High-performance Profiling of High-performance File Systems

LUG 2018 / Chicago

Cameron Harr
(Lustre Ops Lead, LLNL)
Feiyi Wang
Sarp Oral
(ORNL/OLCF)

April 25, 2018

LLNL-PRES-749579
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

Abstract

Production parallel file systems have grown in size to the point where understanding what is on them has become a significant challenge. With capacities in the tens and even hundreds of petabytes, standard tools for gathering file characteristics struggle, or outright fail, to provide results in an acceptable time frame. Additionally, standard tools generally lack the abilities to provide a usable and comprehensive understanding of the entire file system. As a result, many sites lack a good understanding of file size distribution, file striping usage, directory allocation, or other file trends. A small number of third-party tools exist to help address characterization of the file system as policy engines, but their management-focused scope may preclude them from providing on-demand, comprehensive, profiles of a given file system.

To address the lack of a tool that can provide these profiles on a multi-petabyte scale, computer scientists at Oak Ridge National Laboratory's Oak Ridge Leadership Compututing Facility (OLCF) developed fprof, a high-performance, flexible, file system profiler. Inside Lawrence Livermore National Lab's Livermore Computing (LC) division, we are using fprof to profile a large number of production Lustre file systems on a monthly basis. fprof uses parallelization to walk the file system quickly, for instance, profiling a 5PB, 1.3 Billion file file system in a little more than 16 hours. fprof includes a wide list of both general and Lustre-specific properties a file system administrator or owner might be interested in, allowing one to spot interesting trends or catch alarming behavior before it becomes a problem.

This presentation addresses the implementation of fprof at LC and OLCF, along with its benefits and some challenges encountered along the way.

Lawrence Livermore National Lab

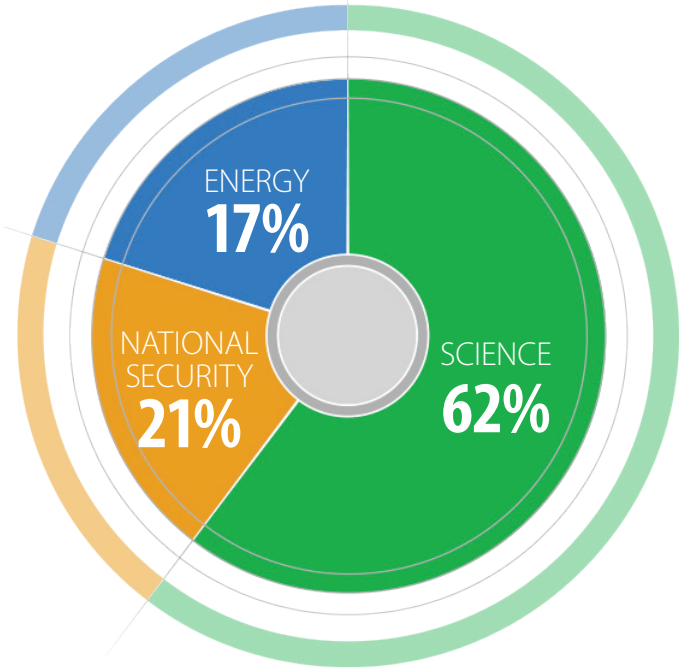
- US DoE / NNSA
  - Missions:
    - Biosecurity
    - Defense
    - Intelligence
    - Science
    - Counterterrorism
    - Energy
    - Nonproliferation
    - Weapons

Oak Ridge National Lab

- US DoE / Office of Science
  - Missions:
    - Discovery and Innovation
    - Clean Energy
    - Global Security
  - … And a lot more

