Solaris Volume Manager Performance Best Practices

Glenn P. Fawcett, Strategic Application Engineering

Sun BluePrints™ OnLine—November 2003

http://www.sun.com/blueprints

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95045 U.S.A.
650 960-1300

Part No. 817-4368-10
Revision 1.0, 10/30/03
Edition: November 2003

Please
Recycle

Adobe PostScript

Solaris Volume Manager Performance Best Practices

Compelling new features such as soft partitioning and automatic device relocation make the Solaris™ Volume Manager software a viable candidate for your storage management needs. Solaris Volume Manager features enhance storage management capabilities beyond what is handled by intelligent storage arrays with hardware RAID.

Beginning with the Solaris™ 9 Operating Environment (OE), Solaris Volume Manager software is integrated with the Solaris OE and does not require additional license fees. Simply install the Solaris 9 OE and begin using Solaris Volume Manager software. The new Solaris Volume Manager features, coupled with the improved distribution model, make Solaris Volume Manager software a natural choice.

The Strategic Applications Engineering (SAE) group within Sun Microsystems performs industry standard benchmarks. Our Benchmarking activity enables us to characterize Solaris Volume Manager performance, and to develop best practice recommendations to best serve our customers. This article provides some of those recommendations for Solaris Volume Manager performance best practices.

This article is intended for system, storage, and database administrators.

The topics in this article include:

Solaris Volume Manager Performance Overview

This section explains how Solaris Volume Manager software performs with respect to its competition. To compare performance with Veritas VxVM 3.5, a benchmark was run that submits random 8-Kbyte reads through multiple threads to simulate an online transaction processing (OLTP) environment.

Solaris Volume Manager software and VxVM performance were tested using volumes created on nine Sun StorEdge™ T3 arrays using RAID 0 LUNs. Two concatenated volumes per LUN were created for a total of 18 volumes. Starting with 8 threads directed at each volume, additional threads were added until there were 16 random threads per volume. Performance results are shown in FIGURE 1.

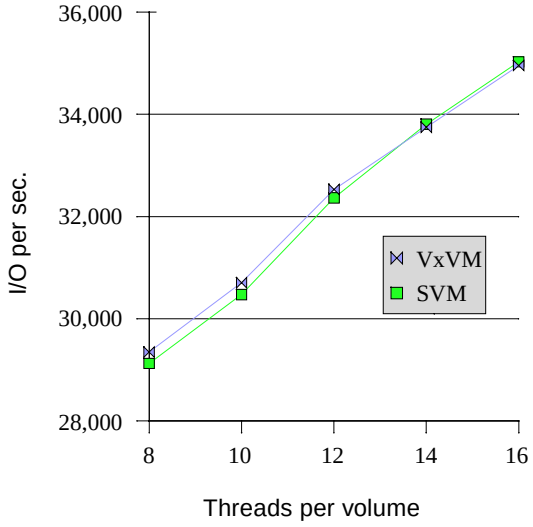

FIGURE 1 Solaris Volume Manager Software and VxVM Performance With 8-Kbyte I/O

Solaris Volume Manager software and Veritas performance are nearly the same. System CPU utilization at 35,000 I/O per second is only 14% for Solaris Volume Manager software and 15% for Veritas. This leaves plenty of CPU resources to achieve good application performance.

Solaris Volume Manager Striping Considerations

Software striping is becoming far less necessary as more intelligent hardware storage solutions emerge. Before considering Solaris Volume Manager striping, investigate the capabilities of your current I/O subsystem. If software striping is still deemed necessary, then take care when configuring the volumes.

The main problem with Solaris Volume Manager striping is split I/O. Splitting of an individual I/O operation to more than one disk degrades performance. There are several ways to avoid this:

- Use hardware striping where possible. This prevents Solaris Volume Manager software or the operating system from having to split an I/O operation.
- Increase the stripe width to lessen the frequency of splitting an I/O operation.
- Calculate alignment based on the I/O size and offsets. This works well for databases with a known I/O size.

