IBM Spectrum Scale:

Performance and

.. field report ...

Olaf Weiser

IBM

Disclaimer

- IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.
- IBM reserves the right to change product specifications and offerings at any time without notice. This publication could include technical inaccuracies or typographical errors. References herein to IBM products and services do not imply that IBM intends to make them available in all countries.

optimized code for small DIO

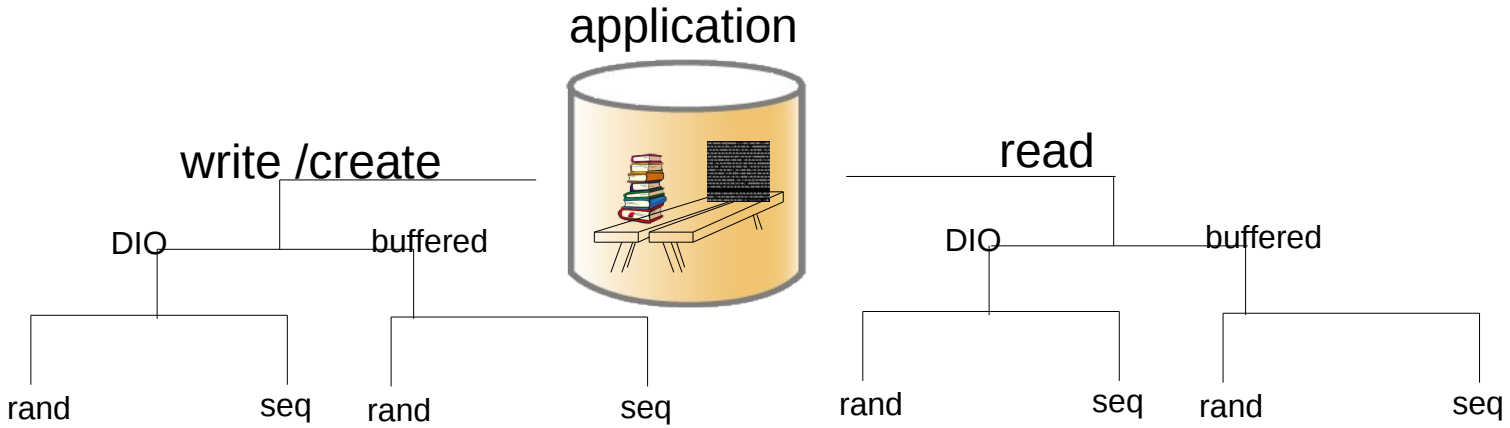
DMA enhancement

Performance improvement for databases and small IO

IBM Spectrum Scale

Challenge:

– Data bases writing into large files, but with small IO, mostly O_DIRECT



– wide range of different IO patters
– IO size : block aligned or 4K...256K or even up to 1M … 64M … or something in between
– improving one workload can impact others and vice versa

A never documented work around is now obsolete

– now, hereby we document ***disableDIO***

A never documented work around is now obsolete

– now, hereby we document ***disableDIO***

– you **should not** use ***disableDIO***

USE the new setting from the following slides

Performance improvement for databases and small IO application

IBM Spectrum Scale

old code , gpfs R < 5.0.4.2

```
[root@oldNode]# cat <<EOF > /tmp/aioseq.fio
> [seq-aio-dio-write]
> filename=f10G
> rw=write
> direct=1
> ioengine=libaio
> iodepth=128
> bs=32k
> size=10g
> EOF
[root@oldNode]# fio --directory=/gpfs/ess3k1M /tmp/aioseq.fio | grep WRITE
 WRITE: bw=260MiB/s (273MB/s), 260MiB/s-260MiB/s (273MB/s-273MB/s), io=10.0GiB (10.7GB), run=39356-39356msec
[root@fscc-fab3-2-a lib]#
```

setting:

– old code,

– regular GPFS behavior

– disableDIO=default (not set)

behind the scenes

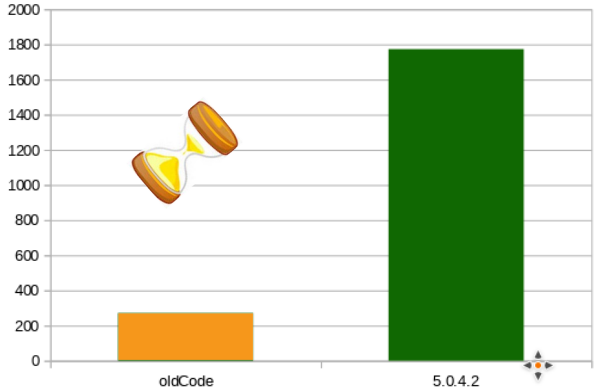
```
07:41:18.858672  W  data  3:13762240  64  0.757  31507  7864  C0A82D14:5E4C4BB8 cli  10.10.10.121 MBHandler AioWorkerT
07:41:18.858688  W  data  3:13762304  64  0.758  31507  7864  C0A82D14:5E4C4BB8 cli  10.10.10.121 MBHandler AioWorkerT
07:41:18.858412  W  data  3:13761088  64  1.041  31507  7864  C0A82D14:5E4C4BB8 cli  10.10.10.121 MBHandler AioWorkerThread
07:41:18.858721  W  data  3:13762432  64  0.740  31507  7864  C0A82D14:5E4C4BB8 cli  10.10.10.121 MBHandler AioWorkerThread
07:41:18.858480  W  data  3:13761408  64  0.993  31507  7864  C0A82D14:5E4C4BB8 cli  10.10.10.121 MBHandler AioWorkerThread
07:41:18.858597  W  data  3:13761984  64  0.885  31507  7864  C0A82D14:5E4C4BB8 cli  10.10.10.121 MBHandler AioWorkerThread
07:41:18.858647  W  data  3:13762112  64  0.842  31507  7864  C0A82D14:5E4C4BB8 cli  10.10.10.121 MBHandler AioWorkerThread
```