OAK RIDGE National Laboratory | 75 YEARS

U.S. DEPARTMENT OF ENERGY | Office of Science

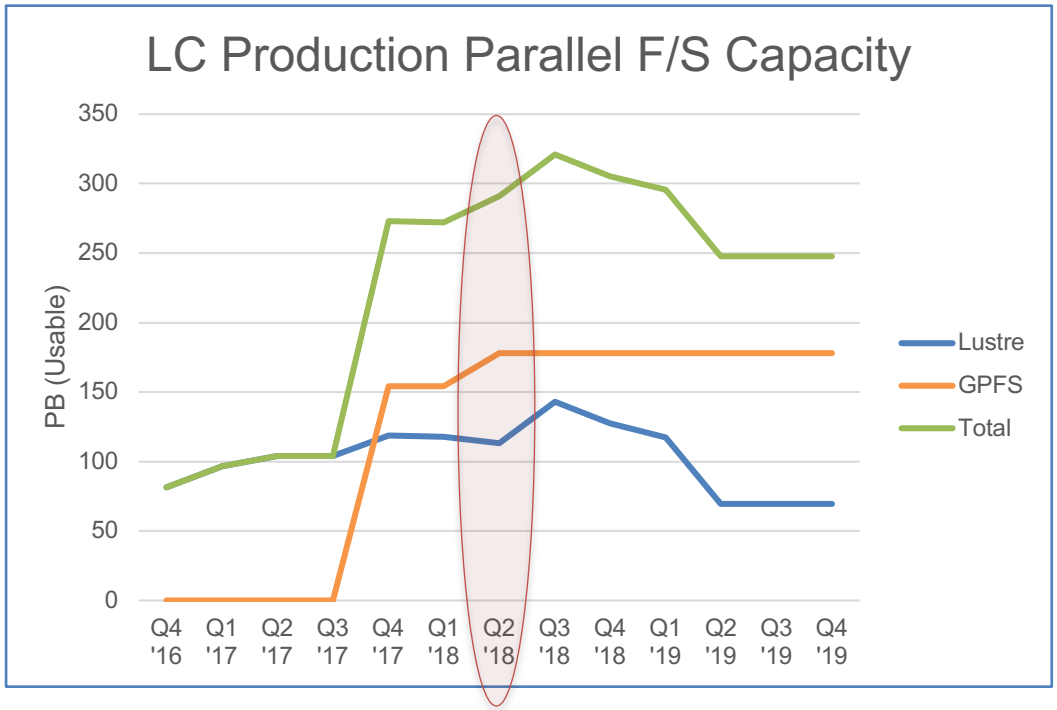


Livermore Computing (LC)

- Lots o' flops
  - Classified: ~150 PF
    - Sierra: ~125 PF
      - 150 PB GPFS
    - Sequoia: 20PF, #6 TOP500
  - Unclassified: ~14PF
- 4+ Data centers
  - Terascale Facility
    - 45MW now
    - 85MW by 2022

Lustre @ LC

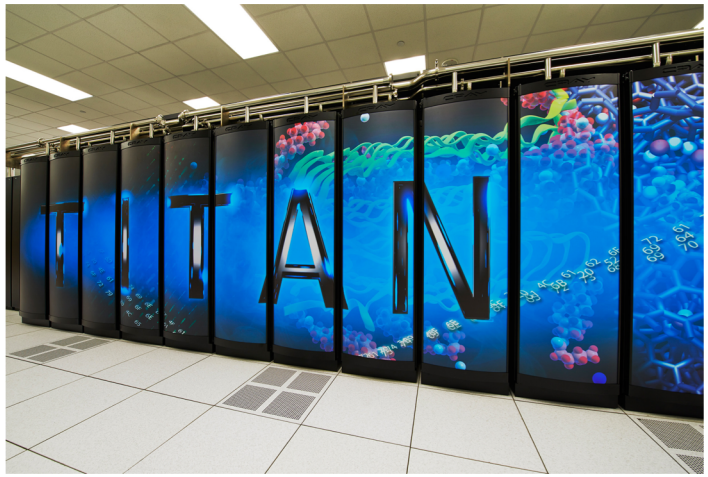
- Production Lustre
  - 12 production file systems
  - >118 PB (useable)
  - ~15B files
- Multi-generation
  - Lustre 2.5 (NetApp/Cray)
    - 1 MDS
  - Lustre 2.8 (RAID Inc.)
    - JBODs
    - 4-16 MDS
      - DNE v1



filesystem	Used Space in TB	Percent Full	Millions of files	Average File Size in KB
/p/lscratchd	4294	78%	1035	4456
/p/lscratche	3759	69%	1130	3573
/p/lscratchf	1679	77%	809	2229
/p/lscratchh	6807	41%	3363	2173
/p/lscratchrza	5734	69%	1207	5099
/p/lscratchrzb	461	42%	361	1371
/p/lscratchv	3659	56%	1638	2399