Increase the Default Solaris Volume Manager Stripe Width

The probability of splitting an I/O operation is inversely proportional to the stripe width. Consider an OLTP system which mostly performs 8-Kbyte I/O. A 32-Kbyte stripe width has a probability of splitting 1 of 4 I/O operations, whereas a 1-Mbyte stripe width splits only 1 of 128 I/O operations (less than one percent). Increasing the stripe width is the single most important improvement you can make to decrease the probability of splitting an I/O operation.

Make the Stripe Width Large Relative to the I/O Size

If you use the `metainit` command without the `-i` option, Solaris Volume Manager software uses a default of 16 Kbytes for the stripe width. A narrow stripe width doesn't allow the application to take advantage of read-ahead for sequential I/O provided by the underlying storage subsystem. A stripe width of 1 Mbyte or greater is common, especially when implementing a stripe and mirror everywhere (SAME) strategy.

Align Soft Partitions on the Stripe Boundary

Before the Solaris 8 OE Solaris Volume Manager software, each volume had to match a hard disk partition or volume table of contents (VTOC). With this older scheme, there is a direct relationship between a Solaris Volume Manager device and the physical disk or LUN. This scheme limits the number of volumes to the number partitions or VTOCs that can be created.

Soft partitioning enables one piece of disk to be partitioned into more slices than is possible by a device VTOC. Soft partitioning provides a great amount of flexibility because it enables volumes to be created on top of individual disks or existing Solaris Volume Manager volumes. Layering of volumes is particularly useful with large RAID devices, which can easily exceed 1 terabyte. But soft partitioning on top of striped hard partitions can lead to the split I/O problem.

As mentioned, if the stripe width is large compared to the I/O size, there is less splitting of I/O. In the case where 8 Kbytes is the I/O size and 1 Mbyte is the stripe width, less than one percent of the I/O operations are split. But still, there are some split I/O operations. To get total alignment, striped soft partitions must be aligned on a stripe boundary.

Consider the example of an Oracle application that uses an 8-Kbyte block size. The following example shows the default soft partitions or layered volumes that are created with the `metainit` command.

```
# metainit d10 1 2 c3t1d0s7 c5t1d0s7 -i 256k
# metainit d11 -p d10 1024m
# metainit d12 -p d10 1024m
```

Meta device `d11` happens to begin on a stripe boundary as shown in FIGURE 2. Random block reads to this device are exactly 8 Kbytes in size and involve exactly one disk. Due to the 512-byte label added to a soft partition, meta device `d12` will split I/O that falls on the stripe boundary.

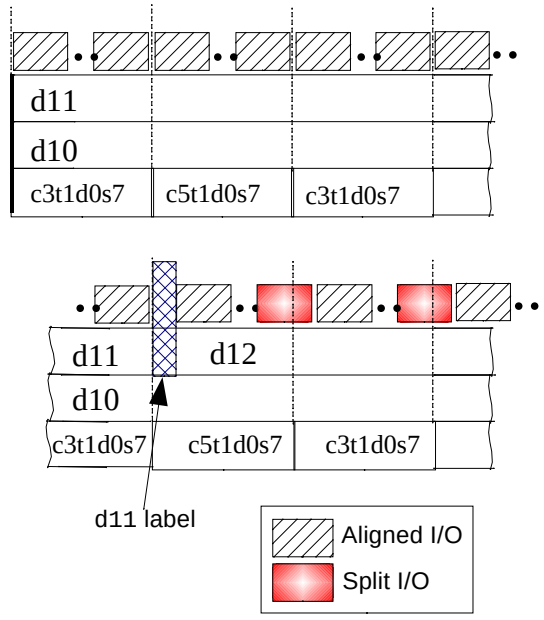

FIGURE 2 Split I/O Due to Watermark

To correct this, use the `-o` option to specify the offset to be a multiple of the stripe width. The following example shows how to use the `-o` option with the `metainit` command to specify the starting offset of the soft-partition to align on the stripe boundary.

```
# metainit d10 1 2 c3t1d0s7 c5t1d0s7 -i 256k
# metainit d11 -p d10 -o 512 -b 2097152
```

Taking this example one step further, use the following formula to determine the next offset:

nextoffset = *last_offset* + *last_size* + *stripe_width* (in 512-byte blocks)

For this example, the result is:

512 + 2097152 + 512 = 2098176 blocks

The corresponding `metainit` command:

```
# metainit d12 -p d10 -o 2098176 -b 2097152
```

This offset guarantees that the `d12` meta device starts on a stripe boundary and that the 8-Kbyte I/O will not be split unnecessarily.