very good response times from physic / but poor IO pattern

Performance improvement for databases and small IO application

IBM Spectrum Scale

new* code , gpfs *5.0.4.2



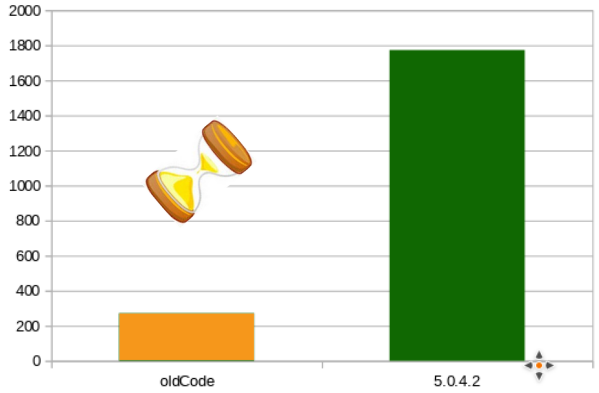

```
[root@newNode]# cat <<EOF > /tmp/aioseq.fio
> [seq-aio-dio-write]
> filename=f10G-1
> rw=write
> direct=1
> ioengine=libaio
> iodepth=128
> bs=32k
> size=10g
> EOF
[root@newNode home]# fio --directory=/gpfs/ess3k1M /tmp/aioseq.fio | grep WRITE
  WRITE: bw=1710MiB/s (1793MB/s), 1710MiB/s-1710MiB/s (1793MB/s-1793MB/s), io=10.0GiB (10.7GB), run=5989-5989msec
[root@newNode home]#
```

behind the scenes

very good response times from physic

```
07:47:51.038923  W  data  2:483407872  2048  0.834  31510  2931  C0A82D14:5E4C4BBB cli  10.10.10.121 Prefetch  WritebehindW
07:47:51.039465  W  data  3:474421248  2048  0.937  31510  2932  C0A82D14:5E4C4BB8 cli  10.10.10.121 Prefetch  WritebehindWorkerThread
07:47:51.040027  W  data  4:485654528  2048  1.066  31510  2933  C0A82D14:5E4C4BB9 cli  10.10.10.121 Prefetch  WritebehindWorkerThread
07:47:51.040639  W  data  1:487901184  2048  0.867  31510  2934  C0A82D14:5E4C4BBA cli  10.10.10.121 MBHandler AioWorkerThread
07:47:51.041171  W  data  2:485654528  2048  0.975  31510  2935  C0A82D14:5E4C4BBB cli  10.10.10.121 Prefetch  WritebehindWorkerThread
07:47:51.041712  W  data  3:476667904  2048  0.995  31510  2936  C0A82D14:5E4C4BB8 cli  10.10.10.121 Prefetch  WritebehindWorkerThread
07:47:51.042337  W  data  4:487901184  2048  1.068  31510  2937  C0A82D14:5E4C4BB9 cli  10.10.10.121 Prefetch  WritebehindWorkerThread
07:47:51.042839  W  data  1:490147840  2048  1.150  31510  2938  C0A82D14:5E4C4BBA cli  10.10.10.121 MBHandler AioWorkerThread
```

Performance improvement for databases and small IO application



– to enable the new enhancement

[root@newNode]# mmdiag --config | grep -e **dioSmallSeqWriteBatching**
dioSmallSeqWriteBatching 1
[root@newNode home]#

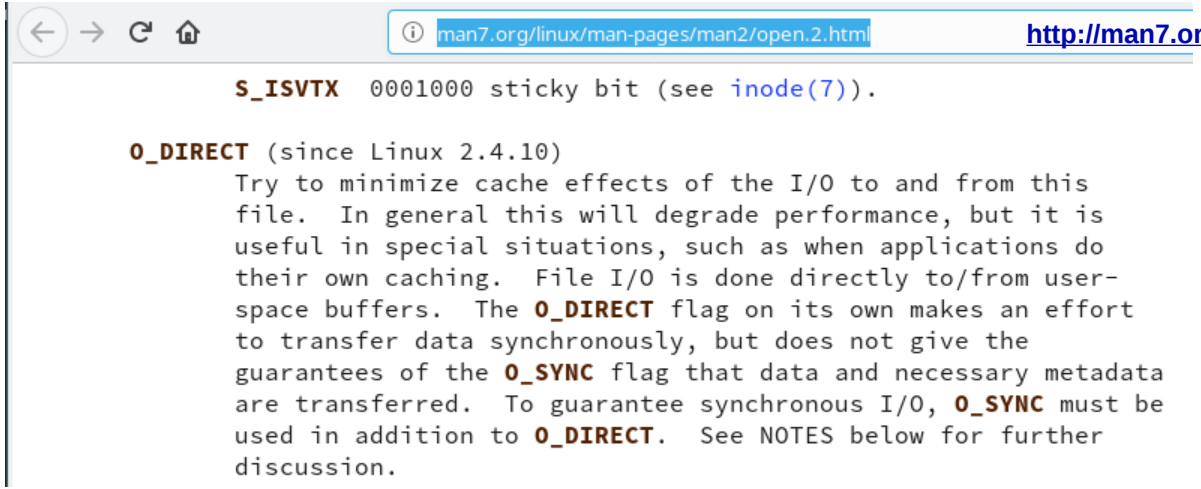
– can be set dynamically (-i) and per node/nodeclass

– aioSyncDelay is used, retrieved from the dioSmallSeqWriteBatching parameter

– **the definition:**

Add a heuristic that executes small sequential AIO/DIO writes as buffered I/O (+ sync) so that multiple small writes can be combined into a single, larger I/O. Data integrity is guaranteed.

