From 512 to 4K
A case study in supporting large sector size SSDs in Solaris

Bo Zhou
Sun Microsystems
Agenda

- Introduction
- Background
- Design & Implementation
- Future Work
- Performance Comparison
- Conclusion
Two trends of storage devices

- Craving for Speed
  - Hard disk drives bring down the system performance because of the rotational traits
  - The advent of Solid-State Drive (SSD) represents a sea change in storage system
    - Exceptional bandwidth
    - Excellent random I/O performance
    - Save power and improve reliability
  - The optimal sector size of SSD is typically 4KB
Introduction – Need for Capacity

- Craving for Capacity
  - more platters, heads and parts introduce more heat, more vibration and more opportunity for errors
  - Large sector size brings benefits
    - High areal of density while maintaining data integrity
    - Reliability
  - Sample 4K HDD drives are available now
Solaris can not use SSDs with sector sizes other than 512 bytes.

Many modules and drivers hardcode disk sector/block size as 512 bytes.

Many applications are sector/block size sensitive

Read-Modify-Write (RMW) can apply to disk driver, but sacrifices performance

The SSD/Flash drives are already in use now

The performance of SSD can be increased if I/O is sector size aligned
Background – Key Concept

- **Physical sector/block**
  - The physical unit of storage on the surface of the disk
  - The smallest unit of data which can be physically written to or read from the disk

- **Logical sector/block**
  - The disk drive presents itself to outside world as a linear address space of logical blocks
  - The size may be in principle different from that of the physical blocks
Emulation Mode

- The physical sector size is 512 bytes, the disk firmware presents the size as 4K bytes for the upper layers.
- The physical sector size is 4K bytes, the disk firmware presents the size as 512 bytes for the upper layers, such as SSD.
- Disk drivers can also do the emulation
  - Export large sector size to the upper modules and apps
  - Handle the block size and address translation inside driver
Design & Implementation - Overview

* Today

File Systems/Applications
- buf(512B)
- sd(7D)
- LSS = 512B

Disk/Array Controller firmware
- Media
  - PSS = 512B

* Tomorrow

Emulation mode/SSD
- File Systems/Applications
  - buf(512B)
  - sd(7D)
  - LSS = 512B

Native 4KB sector
- File Systems/Applications
  - buf(512B)
  - sd(7D)
  - LSS = 4KB

Migration disks
- File Systems/Applications
  - buf(512B)
  - sd(7D)
  - LSS = 4KB

- File Systems/Applications
  - buf(512B)
  - sd(7D)
  - Media
    - PSS = 4KB

LSS: Logical Sector Size
PSS: Physical Sector Size

Storage Developer Conference 2009
© 2009 Insert Copyright information here. All rights reserved.
Design & Implementation - Overview

System Call/VFS

ZFS

Block Interface

sd

SCSA/MPxIO

FC  iSCSI  mpt

SSD/HDD

Solaris Host Driver Stack

User
Kernel

Filesystem layer

buf (9S)

Target Driver layer

scsi_pkt (9S)

HBA Driver layer

Storage Developer Conference 2009
© 2009 Insert Copyright information here. All rights reserved.
Buf(9S) is kept intact
- Backward compatible with most existing modules
- Logical block size is fixed at 512B/block
- SCSI disk driver (sd) is responsible for translating between physical sector size and logical block size

Query disk sector size
- Applications will not be automatically supported on large sector size disks: for physical sector size/address sensitive applications, use ioctl to query the size
VTOC label

Solaris VTOC 16 - x86 platform
- MBR is at the first 512 bytes of sector 0, sector size can be 512B or larger ones
- VTOC label is at the first 512 bytes of sector 1 of Solaris partition

Solaris VTOC 8 – SPARC platform
- VTOC label is at the first 512 bytes of sector 0
Design & Implementation - Label

Sector 0

- MBR
- VTOC label

Sector 1

- VTOC label

VTOC 8

- VTOC
- MBR
- NULL
- EFI (Extensible Firmware Interface) label
  - PMBR is at the first 512 bytes of sector 0, the remaining bytes are set to 0
  - GPT header is at the first 512 bytes of sector, the remaining bytes are set to 0
  - Partition entries are at the sectors after GPT header
Design & Implementation - Label