To administer volumes using this scheme, you must keep track of offsets. While this is easy during the initial creation of volumes, it is useful to be able to observe the current settings. The following example shows how the `metastat -p` command can be used to show the volume offsets. This command is useful for documenting the storage configuration as well as for ongoing administration.

```
# metastat -p |egrep 'd2'
  d208 -p d2 -o 61473953 -b 9680
  d2 1 1 c5t1d0s0
  d207 -p d2 -o 61464272 -b 9680
  d206 -p d2 -o 60440255 -b 1024016
  d205 -p d2 -o 59416238 -b 1024016
  d204 -p d2 -o 52264621 -b 7151616
  d203 -p d2 -o 45113004 -b 7151616
  d202 -p d2 -o 37961387 -b 7151616
  d201 -p d2 -o 30809770 -b 7151616
  d29 -p d2 -o 25677481 -b 5132288
  d28 -p d2 -o 20545192 -b 5132288
  d27 -p d2 -o 15412903 -b 5132288
  d26 -p d2 -o 10280614 -b 5132288
  d25 -p d2 -o 5148325 -b 5132288
  d24 -p d2 -o 16036 -b 5132288
  d23 -p d2 -o 10691 -b 5344
  d22 -p d2 -o 5346 -b 5344
  d21 -p d2 -o 1 -b 5344
```

Make the Software Stripe Width a Multiple of the Hardware Segment Size

Hardware RAID and software RAID are sometimes combined to increase availability and throughput, but there can be performance consequences if the underlying storage layout is not considered.

Make sure that you understand the stripe unit or segment size of the underlying storage architecture when implementing software striping on top of hardware RAID. Failure to do this can cause a single I/O operation to be split between multiple devices on the underlying storage. This unwanted split I/O operation increases latency and degrades overall throughput.

The underlying segment size differs from array to array. For example:

- Sun StorEdge 9980 system uses a 48-Kbyte segment size.
- Sun StorEdge 6910 system and Sun StorEdge T3 arrays use 32-Kbyte or 64-Kbyte segment size (64-Kbyte provides the best performance).

When combining multiple LUNs, it is best to use a fairly large stripe width. For the StorEdge 9980 example, a stripe width of 20 * 48 Kbytes, or 960 Kbytes, is a good place to start for an expected 8-Kbyte I/O size. This enables the LUN to benefit from read-ahead while reducing the probability of splitting an I/O operation.

Limit Striping and Meta Devices

When coming from a Veritas background, there is a tendency to want to map Veritas concepts to Solaris Volume Manager software. VxVM creates *subdisks* for every portion of disk that is used in a stripe. This can be simulated with Solaris Volume Manager software by using soft-partitions. There are no performance implications, but this technique can use an excessively large number of meta devices, especially as the number of devices in a stripe is increased.

Consider an example where 200 volumes are created as stripes across 16 drives. If a soft-partition is created for each subdisk, a total of 16*200+200 = 3400 meta devices is needed. If the soft-partitions are created on top of a striped hard partition, only 1+200 = 201 meta devices are needed.

By default, Solaris Volume Manager software can create only 128 meta devices. You can increase this to a maximum of 8192 by modifying the `nmd` field in the `/kernel/drv/md.conf` file. To get this to take effect, the machine must be rebooted with `boot -r`. This procedure is discussed in more detail in the Solaris Volume Manager software administration guide.

Software RAID Considerations

Software mirroring and RAID 5 are used to increase the availability of a storage subsystem. This has become much less necessary with more intelligent storage solutions that implement hardware mirroring and RAID 5. While software RAID 5 is really not a good idea for performance, there is sometimes a need to use software mirroring.

Use Default Round-Robin Read Policy

Software mirroring uses additional bandwidth on writes but performs well on reads. By default, Solaris Volume Manager software implements a round-robin read policy, which balances I/O across both sides of the mirror. For a well-balanced I/O subsystem, round-robin works the best.

