beginning sector needs to be a multiple of 4096 bytes. The EFI DEFAULT is NOT 4K aligned.

The size of the EFI label is usually 34 sectors (sector 0 to sector 33), so partitions usually start at sector 34. This feature means that no partition can start at sector zero (0).

When the EFI label is created, the first partition by default starts at sector 34, hence it is NOT 4K aligned. You can use the format command to repartition the LUN so that there is a single partition starting at sector 39 (which is the 40th sector, counting from sector 0) to the end of the LUN. This will ensure 4K alignment (as 40 is divisible by 8, and 8 sectors is 4k).

Problem

Solid State Disk Drives (SSD) use NAND FLASH memory for storage. The storage array is aligned on block boundaries that are different from conventional disks that use 512Byte sectors. Contemporary SSDs commonly use 4KByte alignment. Depending on the drive firmware and cache storage on the drive, performance may be adversely affected due to excessive read/modify/write operations when transfers are not aligned.

Partitioning tools in use still use the concept of cylinders, tracks and sectors. This carries over to SSDs as well except that the cylinder, tracks and sectors are now virtual. Some tools maximize the number sectors and tracks; for example in Linux, a virtual cylinder is described as 63 Sectors and 255 tracks. A 24GB SSD will be presented as 2987 cylinders, 255 tracks and 63 sectors and a 32GB SSD as 3890 cylinders, 255 tracks and 63 sectors. Setting up a partition based on an arbitrary number of cylinders has a high probability of not being 4K aligned and a subsequent drop in performance.

Bytes per cylinder = 512 bytes/sector * 63 sectors * 255 tracks = 8225280

This is 1962.13740458 4K block. This is clearly NOT 4K aligned.

Unlike Linux and other systems using fdisk for partitioning, Solaris uses Format, which assigns different values to sectors per track and tracks per cylinder. Format assigns 128 sectors and 16 heads (Bytes per cylinder =

512 bytes/sector * 128 sectors * 16 tracks = 1048576) which is 256 4K blocks. This means that any partition created on any arbitrary cylinder boundary will be 4K block aligned.

Using Format

The SPARC Solaris tool for creating disk labels and partitioning disk drives is format. If a drive is not labeled, the format command prompts the user with a "Label it now?" message. The format utility includes verify command, which displays the assignment of sectors and heads. This allows the bytes per cylinder to be checked and also shows the start in cylinders permitting verification of alignment.

SMI vs EFI

The default label created by the format tool is an SMI (Sun Microsystems) label. This uses a (virtual) cylinder, sector, track notation. An alternative label is the EFI (Extensible Firmware Interface). This label uses simpler sector value to define the beginning of a partition. The tool shows the value of the sector size. To set or verify that a partition is on a 4K boundary, the sector size value is multiplied by the start of partition value; the result must be an integer multiple of 4096 to be 4K aligned.

Please note there are restrictions on EFI labels; search oracle.com for more information on EFI.

3.3. Windows

Windows Server 2012 and 2008

These operating systems automatically align the I/Os with 4k block size.

3.4. VMware ESXi

To ensure that the virtual machines are performing efficiently, make sure that you partition your virtual machine disks (VMDKs) inside the VMs with the correct alignment. The VMFS partition itself is aligned correctly, but your VMDKs inside this might not be. You must align your VMDKs properly, otherwise your VMs will give misaligned I/O, which has a negative impact on the VM's performance.

Note: This also applies if you are directly presenting Violin LUNs as RDM devices to ESX VMs. Follow the same steps in this document to align the virtual machines disks – in other words, treat the Virtual Machines just like you would treat a physical host with respect to alignment.

What is the Issue with the Guest OS Disk Misalignment?

This issue is applicable to the physical and virtual disk and not because of the VMFS layer or any other means. It is the limitation of the OS and how it partitions in the given HDD. Just like to the physical world, in the virtual world, the Hypervisor also gives an HDD but it is a virtual HDD. But the OS doesn't know if it is a virtual disk or physical disk, so eventually the OS won't complete the partition with correct alignment.

The VMFS is aligned with the 1MB boundary (starts from the LBA 2048). The VMFS 5 block size is 1MB. The VMFS and the underlying storage is already aligned. When a misaligned Guest OS sends a READ/WRITE request to the hypervisor layer, that is from the VMDK to the VM SCSI controller to the VMFS and finally to Concerto/ACM/VCM. The storage looks for more than one physical sector or chunk. This is an overhead for the storage and will be an overhead to the VMkernel, because the VMkernel has to wait until the array performs the task.

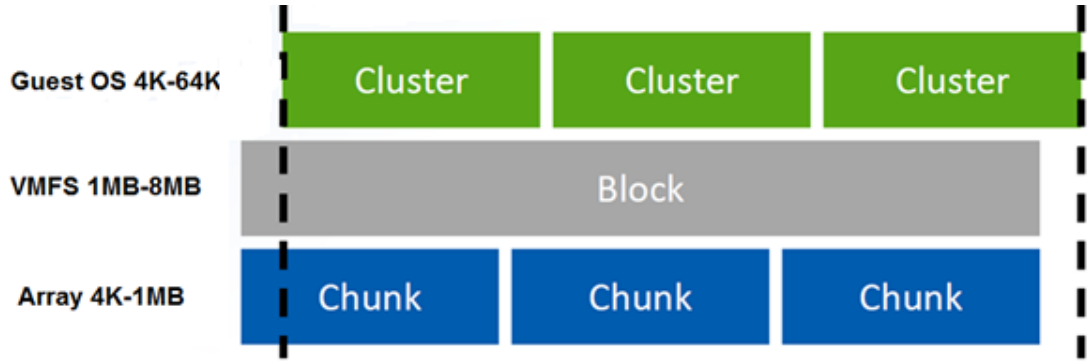

Figure 3: ESXi Guest OS Misaligned I/O

If the Guest OS is aligned for one 4KB write/read, it will use only one single Chunk from the storage. This will give good response time and low latency for the I/OPs operation.

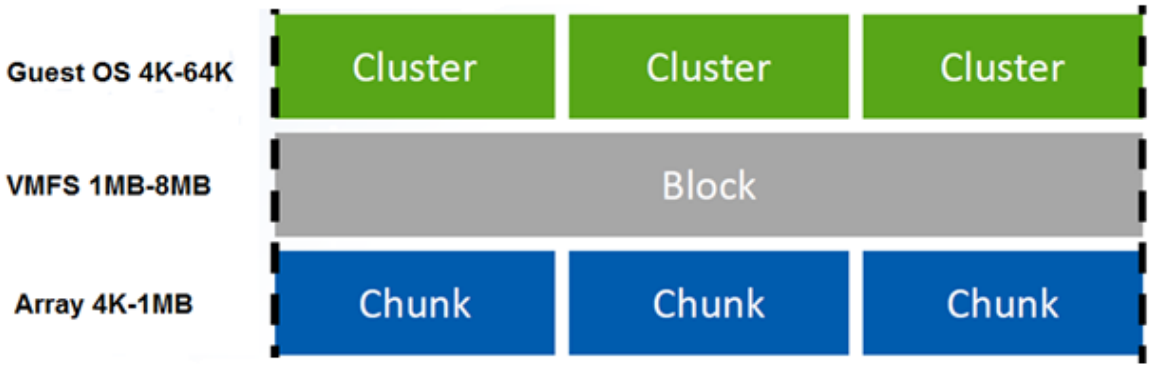

Figure 4: ESXi Guest OS Aligned I/O