Performance improvement for databases and small IO application



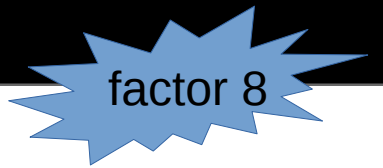
http://man7.org/linux/man-pages/man2/open.2.html

So .. SpectrumScale behaves, that data integraty is assured. O_DIRECT plus! O_SYNC
When ISS acknowledges a WRITE, it is stored safely on persistent storage as intented (or requested) .

A more realistic example from the field

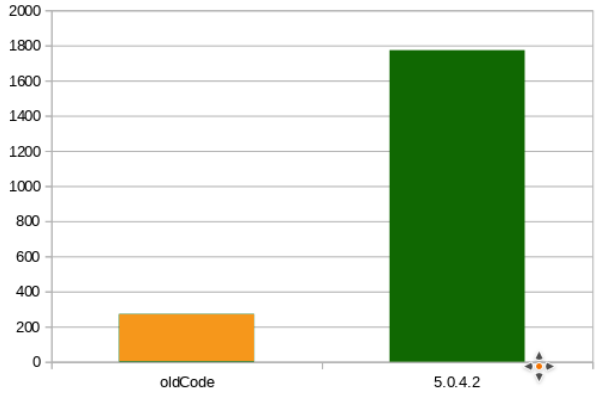
IBM Spectrum Scale

```
root@newNode /home/hwcct240/h/lib>mmdiag  --config | grep -e  dioSmallSeqWriteBatching
 # dioSmallSeqWriteBatching 0
root@newNode /home/hwcct240/h/lib>./fsperf -i sequential -o verbose -m throughput -f 5G -b 64K /gpfs/ess3k1M | grep -i "I/O time:..."
   I/O time:.....    24.3083 s   (Throughput:     210.6 MB/s,     3370.0 op/s)
   Ratio trigger time to I/O time:.0.00027
   I/O time:.....     3.1444 s   (Throughput:     1628.2 MB/s,    26052.6 op/s)
   Ratio trigger time to I/O time:.0.00239
   I/O time:.....     1.3845 s   (Throughput:     3697.9 MB/s,    59166.8 op/s)
   Ratio trigger time to I/O time:.0.00619
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>mmdiag  --config | grep -e  dioSmallSeqWriteBatching -e aioSyncDelay
 # aioSyncDelay 10 (implicit via dioSmallSeqWriteBatching)
 # dioSmallSeqWriteBatching 1
root@newNode /home/hwcct240/h/lib>./fsperf -i sequential -o verbose -m throughput -f 5G -b 64K /gpfs/ess3k1M | grep -i "I/O time:..."
   I/O time:.....     2.9232 s   (Throughput:     1751.4 MB/s,    28023.8 op/s)
   Ratio trigger time to I/O time:.0.00257
   I/O time:.....     2.6425 s   (Throughput:     1937.5 MB/s,    31000.4 op/s)
   Ratio trigger time to I/O time:.0.00270
   I/O time:.....     1.3003 s   (Throughput:     3937.4 MB/s,    62999.4 op/s)
   Ratio trigger time to I/O time:.0.00559
root@newNode /home/hwcct240/h/lib>
```



A more realistic example from the field

```
root@newNode /home/hwcct240/h/lib>mmdiag  --config | grep -e  dioSmallSeqWriteBatching
 # dioSmallSeqWriteBatching 0
root@newNode /home/hwcct240/h/lib>./fsperf -i sequential -o verbose -m throughput -f 5G -b 64K /gpfs/ess3k1M | grep -i "I/O time:..."
   I/O time:.....     24.3083 s   (Throughput:     210.6 MB/s,    3370.0 op/s)
   Ratio trigger time to I/O time:.0.00027
   I/O time:.....      3.1444 s   (Throughput:    1628.2 MB/s,   26052.6 op/s)
   Ratio trigger time to I/O time:.0.00239
   I/O time:.....      1.3845 s   (Throughput:    3697.9 MB/s,   59166.8 op/s)
   Ratio trigger time to I/O time:.0.00619
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>
root@newNode /home/hwcct240/h/lib>mmdiag  --config | grep -e  dioSmallSeqWriteBatching -e aioSyncDelay
 # aioSyncDelay 10 (implicit via dioSmallSeqWriteBatching)
 # dioSmallSeqWriteBatching 1
root@newNode /home/hwcct240/h/lib>./fsperf -i sequential -o verbose -m throughput -f 5G -b 64K /gpfs/ess3k1M | grep -i "I/O time:..."
   I/O time:.....      2.9232 s   (Throughput:    1751.4 MB/s,   28023.8 op/s)
   Ratio trigger time to I/O time:.0.00257
   I/O time:.....      2.6425 s   (Throughput:    1937.5 MB/s,   31000.4 op/s)
   Ratio trigger time to I/O time:.0.00270
   I/O time:.....      1.3003 s   (Throughput:    3937.4 MB/s,   62999.4 op/s)
   Ratio trigger time to I/O time:.0.00559
root@newNode /home/hwcct240/h/lib>
```



– the performance improvement is depending on

  – networt bandwidth
  – back end / disk capabilities
  – client node‘s resources

– the faster the backend is (SSD, NVMe) . .the more you benefit from the new code

– special THANKs to the ***BOSCH team in Suttgart Feuerbach*** for providing the approval to publish this numbers

Summary:

IBM Spectrum Scale

```
[root@ems1 ~]# mmchconfig dioSmallSeqWriteBatching=yes -i
mmchconfig: Command successfully completed
gssio1rd.test:  Unknown config name: dioSmallSeqWriteBatching
gssio2rd.test:  Unknown config name: dioSmallSeqWriteBatching
[..]
```

– it is a client setting
– NSD server don‘t need the new code
– In this case, you can ignore this msg

parameters:

dioSmallSeqWriteBatching	[yes,no]	default is [no]	enable/disable new code enhancement
dioSmallSeqWriteThreshold	#bytes	default is 64K	By default, the optimization kicks in when we see three AIO/DIO writes that are no larger *dioSmallSeqWriteThreshold* bytes each

introduced in GPFS 5.0.4.2 (PTF2)

...from the field...