In addition to round-robin, Solaris Volume Manager software supports Geometric and First features. Geometric splits the logical addresses into ranges. Geometric is used to increase sequential read throughput in a well balanced environment. The First read policy only reads the first subdisk of the mirror. First is useful if the secondary device is slower than the primary.

Decrease Mirror Sync Time With a Large `metasync` I/O Size

Sync performance is dominated by the I/O size. By default, the `metasync` command uses a 32-Kbyte I/O size. This is not ideal. To increase this to 1 Mbyte, use `metasync -r 2048` (number specified in 512-Kbyte blocks). This alters the I/O size and reduces the overall sync time.

To ensure large I/O is used during a system reboot, modify the `/etc/rc2.d/S95svm.resync` script by adding the `-r 2048` option to the `metasync` command. Note that in order for this to work properly, `maxphys` must be at least 1 Mbyte (by default, `maxphys` is set to `256k` so it is worth checking).

Multipathing

Modern storage arrays typically have multiple paths to the same disk or LUN. This feature enables you to increase storage bandwidth and availability. Sun StorEdge Traffic Manager Software, formerly known as MPXIO, is built into the Solaris 9 OE and can be used with Solaris Volume Manager software.

SAE has used multi-pathing to increase the performance of DSS benchmarks. DSS environments have larger I/O sizes which stress controller throughput.

Most customers implement some sort of alternate path scheme for availability. If Sun StorEdge Traffic Manager software is used, multiple paths provide availability in the event of a controller failure and load balancing provides increased bandwidth.

Solaris Volume Manager Performance With UFS File Systems and Oracle

The debate over file systems versus raw volumes is less of a topic of discussion these days. A vast majority of data centers use file systems and have very good reasons for doing so. Sun understands this and has improved the performance of UFS file systems to nearly match that of raw file systems. With the introduction of the *Concurrent Direct I/O* feature in the Solaris 8 3/01 OE, the last of the UFS performance bottlenecks have disappeared.

To get Solaris Volume Manager software to perform well with UFS file systems, no special performance tuning is required. To enable good UFS performance, you need to address a few simple points:

- Write-On-Write is not a problem for Oracle database software. The `metainit` man page describes a problem that can cause both sides of a mirror to have different data. If the contents of buffers are changed while the data is in-flight to disk, then different data can end up on each side of the mirror. The `metainit` man page suggests the following `/etc/system` file setting:
  ```
  md_mirror:md_mirror_wow_flg=0x20
  ```
  This setting results in stable copies for raw and direct I/O. If this parameter is set, it significantly degrades write performance. Oracle software does not exhibit this problem because it does not allow changes to the buffer while a write is in-flight.
- Use an 8-Kbyte database block size in conjunction with an 8-Kbyte fragment size for the file system. This block size ensures alignment of database blocks with underlying storage. If blocks are not properly aligned or are too small, write performance can suffer. Consider the example of a database using 2-Kbyte blocks on a file system. The file system has a block size of 8 Kbytes, so 4 database blocks will fit in one FS block. When the database issues a write on a 2-Kbyte block, the file system must write all 8 Kbytes at once. This means that the remaining 75% of the block will have to be read before it can be written. If you use a 2-Kbyte database block size on file systems, you can prevent the read-modify-write phenomenon by using the `forcedirectio` mount option to bypass the file system on I/O operations.

Caution – If you are currently using UFS file systems and are not mounting them with the `forcedirectio` option, be careful to analyze your application before enabling direct I/O. It is quite possible that some of the objects in your database are benefiting from the UFS buffer cache. Turning on direct I/O can increase the amount of physical I/O and reduce the transaction rate as a result of bypassing the FS cache.

- If you are using direct I/O, make sure that you are using the Solaris 8 03/01 OE at minimum. The Solaris 8 03/01 OE provides Concurrent Direct I/O which eliminates the single writer lock that can dramatically improve performance.
- Mount file systems with the logging option. This saves on recovery time.
- If you are not mounting with the `forcedirectio` option, be aware of the `SEGMAP_PERCENT` default in the Solaris 8 OE. This variable was introduced to restrict the amount of memory used for address translations. By default, this value is 12 percent of physical memory. A large memory system with heavy UFS usage can benefit from a higher value.