Oak Ridge Leadership Computing Facility (OLCF)

- World-class HPC resources
  - Summit: ~207PF
    - 104 -> 250 PB GPFS
  - Titan: 27PF, #5
- Lustre
  - 2x 14 PB file systems
    - Lustre 2.8 (w/o DNE)
    - 1 TB/s
    - > 20K clients
    - DDN back-end



Intro to Fprof

- What is fprof?
  - MPI-enabled file system profiler
    - File counts, sizes, types
    - Dir counts, sizes
    - Summary / Histogram
  - … and analyzer
    - Estimates usage of f/s migration using different block sizes
    - Looks at file striping usage
  - Not a policy engine
  - Compatible with POSIX f/s (not Lustre-specific)
  - Python-based
  - Component of *pcircle* project (https://github.com/olcf/pcircle)

Fprof Internals

- Work-stealing algorithm
  - Each worker maintains independent work queue
  - When worker's queue is empty, it requests work from neighbors
  - Neighbor processes split off and distribute work
- Distributed, self-balanced termination
  - Dijkstra-Scholten algorithm
- Error Recovery
  - Catch error codes
  - Timeout on unresponsive files

How to get/build fprof

- Quick Start
  1. $ brew install pkg-config libffi openmpi python
  2. $ pip2 install virtualenv
  3. $ virtualenv pcircle
  4. $ source ~/pcircle/bin/activate
  5. $ (pcircle) $ pip2 install git+https://github.com/olcf/pcircle@dev
- Alternatively…
  1. $ git clone https://github.com/olcf/pcircle.git pcircle/
  2. $ cd pcircle && python setup.py install [--prefix=<install path>]
     - Fix Python dependencies as necessary

Standard Fprof Attributes
(lscratchh – Jan 2018)

Attribute	Value
Directory count	13,421,535
Sym links count	824,161
Hard linked files	30,902
File count	2,499,341,395
Zero byte files	50,323,758
Sparse files	1,678,350,129
Skipped count	12,953
Total file size	4408.83 TiB
Avg file size	1.85 MiB
Max files within dir	165,187,097
Tree walk time	3d 6h 1m

How to run fprof

- $ ./pythont2/bin/fprof --help

```
usage: fprof [-h] [--version] [-v] [--loglevel LOGLEVEL] [-i INTERVAL]
             [--perfile] [--inodesz INODESZ] [--gpfs-block-alloc] [--dii]
             [--topn-files TOPN_FILES] [--perprocess] [--syslog] [--profdev]
             [--item ITEM] [--exclude FILE] [--lustre-stripe]
             [--stripe-threshold N] [--stripe-output] [--sparse] [--cpr]
             [--cpr-per-file] [--dirprof] [--dirbins INT [INT ...]]
             [--topn-dirs TOPN_DIRS]
             path [path ...]
```