Tiny little details , but very helpful ;-)

– last recent releases

– cluster hang situation

– adjust NSDworker to the environment

SpectrumScale – file system format

In IBM Spectrum Scale 5.0.4, new file systems are created at file system format level 22.00. To update a file system from an earlier format to format level 22.00, issue the following command:

```
mmchfs Device -V full
```

where *Device* is the device name of the file system. The following features of IBM Spectrum Scale 5.0.4 require a file system to be at format number 22.00 or later:

- Support for thin provisioned storage devices and NVMe SSDs.
- Support for linking GPFS dependent filesets inside AFM and AFM-DR filesets.

SpectrumScale – sort file system / release versions to version string

IBM Spectrum Scale

[root@fscc-fab3-2-a ~]# cat /usr/lpp/mmfs/bin/mmglobfuncs |grep "(806)" -B 1 -A 60

```
#  format     Release        Notes
#     0     2.3.0.2 (806)    base GPFS 2.3 release
#     1     3.1.0.1 (902)    base GPFS 3.1 release
#     2     3.1.0.3 (904)    support for sdp sockets (obsolete)
#     3     3.2.0.0 (1002)   base GPFS 3.2 release
#     4     3.2.1.3 (1008)   FGDL release
#     5     3.2.1.5 (1010)   Windows release
#     6     3.2.1.6 (1011)   support for external attributes in inodes
#  1100   3.3.0.0 (1100)   GPFS 3.3 initial development
#  1101   3.3.0.0 (1100)   GPFS 3.3 intermidiate development
#  1102   3.3.0.0 (1102)   GPFS 3.3 intermidiate development
#  1103   3.3.0.0 (1103)   GPFS 3.3 official base release
#  1105   3.3.0.2 (1105)   GPFS 3.3 restore dmapi
#  1200   3.4.0.0 (1200)   GPFS 3.4 base release (planned)
#  1201   3.4.0.0 (1201)   GPFS 3.4 enable full inode64 & per fileset quota
#  1202   3.4.0.0 (1202)   GPFS 3.4 enable FILESETSV2
#  1203   3.4.0.0 (1203)   GPFS 3.4 base release (actual)
#  1206   3.4.0.3 (1206)   GPFS 3.4 different metadata block size
#  1207   3.4.0.4 (1207)   GPFS 3.4 striped logs & user level fileset commands
#  1210   3.4.0.7 (1210)   GPFS 3.4 GPFS-SNC
#  1300   3.5.0.0 (1300)   GPFS 3.5 base release (planned)
#  1301   3.5.0.0 (1301)   GPFS 3.5 enable SNC
#  1302   3.5.0.0 (1302)   GPFS 3.5 store IPv6 in compressed form
#  1305   3.5.0.3 (1305)   GPFS 3.5 TL1
#  1320   3.5.0.7 (1320)   GPFS 3.5 TL2 base release (planned)
#  1321   3.5.0.7 (1321)   GPFS 3.5 pool properties
#  1322   3.5.0.7 (1322)   GPFS 3.5 pool properties + locality group vector
#  1323   3.5.0.7 (1323)   GPFS 3.5 pools + locality group vector + encryption
#  1340   3.5.0.11 (1340)  GPFS 3.5 TL3
#  1400   4.1.0.0 (1400)   GPFS 4.1 base release (planned)
#  1401   4.1.0.0 (1401)   GPFS 4.1 enable tscomm41
#  1402   4.1.0.0 (1402)   GPFS 4.1 enable CCR and Config41
#  1403   4.1.0.0 (1403)   GPFS 4.1 enable GPT_NSD and DISK_4K_SECTOR
#  1404   4.1.0.0 (1404)   GPFS 4.1 enable quota in sgDesc
#  1410   4.1.0.4 (1410)   GPFS 4.1 TL1
#  1420   4.1.1.0 (1420)   GPFS 4.1 TL2 initial development
#  1421   4.1.1.0 (1421)   GPFS 4.1 TL2 enable QOS
#  1422   4.1.1.0 (1422)   GPFS 4.1 TL2 enable fileset compliance plus semantics
#  1423   4.1.1.0 (1423)   GPFS 4.1 TL2 enable inode-expansion v2 semantics
#  1427   4.1.1.4 (1427)   GPFS 4.1 TL2 PTF4 enable 4KN dataOnly disk in non-4K aligned FS
#  1443   4.1.1.20 (1443)  GPFS 4.1.1.20 rapid repair plus
#  1500   4.2.0.0 (1500)   GPFS 4.2 base release enable CSTORE
#  1501   4.2.0.0 (1501)   GPFS 4.2 base release enable ZIP
#  1502   4.2.0.1 (1502)   GPFS 4.2 PTF1 CCR security (no fsVersion updated)
#  1510   4.2.1.0 (1510)   GPFS 4.2 TL1 ENC CONSUMABILITY (no fsVer updated)
#  1511   4.2.1.1 (1511)   GPFS 4.2 PTF1 noAuthentication (no fsVer updated)
#  1600   4.2.2.0 (1600)   GPFS 4.2 TL2 initial development
#  1700   4.2.3.0 (1700)   GPFS 4.2 TL3 initial development
#  1709   4.2.3.9 (1709)   GPFS 4.2.3.9 rapid repair plus
#  1800   5.0.0.0 (1800)   GPFS 5.0 CORAL release
#  1900   5.0.1.0 (1900)   GPFS 5.0.1 initial development (no fsVer updated)
#  1901   5.0.1.0 (1901)   GPFS 5.0.1 rapid repair plus
#  2000   5.0.2.0 (2000)   GPFS 5.0.2 initial development
#  2001   5.0.2.0 (2001)   GPFS 5.0.2 --auto-resume for tschdisk suspend
#  2100   5.0.3.0 (2100)   GPFS 5.0.3 Genomic compression
#  2200   5.0.4.0 (2200)   GPFS 5.0.4 Thin provisioning
```

