

I/O Size and Alignment Reporting for Storage Devices

Martin K. Petersen, Oracle Linux Engineering <martin.petersen@oracle.com>

Introduction

The industry migration towards harddrives with 4KB physical blocks has caused us to make some changes to the Linux block I/O stack. These changes leverage existing parameters in SCSI Block Commands and the ATA Command Set to provide reporting of alignment and preferred I/O request sizes. The values provided are used by partition tools, LVM and MD as well as filesystems to ensure that data is layed out in an optimal fashion.

This document provides a set of recommendations for storage device vendors about which values to report to make Linux perform optimally.

1 Logical vs. Physical Block Size

Until recently what has been commonly referred to as *sector size* has been both the unit used by the programming interface to address a location on disk as well as the size used internally by the drive firmware to organize user data. In this document (and in the Linux kernel) we distinguish between:

- *physical block size* is the size used internally by the device firmware
- *logical block size* is the unit used to address a location on the storage

2 ATA Command Set

Linux will generally assume that an ATA device has both a logical and physical block size of 512 bytes. If that is not the case the following words must be exported by the device:

2.1 512-byte Logical Block Size, 4096-byte Physical Block Size

Word 106 bit 15 must be 0 and bit 14 must be 1. This signals that word 106 is valid. Bit 13 must be set to indicate that bits 3:0 are valid and that word 209 is specified.

Bits 3:0 indicate the physical block size exponent. I.e. 2^n logical blocks/physical block. For a drive with 4096-byte physical blocks and 512-byte logical blocks, a value of 3 is reported in bits 3:0 ($2^3 = 8$).

Command	Word	Bits	Hex
IDENTIFY DEVICE	106	0110 0000 0000 0011	0x6003
IDENTIFY DEVICE	209	0100 0000 0000 0000	0x4000

Table 1 512-byte logical, 4096-byte physical, 0-aligned

If the first logical block does not begin on a boundary aligned to the physical block size, word 209 should be reported by the device. Bit 15 must be zero, bit 14 must be 1. Bits 13:0 indicate the logical sector offset within the first physical sector where the first logical sector is placed.

For compatibility with legacy operating systems some vendors may align the blocks so that LBA 63 falls on a physical block boundary. In that case an alignment of 1 should be reported. Please note that ATA alignment reporting is different from T10's READ CAPACITY(16). ATA indicates (negative) offset from the beginning of the physical block. T10 SBC reports the lowest aligned LBA.

Command	Word	Bits	Hex
IDENTIFY DEVICE	106	0110 0000 0000 0011	0x6003
IDENTIFY DEVICE	209	0100 0000 0000 0001	0x4001

Table 2 512-byte logical, 4096-byte physical, 1-aligned

2.2 4096-byte Logical Block Size, 4096-byte Physical Block Size

If a drive uses 4KB logical and physical blocks, IDENTIFY DEVICE words 106, 117, and 118 must be specified.

Word 106 bit 15 must be 0 and bit 14 must be 1. This signals that word 106 is valid. Bit 12 must be set to indicate that words 117 and 118 are specified.

Word 117 (LSB) and 118 (MSB) specify the logical block size in words. I.e. for a 4KB drive that would be 2048.

Command	Word	Bits	Hex
IDENTIFY DEVICE	106	0101 0000 0000 0000	0x5000
IDENTIFY DEVICE	117	0000 1000 0000 0000	0x0800
IDENTIFY DEVICE	118	0000 0000 0000 0000	0x0000

Table 3 4096-byte logical, 4096-byte physical

2.3 Medium Rotation Rate

If the storage device is non-rotational (i.e. SSD or array) word 217 of IDENTIFY DEVICE should be set to one. The I/O scheduler may then be less aggressive in terms of seek avoidance. Any other value reported in 217 will be ignored.

Command	Word	Bits	Hex
IDENTIFY DEVICE	217	0000 0000 0000 0001	0x0001

Table 4 Medium Rotation Rate

3 SCSI Block Commands

Linux will generally assume that a SCSI block device has matching logical and physical block sizes. If that is not the case, the device must report the following in READ CAPACITY(16):

Bytes 8 (MSB) - 11 (LSB) indicate the device's logical block size.

Byte 13 indicates the physical block size exponent. I.e. 2^n logical blocks/physical block. For a drive with 4096-byte physical blocks and 512-byte logical blocks, a value of 3 is reported in bits 3:0 ($2^3 = 8$).

Byte 14, bits 5:0 (MSB) and byte 15 specify the lowest aligned LBA. Please note that this reporting is different from the method used in ATA.

Also note that Linux will only issue READ CAPACITY(16) to devices that report a PRODUCT REVISION LEVEL of 0x5 in standard INQUIRY response. Consequently, any device whose physical block size is different from its logical ditto must report compliance to at least SBC version 2.