- $ [srun -n | mpirun -np] fprof [OPTS] /<dir>/<to>/<profile>

Fprof Results (1)
(lscratchrza)

```
$ srun -N8 -n96 ~fprof/fprof/pythont2/bin/fprof --dirprof --dirbins 0 100 1000 10000 100000 1000000
10000000 100000000 --topn-dirs 5 --topn-files 5 /p/lscratchrza
Running Parameters:

          fprof version:      0.17-b1+10.gc823721
          Full rev id:        c823721f4dd42ec461386bcd878864963c18ce03
          Num of hosts:       8
          Num of processes:   96
          Syslog report:      no
          Dir bins:           [0, 100, 1000, 10000, 100000, 1000000, 10000000, 100000000]
          Stripe analysis:    no
          Root path:          ['/p/lscratchrza']

Start profiling ...
Scanned files: 86,328          Processing rate: 2,843 /s   HWM mem: 4.96 GiB       Work Queue: 953,963
Scanned files: 190,389         Processing rate: 3,444 /s   HWM mem: 4.97 GiB       Work Queue: 996,058
…
Scanned files: 1,087,144,010   Processing rate: 52,309/s   HWM mem: 11.48 GiB      Work Queue: 83,002
Scanned files: 1,088,583,492   Processing rate: 48,000/s   HWM mem: 11.48 GiB      Work Queue: 69,179
Scanned files: 1,090,350,484   Processing rate: 58,936/s   HWM mem: 11.48 GiB      Work Queue: 65,438

Fprof epilogue:
…
```

Fprof Results (2)
(lscratchrza)

```
Fprof epilogue:

          Directory count:         19,675,094
          Sym links count:         1,375,161
          Hard linked files:       160,682
          File count:              1,072,456,101
          Zero byte files:         663,099
          Sparse files:            360,343,869
          Skipped count:           108
          Total file size:         5141.07 TiB
          Avg file size:           5.03 MiB
          Max files within dir:    623,520
          Tree walk time:          8h 20m
          Fprof loads:             [13759878, 8841050, 13611364, 12080406, 12732872, 10577949,
11885219, 9652730, 12751017, 11889945, 12920296, 12979376, 13932984, 7282767, 8758004, 12349807,
11604780, 9589568, 8947956, 11137446, 12880504, 12901442, 9625123, 11398163, 11642491, 11375671,
11624517, 13412809, 8837763, 13207526, 12527818, 13475660, 12700919, 10582276, 11575621, 9943402,
12495729, 13039990, 11125767, 12116475, 10210026, 10347832, 10347747, 11871533, 11880984, 10931227,
12285024, 12771166, 8923104, 12091795, 10841860, 11129354, 3853038, 12917747, 10176320, 12801694,
10934851, 12770960, 9619496, 9027650, 10114950, 13286744, 13468219, 12695437, 10485745, 12159023,
13270771, 11482909, 11506223, 9715437, 11628267, 8405102, 10779071, 11517819, 10388690, 9815811,
10538035, 9508142, 11903220, 11947783, 11117320, 13962843, 11308282, 11919568, 9997751, 11099777,
9263740, 9120410, 13969619, 10429922, 12716073, 12710134, 11543161, 12050105, 14890467, 11906137]

Fileset Histogram
…
```

Fprof Results (3)
(lscratchrza)

Fileset Histogram

Buckets	Num of Files	Size	%(Files)	%(Size)
<= 0.00 KiB	663,099	0.00 KiB	0.06%	0.00%
<= 4.00 KiB	24,632,403	20.32 GiB	2.30%	0.00%
<= 8.00 KiB	9,666,431	59.54 GiB	0.90%	0.00%
<= 16.00 KiB	7,430,425	81.06 GiB	0.69%	0.00%
<= 32.00 KiB	32,362,013	805.73 GiB	3.02%	0.02%
<= 64.00 KiB	38,944,383	1.78 TiB	3.63%	0.03%
<= 128.00 KiB	48,133,956	4.15 TiB	4.49%	0.08%
<= 256.00 KiB	73,437,133	12.48 TiB	6.85%	0.24%
<= 512.00 KiB	133,707,067	43.50 TiB	12.47%	0.85%
<= 1.00 MiB	204,968,577	150.03 TiB	19.11%	2.92%
<= 2.00 MiB	173,513,687	228.27 TiB	16.18%	4.44%
<= 4.00 MiB	117,150,200	319.64 TiB	10.92%	6.22%
<= 16.00 MiB	157,029,129	1174.44 TiB	14.64%	22.84%
<= 32.00 MiB	37,624,205	808.27 TiB	3.51%	15.72%
<= 64.00 MiB	6,418,957	253.47 TiB	0.60%	4.93%
<= 128.00 MiB	3,688,231	313.96 TiB	0.34%	6.11%
<= 256.00 MiB	1,679,870	296.11 TiB	0.16%	5.76%
<= 512.00 MiB	727,053	252.56 TiB	0.07%	4.91%
<= 1.00 GiB	372,766	265.87 TiB	0.03%	5.17%
<= 4.00 GiB	231,478	480.76 TiB	0.02%	9.35%
<= 64.00 GiB	74,957	499.93 TiB	0.01%	9.72%
<= 128.00 GiB	62	5.08 TiB	0.00%	0.10%
<= 256.00 GiB	4	792.09 GiB	0.00%	0.02%
<= 512.00 GiB	4	1.23 TiB	0.00%	0.02%
<= 1.00 TiB	4	2.85 TiB	0.00%	0.06%
<= 4.00 TiB	5	15.10 TiB	0.00%	0.29%
> 4.00 TiB	2	9.86 TiB	0.00%	0.19%