SpectrumScale – file system format - cont ()

IBM Spectrum Scale

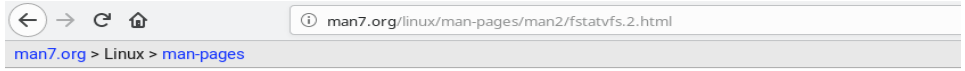
In IBM Spectrum Scale 5.0.0, new file systems are created at format number 18.00. To update the format of an earlier file system to format number 18.00, issue the following command:

```
mmchfs Device -V full
```

where *Device* is the device name of the earlier file system. The following features of IBM Spectrum Scale 5.0.0 require a file system to be at format number 18.00 or later:

- Smaller subblock sizes for file systems that have a large data block size

SpectrumScale – file system format - cont ()



NAME | SYNOPSIS | DESCRIPTION | RETURN VALUE | ERRORS | ATTRIBUTES | CONFORMING TO | NOTES | SEE ALSO | COLOPHON

STATVFS(3) **Linux Programmer's Manual** **STATVFS(3)**

NAME top

statvfs, fstatvfs - get filesystem statistics

SYNOPSIS top

```
#include <sys/statvfs.h>

int statvfs(const char *path, struct statvfs *buf);
int fstatvfs(int fd, struct statvfs *buf);
```

DESCRIPTION top

The function **statvfs**() returns information about a mounted filesystem. *path* is the pathname of any file within the mounted filesystem. *buf* is a pointer to a *statvfs* structure defined approximately as follows:

```
struct statvfs {
    unsigned long  f_bsize;    /* Filesystem block size */
    unsigned long  f_frsize;   /* Fragment size */
    fsblkcnt_t     f_blocks;   /* Size of fs in f_frsize units */
```

– according to linux docs, there is
  (1) fragment size
  (2) block size

http://man7.org/linux/man-pages/man2/fstatvfs.2.html

SpectrumScale – file system format - cont ()

```
[root@gssio1 essGL2_16M]# cat myfree.c | tail -10
                return;
        }

        printf("%s, mounted on %s:\n", fs->mnt_dir, fs->mnt_fsname);

        /* printf("\tf_type: %s\n", type2str(vfs.f_type)); */
        printf("\tf_bsize: %ld\n", vfs.f_bsize);
        printf("\tf_frsize: %ld\n", vfs.f_frsize);
}
```

```
#include <sys/types.h>
#include <sys/vfs.h>
```

default

SpectrumScale – file system format - cont ()

IBM Spectrum Scale

```
[root@gssio1 essGL2_16M]# cat myfree.c | tail -10
                return;
        }

        printf("%s, mounted on %s:\n", fs->mnt_dir, fs->mnt_fsname);

        /* printf("\tf_type: %s\n", type2str(vfs.f_type)); */
        printf("\tf_bsize: %ld\n", vfs.f_bsize);
        printf("\tf_frsize: %ld\n", vfs.f_frsize);
}
```

```
#include <sys/types.h>
#include <sys/vfs.h>
```

default

```
[root@ems1 essGL2_16M]# ./a.out
/gpfs/essGL2_16M, mounted on /gpfs/essGL2_16M:
        f_bsize: 16777216
        f_frsize: 16777216
[root@ems1 essGL2_16M]#
[root@ems1 essGL2_16M]#
```

```
[root@ems1 essGL2_16M]# mmfsadm dump config | grep -i linuxstatfsUnits
 # linuxStatfsUnits fullblock
```

SpectrumScale – file system format - cont ()

IBM Spectrum Scale

– new parameter to control behavior for statfs

```
[root@ems1 essGL2_16M]# mmchconfig linuxStatfsUnits=subblock -i
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
[root@ems1 essGL2_16M]# mmfsadm dump config | grep -i linuxstatfsUnits
 # linuxStatfsUnits subblock
[root@ems1 essGL2_16M]# ./a.out
/gpfs/essGL2_16M, mounted on /gpfs/essGL2_16M:
        f_bsize: 16384
        f_frsize: 16384
[root@ems1 essGL2_16M]#
```

```
struct statvfs {
    unsigned long  f_bsize;    /* Filesystem block size */
    unsigned long  f_frsize;   /* Fragment size */
    fsblkcnt_t     f_blocks;   /* Size of fs in f_frsize units */
```

SpectrumScale – file system format - cont ()

IBM Spectrum Scale

– new parameter to control behavior for statfs [fullblock,subblock,posix]

```
[root@ems1 essGL2_16M]# mmchconfig linuxStatfsUnits=POSIX -i
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
[root@ems1 essGL2_16M]# mmfsadm dump config | grep -i linuxstatfsUnits
 # linuxStatfsUnits posix
[root@ems1 essGL2_16M]# ./a.out
/gpfs/essGL2_16M, mounted on /gpfs/essGL2_16M:
        f_bsize: 16777216
        f_frsize: 16384
[root@ems1 essGL2_16M]#
```

```
struct statvfs {
    unsigned long  f_bsize;    /* Filesystem block size */
    unsigned long  f_frsize;   /* Fragment size */
    fsblkcnt_t     f_blocks;   /* Size of fs in f_frsize units */
```

