How to Size Main Memory for ZFS Deduplication

November 2011

by Dominic Kay

How to determine if enabling ZFS deduplication, which removes redundant data from ZFS file systems, will save you disk space without reducing performance.

What Is ZFS Deduplication?

In Oracle Solaris 11, you can use the deduplication (`dedup`) property to remove redundant data from your ZFS file systems. If a file system has the `dedup` property enabled, duplicate data blocks are removed as they are written to disk. The result is that only unique data is stored on disk and common components are shared between files, as shown in Figure 1.

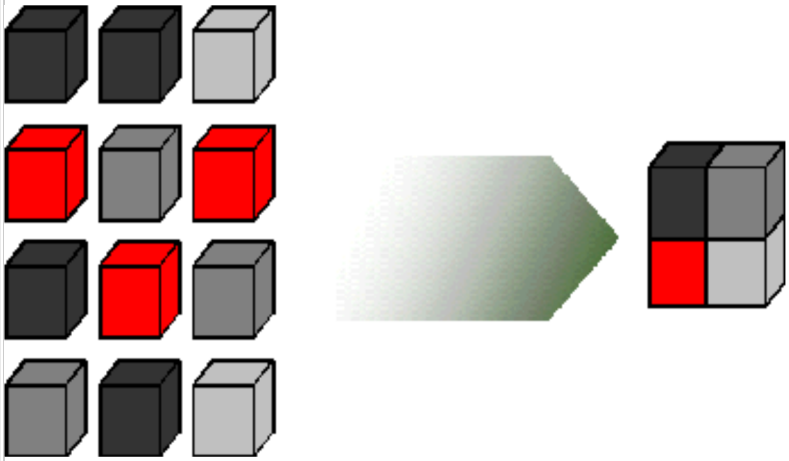

Figure 1. Only Unique Data Is Stored on Disk

In some cases, this can result in tremendous savings in disk space usage and cost. Deduplication is easily enabled for a file system, for example:

```
# zfs set dedup=on mypool/myfs
```

Deduplication can result in considerable storage space savings for certain types of data, such as virtual machine images. Other types of data, such as text, might more efficiently be stored using data compression, which is also available in ZFS.

Before starting to use deduplication, there are two issues that need to be investigated:

Is it worth using deduplication on this particular data?
Does the server have enough memory installed to undertake deduplication?
Guidance on these two issues is given below.

Is it Worth Using Deduplication on this Particular Data?

To determine if your data would benefit from deduplication space savings, use the ZFS debugging tool, `zdb`. If your data is not "dedup-able," there is no point in enabling `dedup`.

Deduplication is performed using checksums. If a block has the same checksum as a block that is already written to the pool, it is considered to be a duplicate and, thus, just a pointer to the already stored block is written to disk.

Therefore, the process of trying to deduplicate data that cannot be deduplicated simply wastes CPU resources. Deduplication in ZFS is in-band. The deduplication occurs when you write to the disk. This is when the (unnecessary) CPU load will be incurred.

For example, if the estimated deduplication ratio is greater than 2, you might see deduplication space savings. In the example shown in Listing 1, the deduplication ratio is less than 2, so enabling `dedup` is not recommended.

Listing 1: Determining the Deduplication Ratio

```
# zdb -S tank
Simulated DDT histogram:

bucket              allocated                       referenced
refcnt  blocks  LSIZE  PSIZE  DSIZE  blocks  LSIZE  PSIZE  DSIZE
------  ------  -----  -----  -----  ------  -----  -----  -----
     1  2.27M   239G   188G   194G   2.27M   239G   188G   194G
     2   327K  34.3G  27.8G  28.1G    698K  73.3G  59.2G  59.9G
     4  30.1K  2.91G  2.10G  2.11G    152K  14.9G  10.6G  10.6G
     8  7.73K   691M   529M   529M   74.5K  6.25G  4.79G  4.80G
    16    673  43.7M  25.8M  25.9M   13.1K   822M   492M   494M
    32    197  12.3M  7.02M  7.03M   7.66K   480M   269M   270M
    64     47  1.27M   626K   626K   3.86K   103M  51.2M  51.2M
   128     22   908K   250K   251K   3.71K   150M  40.3M  40.3M
   256      7   302K    48K  53.7K   2.27K  88.6M  17.3M  19.5M
   512      4   131K  7.50K  7.75K   2.74K   102M  5.62M  5.79M
    2K      1     2K     2K     2K   3.23K  6.47M  6.47M  6.47M
    8K      1   128K     5K     5K   13.9K  1.74G  69.5M  69.5M
 Total  2.63M   277G   218G   225G   3.22M   337G   263G   270G
dedup = 1.20, compress = 1.28, copies = 1.03,
dedup * compress / copies = 1.50
```

Does the Server Have Enough Memory Installed to Undertake Deduplication?

The reason this question needs to be answered is that the deduplication tables consume memory and eventually spill over and consume disk space. At that point, ZFS has to perform extra read and write operations for every block of data on which deduplication is attempted. This causes a reduction in performance.

Furthermore, the cause of the performance reduction will be difficult to determine if you are unaware that deduplication is active and can have adverse effects. A system that has large pools with small memory areas will not perform deduplication well. Some operations, such as removing a large file system with `dedup` enabled, will severely decrease system performance if the system doesn't meet the memory requirements.

Calculate memory requirement as follows:

Each in-core deduplication table (DDT) entry is approximately 320 bytes.
Multiply the number of allocated blocks by 320.
Here's an example using the data from the `zdb` information in Listing 1:

In-core DDT size (2.63M) x 320 = 841.60M of memory is required

Conclusion

After you evaluate the two constraints on deduplication, the deduplication ratio and the memory requirements, you can make a decision about whether to implement deduplication and what the likely savings will be.

For More Information

Here are some additional resources:

Download Oracle Solaris 11
Access Oracle Solaris 11 product documentation
Access all Oracle Solaris 11 how-to guides
Learn more with Oracle Solaris 11 training and support

Revision 1.0, 11/08/2011