Fprof Results (4)
(lscratchrza)

Directory Histogram

Buckets	Num of Entries	%(Entries)
<= 0	4,294,001	21.82%
<= 100	13,459,697	68.41%
<= 1,000	1,794,969	9.12%
<= 10,000	123,342	0.63%
<= 100,000	2,888	0.01%
<= 1,000,000	197	0.00%
<= 10,000,000	0	0.00%
<= 100,000,000	0	0.00%

Top N File Report:

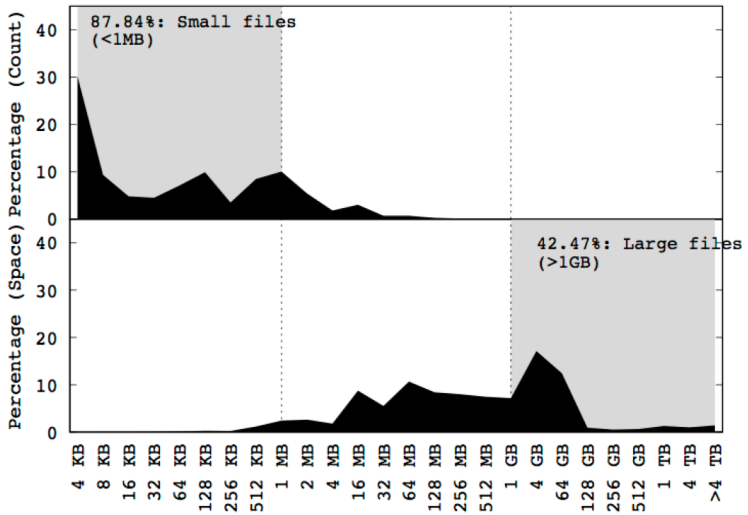
1: /p/lscratchrza/user1/save.scratchzb.mar29/Mar/foo-2014-09-15.tar (5.75 TiB)
2: /p/lscratchrza/user1/save.scratchzb.mar29/Mar/foo/ToCab.tar (4.11 TiB)
3: /p/lscratchrza/user1/save.scratchzb.mar29/Mar/foo-2014-09-15.tar (3.86 TiB)
4: /p/lscratchrza/user1/save.scratchzb.mar29/Mar/foo-2015-01-09.tar (3.54 TiB)
5: /p/lscratchrza/user2/he_modeling.tgz (3.36 TiB)

Top N Directory Report:

1: /p/lscratchrza/user10/Ensemble/1/VIZ/json (623,520 items)
2: /p/lscratchrza/user11/Dmp (576,000 items)
3: /p/lscratchrza/user5/contrast (489,393 items)
4: /p/lscratchrza/user10/Ensemble/2/VIZ2/json (460,512 items)
5: /p/lscratchrza/user7/noread/Dmp (453,312 items)