Dead lock
or
my cluster seems to hang

SpectrumScale – deadlock detection

IBM Spectrum Scale

```
[root@fscc-sr650-14 ~]# mmlsnode -N waiters -L
ece_14.localnet.com:  Waiting 0.0316 sec since 18:14:55, monitored, thread 407779 WritebehindWorkerThread: on ThCond 0x7FD96401A4C0 (MsgRecordCondva
n node 10.0.12.16 <c0n4>
ece_14.localnet.com:  Waiting 0.0213 sec since 18:14:55, monitored, thread 407774 WritebehindWorkerThread: on ThCond 0x7FD91C00E610 (MsgRecordCondva
n node 10.0.12.17 <c0n2>
ece_14.localnet.com:  Waiting 0.0193 sec since 18:14:55, monitored, thread 407777 WritebehindWorkerThread: on ThCond 0x7FDA44039EB0 (IOBundleNSPDCon
ece_14.localnet.com:  Waiting 0.0189 sec since 18:14:55, monitored, thread 407760 WritebehindWorkerThread: on ThCond 0x7FD96401B1C0 (MsgRecordCondva
n node 10.0.12.15 <c0n3>
ece_14.localnet.com:  Waiting 0.0173 sec since 18:14:55, monitored, thread 407804 PrefetchWorkerThread: on ThCond 0x7FD8A801ED30 (MsgRecordCondvar),
ode 10.0.12.16 <c0n4>
ece_14.localnet.com:  Waiting 0.0163 sec since 18:14:55, monitored, thread 407770 WritebehindWorkerThread: on ThCond 0x7FD8C8016470 (MsgRecordCondva
n node 10.0.12.16 <c0n4>
ece_14.localnet.com:  Waiting 0.0153 sec since 18:14:55, monitored, thread 407811 WritebehindWorkerThread: on ThCond 0x7FD92C00D470 (MsgRecordCondva
n node 10.0.12.18 <c0n5>
ece_14.localnet.com:  Waiting 0.0116 sec since 18:14:55, monitored, thread 407812 WritebehindWorkerThread: on ThCond 0x7FD80801A680 (MsgRecordCondva
n node 10.0.12.13 <c0n1>
ece_14.localnet.com:  Waiting 0.0109 sec since 18:14:55, monitored, thread 407797 PrefetchWorkerThread: on ThCond 0x7FD9580F55E0 (MsgRecordCondvar),
ode 10.0.12.16 <c0n4>
ece_14.localnet.com:  Waiting 0.0093 sec since 18:14:55, monitored, thread 407769 WritebehindWorkerThread: on ThCond 0x7FDA44039C10 (IOBundleNSPDCon
ece_14.localnet.com:  Waiting 0.0073 sec since 18:14:55, monitored, thread 407781 WritebehindWorkerThread: on ThCond 0x7FD88C01D540 (MsgRecordCondva
n node 10.0.12.17 <c0n2>
ece_14.localnet.com:  Waiting 0.0073 sec since 18:14:55, monitored, thread 407796 PrefetchWorkerThread: for NSD I/O completion on node 10.0.12.15 <c
ece_14.localnet.com:  Waiting 0.0068 sec since 18:14:55, monitored, thread 407782 WritebehindWorkerThread: on ThCond 0x7FD9FC00E460 (MsgRecordCondva
n node 10.0.12.15 <c0n3>
ece_14.localnet.com:  Waiting 0.0051 sec since 18:14:55, monitored, thread 407763 PrefetchWorkerThread: for NSD I/O completion on node 10.0.12.18 <c
```

command: mmfsadm -N waiters -L -s [x]

GPFS – dead lock / waiters

IBM Spectrum Scale

All waiters can be broadly divided into four categories:

[1] Waiters that can occur under normal operating conditions and can be ignored by automated deadlock detection.

[2] Waiters that correspond to complex operations and can legitimately grow to moderate lengths.

[3] Waiters that should never be long. For example, most mutexes should only be held briefly.

[4] Waiters that can be used as an indicator of cluster overload. For example, waiters waiting for I/O completions or network availability.

SpectrumScale – deadlock detection

IBM Spectrum Scale

– Monitor waiters in core GPFS code
– Configurable tresholds
– Skip waiters which can be legitimately long
    for e.g. PIT worker

```
waiting on ThCond 0x1110CDD60 (0x1110CDD60) (PitCondvar), reason 'Waiting until pit work is complete' (Long)
```

some waiters to detect overload

– “NSD I/O completion”
– “waiting for exclusive use of connection for sending msg”

.. debug data …
simple investigating waiters

IBM Spectrum Scale

```
0x1107DA670 waiting 3.012068863 seconds, Msg handler mnMsgForceInodeFlags: on ThMutex 0x110617830 (0x110617830)
(LogFile instance)
0x1107CDEB0 waiting 2.835512707 seconds, SG mgr log migrate: for open disk device on disk prodracZ_D1
0x11077C7F0 waiting 28.007734837 seconds, SG Exception LogBufferFull: on ThMutex 0x110617830 (0x110617830)
(LogFile instance)
0x11009FD50 waiting 28.012088486 seconds, Sync handler: on ThCond 0x110617898 (0x110617898) (LogFile buffer
state), reason 'force wait for write complete'
0x11009FA90 waiting 28.010270437 seconds, BRT handler: on ThCond 0x110621838 (0x110621838) (MsgRecord), reason
'waiting for RPC replies' for tmMsgRevoke on node 10.1.11.51
```