512-byte sector size drive

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PMBR</td>
</tr>
<tr>
<td>512</td>
<td>GPT Header</td>
</tr>
<tr>
<td>1024</td>
<td>Partition Entry 0 - 3</td>
</tr>
<tr>
<td>1536</td>
<td>Partition Entry 4 - 7</td>
</tr>
<tr>
<td>2048</td>
<td>Partition Entry 8 - 11</td>
</tr>
<tr>
<td>2560</td>
<td>Partition Entry 12 - 15</td>
</tr>
<tr>
<td>3072</td>
<td></td>
</tr>
<tr>
<td>16896</td>
<td>Partition Entry 124 - 127</td>
</tr>
</tbody>
</table>

4KB sector size drive

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PMBR</td>
</tr>
<tr>
<td>4096</td>
<td>GPT Header</td>
</tr>
<tr>
<td>8192</td>
<td>Partition Entry 0 - 31</td>
</tr>
<tr>
<td>12288</td>
<td>Partition Entry 32 - 63</td>
</tr>
<tr>
<td>16384</td>
<td>Partition Entry 64 - 95</td>
</tr>
<tr>
<td>20480</td>
<td>Partition Entry 96 - 127</td>
</tr>
<tr>
<td>24576</td>
<td></td>
</tr>
</tbody>
</table>
Disk Utilities

- Sector/Block size is hardcoded as 512 bytes
- The EFI label layout changes
- Fdisk layout changes
- Partition tools
- All applications which are sector/block size sensitive needs be to changed by the application developers
SCSI Disk Driver (sd)

- Sd queries the drive’s physical sector size during attach process
- For non-USCSI I/O request, sd checks whether the size is aligned with physical sector size, if it is, sd sends SCSI READ/WRITE commands to the disk drive
- If the I/O is misaligned:
  - RMW is enabled for hard disk drive, transfer is allowed, performance penalty
  - RMW is disabled for hard disk drive, transfer is forbidden, error is returned
RMW applies to large sector size SSDs and HDDs

- For transition from 512 to 4K
- Sacrifices the performance to gain backward compatibility
- Enable/Disable RMW is configurable by end users
- Misalign message prompts to warn user that the performance penalty
ZFS supports large sector size SSDs without any changes
Fit various of sector sizes dynamically
The I/O sizes are typically larger than the sector/block size
ZFS enables hybrid storage pools using high performance SSDs and low cost HDDs
Virtual Machine

- Xen
  - Can use large sector size SSDs as data disk
  - Host OS provides the drive’s sector size to guest OS

LDOM

- Large sector size SSDs could be exported to guest domain as data disk
- Host domain provides sector size information to guest domain
Future Work - BIOS

- **BIOS Firmware**
  - BIOS is responsible for detecting, identifying, and configuring the hardware devices and peripherals.
  - Two methods to deal with large sector size disk drives:
    - Maintain 512 bytes logical sector size backward compatibility by emulating the logical to physical translation within the drive.
    - Utilize a logical sector size which is greater than 512 bytes, the same as the physical sector size, this requires the change for BIOS.
Future Work - Installation

- OS installation
  - The installer must determine what kind of disk drives it is installing on
  - Current installer hardcode the sector size to 512 bytes
  - Installer needs to adjust any internal buffer sizes to accommodate the larger LBA transfers
Future Work - Applications

- Applications will not automatically support large sector size SSDs
- It's applications' responsibility to detect, identify, configure and access the installed drive (SSDs or HDDs), using the native sector size
- ISVs should update their applications
- Sector size aligned I/Os from applications accelerate the performance.
Impact on performance

Partition alignment issue

- The starting address of partition is not aligned with 4KB
- The misaligned I/Os introduces serious performance downgrade
- Can prove 4K aligned I/Os are better than the misaligned ones

Other impacts from both software and hardware
Performance Comparison

- Comparison results from SSD/Flash drive performance team

<table>
<thead>
<tr>
<th></th>
<th>READ IOPS</th>
<th>WRITE IOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>4K misaligned</td>
<td><img src="Image" alt="Graph" /></td>
<td><img src="Image" alt="Graph" /></td>
</tr>
<tr>
<td>4K aligned</td>
<td><img src="Image" alt="Graph" /></td>
<td><img src="Image" alt="Graph" /></td>
</tr>
</tbody>
</table>
Comparison of experimental results with our testbed

- **READ IOPS**
  - 4K misaligned
  - 4K aligned

- **WRITE IOPS**
  - 4K misaligned
  - 4K aligned
Conclusion

- Large sector size SSD and HDD represents the future storage device
- The chasing for speed and capacity never ends
- The transition from 512B to 4KB sector size disk drive should last for a relative long time
- The entire industry food chain needs to prepare for the transition
Questions?