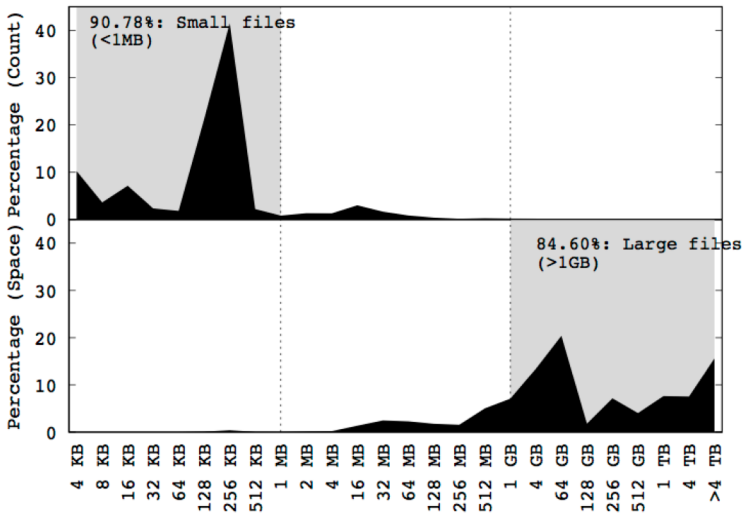
Interesting Finds

- LC (lscratche)



- 88% files <= 1MB
- Files >=1GB == 42% capacity

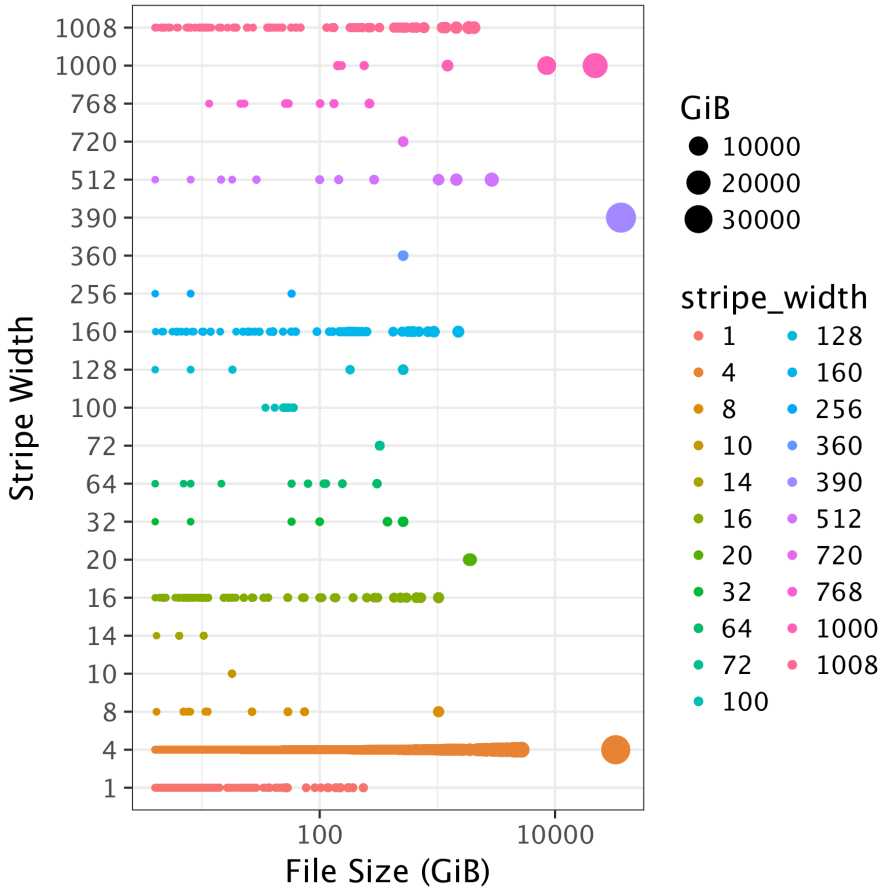
- OLCF (Atlas 1&2)



- 90% files <= 1MB
- Files >= 1GB == 85% capacity
- File Striping >= 4GB
  - ~97% keep default stripe width