– you could … collect further debug data, by snap and trace

.. debug data …
simple investigating waiters

IBM Spectrum Scale

```
   8.032398  17278 TRACE_TS: sgm_rpc_start(origErr 0, flags 0x0): sgMgr 192.168.1.4 err 0

   8.032401  17278 TRACE_TS: tscSend: service 00020001 msg 'sgmMsgSGMount' n_dest 1 data_len 8 msg_id 32
                            msg 0x85AC368 mr 0x85AC298

   8.032402  17278 TRACE_TS: llc_send_msg: cl 0, dest 192.168.1.4, msg_id 32, type 1, len 8

   8.032403  17278 TRACE_TS: acquireConn enter: addr 192.168.1.4 nodeidx 2 add 1
   8.032404  17278 TRACE_TS: acquireConn exit: err 0
   8.032420  17278 TRACE_TS: llc_send_msg: returning 0

   8.032421  17278 TRACE_MUTEX: Thread 0x13C02 (Mount handler) waiting on condvar 0x85AC350
                               (0x85AC350) (MsgRecord): waiting for RPC replies

===== dump waiters =====
0x855E3B0 waiting 8.269320000 seconds, Mount handler: on ThCond 0x85AC350 (0x85AC350) (MsgRecord),
          reason 'waiting for RPC replies' for sgmMsgSGMount on node 192.168.1.4

===== dump tscomm =====
Pending messages:
  msg_id   32, service 2.1, msg_type  1 'sgmMsgSGMount', n_dest 1, n_pending 1
  this 0x85AC298, n_xhold 1, ccP 0x905BB548 cbFn 0x0
    sent by 'Mount handler' (0x855E3B0)
    dest 192.168.1.4     status pending   , err 0, reply len 0
```

.. debug data …
simple investigating waiters

IBM Spectrum Scale

```
   8.032398  17278 TRACE_TS: sgm_rpc_start(origErr 0, flags 0x0): sgMgr 192.168.1.4 err 0

   8.032401  17278 TRACE_TS: tscSend: service 00020001 msg 'sgmMsgSGMount' n_dest 1 data_len 8 msg_id 32
                            msg 0x85AC368 mr 0x85AC298

   8.032402  17278 TRACE_TS: llc_send_msg: cl 0, dest 192.168.1.4, msg_id 32, type 1, len 8

   8.032403  17278 TRACE_TS: acquireConn enter: addr 192.168.1.4 nodeidx 2 add 1
   8.032404  17278 TRACE_TS: acquireConn exit: err 0
   8.032420  17278 TRACE_TS: llc_send_msg: returning 0

   8.032421  17278 TRACE_MUTEX: Thread 0x13C02 (Mount handler) waiting on condvar 0x85AC350
                                (0x85AC350) (MsgRecord): waiting for RPC replies

===== dump waiters =====
0x855E3B0 waiting 8.269320000 seconds, Mount handler: on ThCond 0x85AC350 (0x85AC350) (MsgRecord),
          reason 'waiting for RPC replies' for sgmMsgSGMount on node 192.168.1.4

===== dump tscomm =====
Pending messages:
  msg_id   32, service 2.1, msg_type  1 'sgmMsgSGMount', n_dest 1, n_pending 1
  this 0x85AC298, n_xhold 1, ccP 0x905BB548 cbFn 0x0
    sent by 'Mount handler' (0x855E3B0)
    dest 192.168.1.4     status pending   , err 0, reply len 0
```

SpectrumScale – deadlock detection

IBM Spectrum Scale

– DeadlockDetectionThreshold

– 0 to disable automated deadlock detection

– to enable … set a value in **seconds**

– configurable dynamically

Adjust according to workload to avoid false alarms in deadlock detection

```
mmfsadm dump config |grep dead
   deadlockBreakupDelay 0
   deadlockDataCollectionDailyLimit 10
   deadlockDataCollectionMinInterval 600
   deadlockDetectionThreshold 300
   deadlockDetectionThresholdIfOverloaded 1800
   [...]
```

SpectrumScale – deadlock detection

IBM Spectrum Scale

– DeadlockDetectionThreshold

– 0 to disable automated deadlock detection

– to enable … set a value in **seconds**

– configurable dynamically

Adjust according to workload to avoid false alarms in deadlock detection

```
mmfsadm dump config |grep dead
   deadlockBreakupDelay 0
   deadlockDataCollectionDailyLimit 10
   deadlockDataCollectionMinInterval 600
   deadlockDetectionThreshold 300
   deadlockDetectionThresholdIfOverloaded 1800
   [...]
```

SpectrumScale – deadlock detection

IBM Spectrum Scale

Adjust according to workload to avoid false alarms in deadlock detection

deadlock detected

automated debug collection
- enabled by default
- tuneable

```
on each node
internaldump.140312.22.20.24.deadlock.hs22n65.
kthreads.140312.22.20.24.deadlock.hs22n65.gz
trcrpt.140312.22.20.24.deadlock.hs22n65.gz
```

