close window

Web Exclusive

Print

AIX Flash Cache Statistics

July 2017 | by Jaqui Lynch

Flash Cache (also known as server side caching) has now been around for some time. The bigger challenge has been obtaining and interpreting performance statistics from flash cache as some of the output is not documented in the main pages. In fact, the best documentation on the fields is actually in the DS8800 easy tier documentation.

Flash cache allows an LPAR to use SSDs or flash storage as a read-only cache to improve read performance for spinning disks. In order to benefit from flash cache the workload must be a primarily read workload that re-reads data once it is cached. AIX will decide which data should be cached based on access patterns.

Once the system is up with flash cache installed and configured, then it is time to start the cache. This is done as follows:

```
cache_mgt cache start -t all
```

The command above will start caching for all the hdisks identified as sources for the cmpool. I have 8 x 700GB SSDs in my pool and 88 san disks (20TB) as my sources and it takes quite a while before they are all started. When they are all active you will see the following message:

All caches have been started.

Alternatively, you can start the disks one at a time by typing in: cache_mgt cache start -t hdisk?? replacing ?? with each hdisk name

When the cache is started you will see error log entries similar to the following:

```
C459CBDD   0528191417 I O hdisk12        AIX DISKDD RD/WR/STRAT SWITCHED TO ETCDD

D5BC7A29   0528191417 I O hdisk12        SAN DISK RD_CACHE IS ENABLED
D5BC7A29   0528191417 I O hdisk13        SAN DISK RD_CACHE IS ENABLED
D5BC7A29   0528191417 I O hdisk14        SAN DISK RD_CACHE IS ENABLED
```

Finally, you should see a message that the cache has warmed up.

```
61EC73EF   0529130617 I O ETCACHE        THE CACHE IS NOW WARM
```

As you can see in this case it took almost 20 hours for the cache to warm up—May 28 was a Sunday so the real workload did not start running until Monday, hence there was nothing to warm up the cache till then. But even on a regular day, it can take several hours for the cache to warm up in such a large environment.

When caching is enabled, read requests for the target devices are sent to the caching software, which checks whether the block is in the cache. If it is, then the disk block is provided from the cache and this is noted as a read hit. If some of the data is in the cache then it is a partial read hit. All other reads and all writes will be sent through to the original disk.

Once caching is up and running you can get statistics using the cache_mgt monitor command.

```
cache_mgt monitor get -h -s
```

The issue with the “monitor get” command is that it reports by source hdisk. This is fine if you have three or four source hdisks, but when you have 88 of them it is a lot of data to look at. It also reports since boot time or when the monitor was last started, which makes it difficult to get point in time statistics. One way around this is to use the undocumented pfcras command which provides shorter term statistics as well as an average over all the hdisks.

I set up a cron job to run at 11.59pm each night that grabs the monitor cache stats and stores them. The command I run is:

```
cache_mgt monitor get -h -s >>$logit.cachestats.txt
($logit is setup earlier in the script to be a name with date and time in it)
```

Additionally, I run the following command hourly:

```
pfcras -a dump_stats >>$logit.pfcras.txt
($logit is setup earlier in the script to be a name with date and time in it)
```

This way I have both the longterm and shortterm data saved.

The pfcras command is an undocumented command that shows the last 60 seconds and the last 3600 seconds for every hdisk, but it also provides an average for that time as well. I find pfcras to be far more useful.

After running pfcras or “cache_mgt monitor get” you then get a report. The cache_mgt report provides the following information for each hdisk: start time of statistics, read count, write count, read hit count, partial read hit count, read bytes transferred, write bytes transferred, read hit bytes transferred, partial read hit bytes transferred, promote read count and promote read bytes transferred. These are all counts since boot or since monitoring was started, not percentages. The output from this command is described in the etcadmin -a iostat documentation online.

The pfcras report starts with an overall global cache stats report and then provides information on the cache operations, overall disk statistics and then provides per-lun cache operations statistics for the source disks. It provides statistics for the last 60 and 3600 seconds, which is more granular than what you get out of the monitor report. There is no documentation for this command but you can figure most of it out.

The global cache section indicates whether the cache is warm or not and provides information on the amount of allocated space and the size of the valid data in the space. As an example, my most recent report showed:

```
Sector size:                 512 bytes
Fragment size:           1048576 bytes
Extent size:          1073741824 bytes
Cache size:        6201932775424 bytes

Cache state: Cache is warm

Reporting average statistics for the last 60 and 3600 seconds.

**********************
* Global Cache Stats *
**********************
```

```
Allocated Space:    6134760996864  (98.92% of total cache space)
Valid data:         6035699826688  (98.39% of allocated space)
```

The second section of the report shows actual cache operations statistics. These are same as what you will see reported for each source hdisk, except this is the global average.

Cache operations	60 sec	3600 sec
Hit Rate	98.50%	96.46%
Partial Hit Rate	0.00%	0.00%
Lookups	15105	1431764
Promotes	4	48555
Partial Promotes	0	0
Server Promotes	0	0
Invalidates	1	428
Purges	0	0

The hit rate is the percentage of read operations that were full read hits. i.e. where all the data requested was in the cache. In this report we see that in the last 60 seconds 98.5 percent of read requests were found in the read cache and the average in the past hour was 96.46 percent.

The partial read hit rate is the percentage of read operations where some portion of the read request was in the cache. Since our hit rate is so high it is not surprising that the partial rate is 0 percent.

Promotes are the percentage of read operations that were issued to the SAN as part of the promote into the cache.

The next two sections include the I/O statistics for the cache pool (DAS) and the SAN disks.

DAS I/O Stats

Avg. data read per second	10142010956	10097186003
Avg. read request size	606720	609280
Avg. read latency (usec)	2169	2339
Avg. data written per second	0	0
Avg. write request size	0	0
Avg. write latency (usec)	0	0

SAN I/O Stats

Avg. data read per second	46011	14238246
Avg. read request size	11776	1010176
Avg. read latency (usec)	67412	17428
Avg. data written per second	0	0
Avg. write request size	0	0
Avg. write latency (usec)	0	0

In the final section, there is a report for each source hdisk on the SAN. As with the global cache operations report there is a cache operations section that reports hit rates, etc for this hdisk, and there is also a LUN I/O statistics section that contains information such as data read per second for this hdisk. When monitoring flash cache performance, it is also useful to look at nmon to monitor the cmpool0 volume group. The VGBUSY, VGREAD and VGWRITE tabs in the nmon analyser report will provide very useful data.

Summary

For the right workload, flash cache can make a significant performance difference. We use a combination of real flash for part of our workload and the flash cache to front end the other part of the workload. This combination has made an enormous difference to performance and to the time it takes jobs to run, especially jobs with large sorts or that process large amounts of data. Over time I am hoping that better documentation will come out for the performance monitoring tools, but in the meantime the combination of nmon, pfcras and cache_mgt monitor get should be able to provide you with a strong indicator as to whether flash cache is beneficial for your workload or not.