Administration Tips

This section describes some things to be aware of when implementing Solaris Volume Manager software.

`iostat` Output Shows `sd` and `md` Devices

The `iostat` command is used to gather I/O performance information from the system. Solaris Volume Manager devices show up as `md` devices in the `iostat` output. In addition to Solaris Volume Manager devices, underlying disk statistics are displayed. Make sure that I/O is not accounted for multiple times when analyzing `iostat` output.

In the following output example, `md26` is a soft partition on top of `md2`, and `md2` is a meta device on top of `ssd3`. There is a total of 402.0 reads per second to the `md26` meta device. The system is not performing 3*402 = 1206 reads per second.

```
device        r/s    w/s     kr/s   kw/s ...
 md2        402.0    0.0  51458.7    0.0 ...
 md26       402.0    0.0  51460.5    0.0 ...
 ssd3       402.0    0.0  51463.6    0.0 ...
```

Raw Device Permissions

Solaris Volume Manager software does not directly manage device permissions. You can use the `chown` command to change permissions, but this change is not persistent across system reboots. To make device ownership persistent, modify the `/etc/minor_perm` file.

Summary

Intelligent storage arrays with hardware RAID and advanced configurational options have moved volume management intelligence into the storage solution, thereby decreasing the reliance on software volume management. Solaris Volume Manager software is able to match Veritas performance and does not require additional license fees. This makes Solaris Volume Manager software a compelling choice for software volume management.

The following list summarizes the key performance best practices for the Solaris Volume Manager software.

- **Avoid software striping** – Hardware striping is superior in performance and avoids the split I/O problem.
- **Make software stripe width large compared to the I/O size** – A large stripe width helps decrease the probability of splitting I/O.
- **Align soft partitions on stripe boundary** – If soft partitions are used on top of a striped meta device, make sure they are aligned to avoid splitting I/O.
- **Limit the number of meta devices when striping** – Do not correlate Veritas subdisk concepts when creating a Solaris Volume Manager layout.
- **Avoid software RAID 5** – Use hardware RAID 5 or software mirroring instead.
- **Increase I/O size when resyncing meta devices** – The default I/O size is too small, resulting in a longer resync time.
- **Use Sun StorEdge Traffic Manager to increase availability and throughput.**
- **Use an 8-Kbyte database block size for Oracle data files on UFS** – The Solaris OE uses only an 8-Kbyte I/O on file systems. If a smaller block size is used, extra I/O to *read, modify, then write* would occur on writes. Mounting with direction or using `init.ora` parameters can get around this, but requires careful planning.

About the Author

Glenn Fawcett has twelve years experience tuning large-scale database systems. As a member of the Sun Reference Architecture and Benchmarking group, Glenn and his colleagues are dedicated to devising and disseminating information on how to improve overall performance for various software packages and applications running in an enterprise environment. Glenn frequently serves as the project lead for performance benchmarks with Oracle for OLTP and DSS workloads. Glenn is also a regular speaker at the Sun Users Performance Group (SUPerG) conference.

References

- "Transitioning to Solaris Volume Manager" [Sun white paper, 2002]
- "Comprehensive data management using Solaris Volume Manager" [Sun white paper, 2002]
- *Solaris Volume Manager Administration Guide* from the Solaris 9 12/02 System Administrators Collection
- "Performance Oriented System Administration" a Sun BluePrints OnLine article by Robert Larson [December 2002].

  To access this article online, go to `http://www.sun.com/solutions/blueprints/browsesubject.html`
- "Oracle File System Integration and Performance" by Richard McDougall, and Sriram Gummuluru [August 2000].
  To access this article online, go to `http://www.sun.com/migration/ntmigration/nttech/rdbms.html`

Accessing Sun Documentation

You can view, print, or purchase a broad selection of Sun documentation, including localized versions, at:
`http://www.sun.com/documentation`

To reference Sun BluePrints™ OnLine articles, visit the Sun BluePrints OnLine web site at:
`http://www.sun.com/blueprints/online.html`