automated break up
- disabled by default

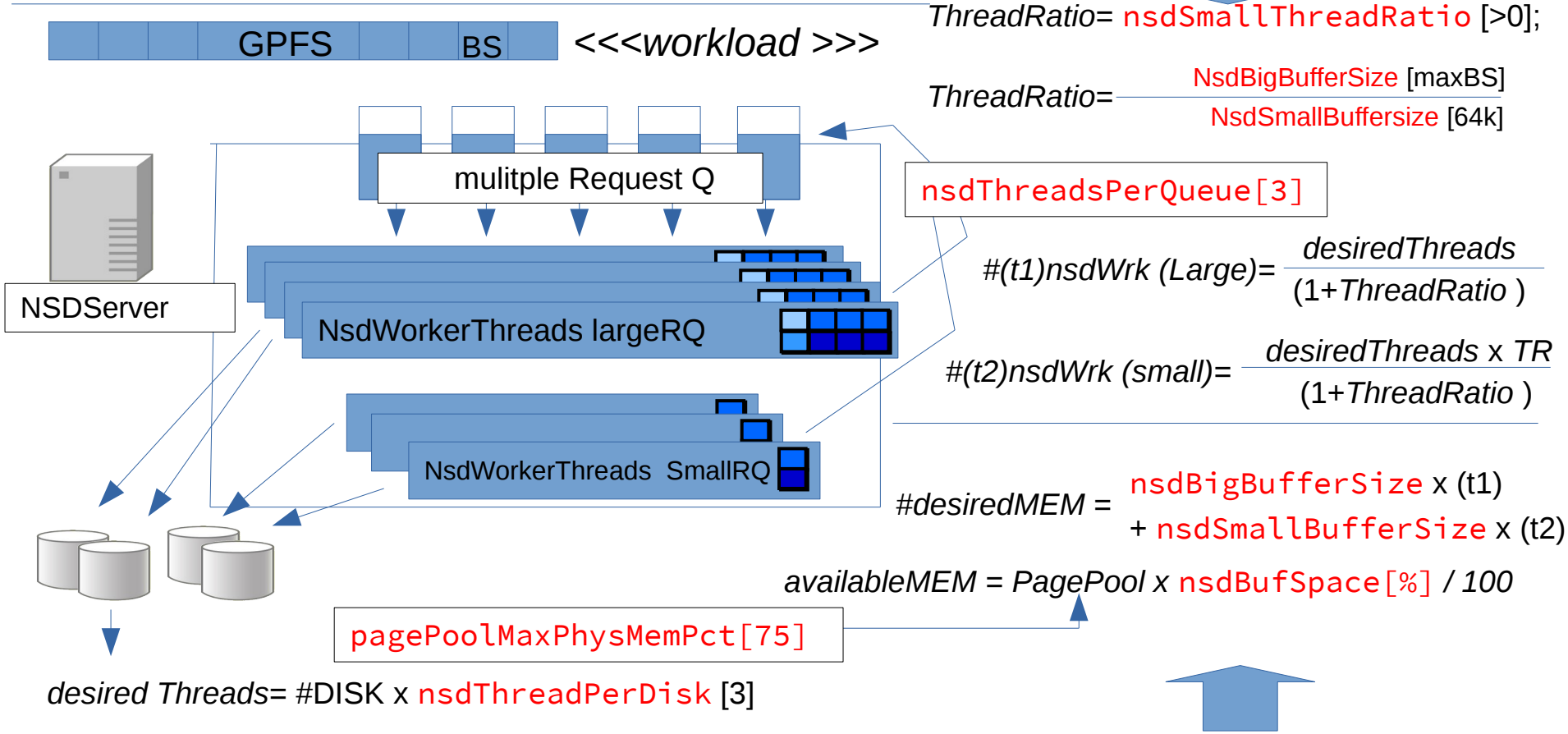
mmfsadm dump deadlock

```
[root@fscc-sr650-13 ~]#  mmfsadm dump deadlock
Waiting 367.6072 sec since 18:19:49, on node ece_13, thread 160772 MMFSADMDumpCmdThread: for mercy

Nodes the deadlock waiters depend on:
ece_13

Effective deadlock detection threshold on ece_13 is 300 seconds
Effective deadlock detection threshold on ece_13 is 180 seconds for short waiters

Cluster ece13-18.localnet.com is not overloaded. The overload index on ece_13 is 0.00029
```

NSD Server/clients Threads

Since GPFS 3.5 - NSD multi-queue

IBM Spectrum Scale



ThreadRatio= nsdSmallThreadRatio [>0];

$$\textit{ThreadRatio} = \frac{\text{NsdBigBufferSize [maxBS]}}{\text{NsdSmallBuffersize [64k]}}$$

nsdThreadsPerQueue[3]

$$\#(t1)\textit{nsdWrk (Large)} = \frac{\textit{desiredThreads}}{(1+\textit{ThreadRatio})}$$

$$\#(t2)\textit{nsdWrk (small)} = \frac{\textit{desiredThreads} \times \textit{TR}}{(1+\textit{ThreadRatio})}$$

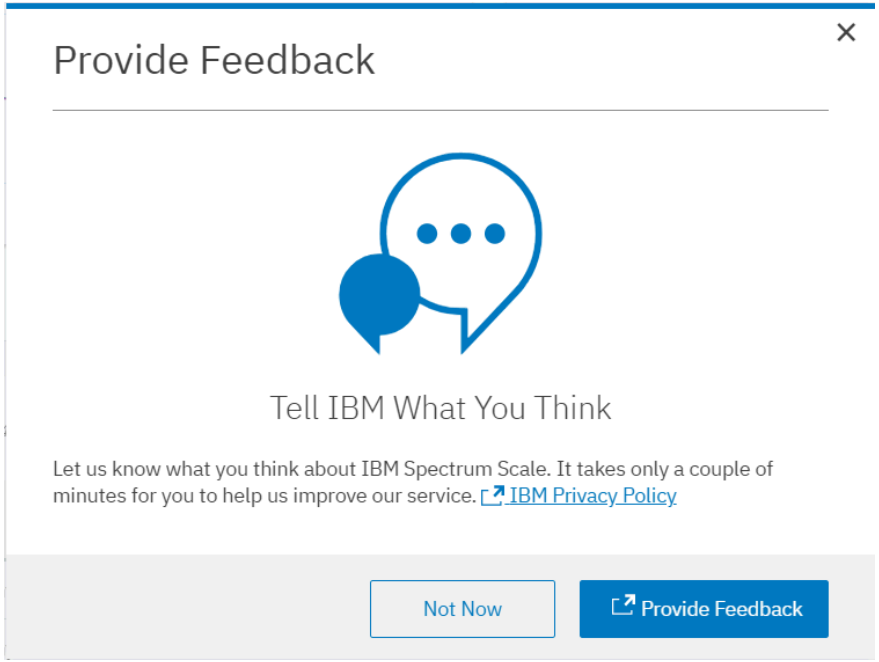
$$\#\textit{desiredMEM} = \text{nsdBigBufferSize} \times (t1) + \text{nsdSmallBufferSize} \times (t2)$$

availableMEM = PagePool x nsdBufSpace[%] */ 100*

pagePoolMaxPhysMemPct[75]

desired Threads= #DISK x nsdThreadPerDisk [3]

Thank you!



Please help us to improve Spectrum Scale with your feedback

- If you get a survey in email or a popup from the GUI, please respond
- We read every single reply