3.1 Identical Logical and Physical Block Sizes

Byte	Value (Hex)	Byte	Value (Hex)
8	0x00	8	0x00
9	0x00	9	0x00
10	0x02	10	0x10
11	0x00	11	0x00
13	0x00	13	0x00
14	0x00	14	0x00
15	0x00	15	0x00

512-byte blocks 4096-byte blocks

Table 5 Identical logical and physical block sizes

3.2 512-byte Logical Block Size, 4096-byte Physical Block Size

When the storage device has a discrepancy between logical and physical block size, the logical blocks per physical block exponent must be specified in READ CAPACITY(16), byte 13.

For compatibility with legacy operating systems some vendors may align the blocks so that LBA 63 falls on a physical block boundary. In that case an alignment of 7 should be reported in byte 15 since 7 is the lowest physically aligned LBA on the device.

Byte	Value (Hex)	Byte	Value (Hex)
8	0x00	8	0x00
9	0x00	9	0x00
10	0x02	10	0x02
11	0x00	11	0x00
13	0x03	13	0x03
14	0x00	14	0x00
15	0x00	15	0x07

0-aligned 1-aligned

Table 6 512-byte logical, 4096-byte physical

3.3 Medium Rotation Rate

If the storage device is non-rotational (i.e. SSD or array) the MEDIUM ROTATION RATE of the Block Device Characteristics (0xB1) VPD page should be set to one. The I/O scheduler may then be less aggressive in terms of seek avoidance. Any other value reported will be ignored.

Byte	Value (Hex)
4 (MSB)	0x00
5 (LSB)	0x01

Table 7 Block Device Characteristics

3.4 Block Limits

SCSI Block Commands version 2 introduced the Block Limits (0xB0) VPD as a generic interface for array controllers to report preferred I/O sizes.

Bytes 6 (MSB) and 7 (LSB) indicate the OPTIMAL TRANSFER LENGTH GRANULARITY. Transfers with lengths not equal to a multiple of this many logical blocks may incur significant delays in processing.

Bytes 12 (MSB) - 15 (LSB) indicates the OPTIMAL TRANSFER LENGTH in blocks. Transfers with lengths exceeding this value may incur significant delays in processing.

Byte	Value (Hex)	
6 (MSB)	0x00	
7 (LSB)	0x80	128 512-byte blocks = 64 KB
12 (MSB)	0x00	
13	0x00	
14	0x02	512 512-byte blocks = 256 KB
15 (LSB)	0x00	

Table 8 Example Block Limits response page: 512-byte logical blocks, 64 KB chunk size, 4-way striping

The Linux Kernel vs. I/O Topology

Internally, the Linux kernel operates with several parameters for block devices:

- `logical_block_size` - used to address a location on the device
- `physical_block_size` - smallest unit the device can operate on
- `minimum_io_size` - device's preferred minimum unit for random I/O
- `optimal_io_size` - device's preferred unit for streaming I/O
- `alignment_offset` - the number of bytes the beginning of the Linux block device (partition/MD/LVM device) is offset from the underlying physical alignment

The logical and physical block sizes are set based on the values reported by the physical storage. Generally, `logical_block_size` \leq `physical_block_size` \leq `minimum_io_size`. If no value is specified for the physical block size, the logical one applies. And if the storage device does not set `OPTIMAL_TRANSFER_LENGTH_GRANULARITY` in the Block Limits VPD, then the physical block size applies.

Linux will attempt to align all data structures on a `minimum_io_size` boundary, compensating for any device alignment offset reported. Linux will also attempt to issue I/O in multiples of `minimum_io_size`. The `optimal_io_size` value may be used by some filesystems to optimize for sustained write throughput.

When stacking devices, Linux will scale `minimum_block_size` to a value suitable for all underlying devices. Linux will also attempt to align partitions, MD and LVM devices so they start on a physically aligned boundary. The alignment handling code is capable of handling mismatched devices. For instance a 512-byte logical/physical drive combined in a mirror with a 512-byte logical/4096-byte physical drive with 1-alignment will be adjusted to start on LBA 7.

It has been common practice for some RAID devices to compensate for the DOS LBA 63 misalignment internally. Typically this is done implicitly when selecting a Windows or a Linux personality on the LUN in question. Going forward, it is imperative that any alignment compensation is reported in READ CAPACITY(16).

4 Examples

The following table indicates the parameters resulting from the devices in the left column being registered with the Linux kernel. Note that the `min_io` value is the one that filesystems will strive to use. And that Linux will attempt to align on, taking `align_off` into account.

The RAID array examples all provide a Block Limits VPD to augment the information provided in READ CAPACITY(16).

Device	logical	physical	min_io	opt_io	align_off
Disk 512/512	512	512	512	0	0
Disk 512/4KB	512	4096	4096	0	0
Disk 512/4KB, 1-align.	512	4096	4096	0	3585
Disk 4KB/4KB	4096	4096	4096	0	0
RAID0, 64 KB x 4 dr.	512	512	65536	262144	0
RAID1, 16 KB	512	512	16384	0	0
RAID5, 8 KB x 3 dr., 1-align.	512	512	8192	16384	3584

Table 9 Common device types and their resulting parameters

Revision History

Date	Change
2009-07-23	(mkp) Initial Version
2009-08-17	(mkp) A few clarifications