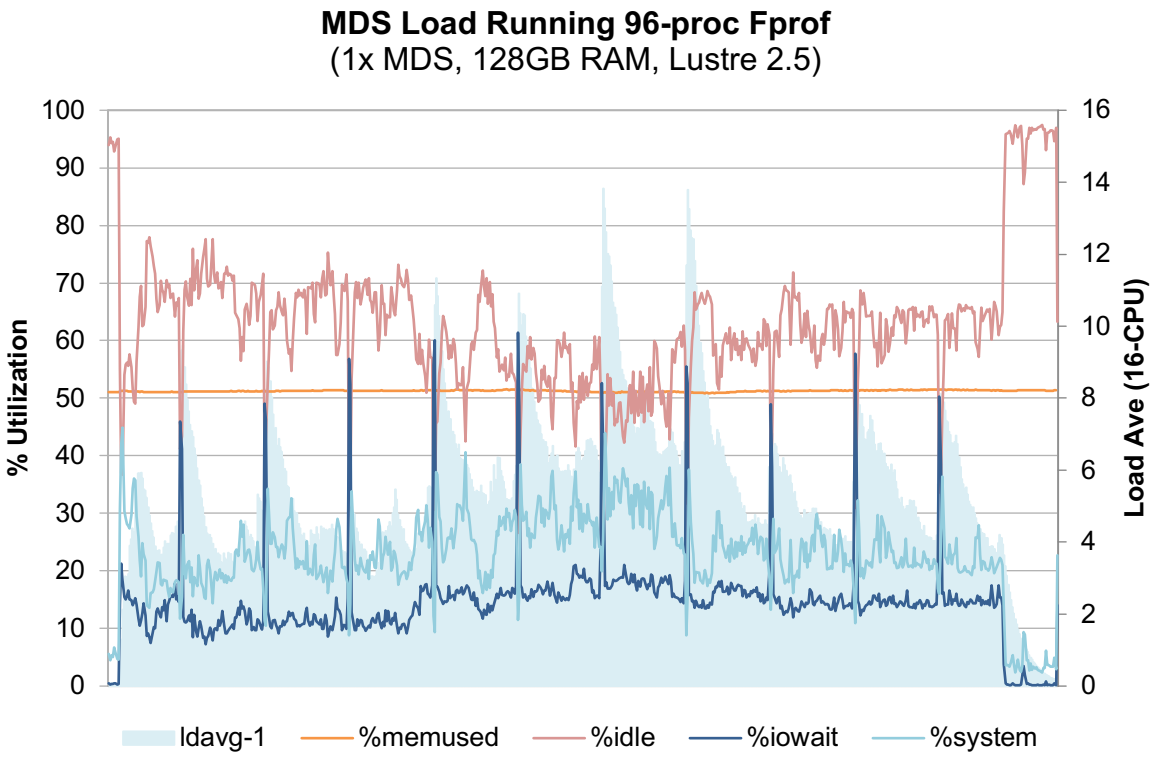
Fprof Stripe Analysis
(Atlas 1&2)

- 96.8% of files >= 4GB use default count=4
- 21 different stripe counts used
- No correlation between file size and stripe width
- Some file explicitly set lower
  - Count=1
- PFL, anyone?



Fprof Performance

- Goals:
  - Run monthly
  - Low impact
    - F/S performance
    - Batch node availability
- MDS load
  - No mem hit
  - Moderate CPU hit
  - IOP rate below peak
  - MDS not bottleneck



Challenges

- Problems encountered
  - Root access for threads
    - Sudo worked, then didn't (suspected MPI limitation)
    - Root-jobs disabled
  - Crashing jobs
    - Limit lru_size right before job, then free it back up:
      - lctl set_param –n ldlm.namespaces.*.lru_size=12000
      - lctl set_param –n ldlm.namespaces.*.lru_size=0
  - OS differences (RHEL 6 and RHEL 7 derivatives)
    - Separate pcircle and python dirs
  - On one file system (lscratchh), correlation between Spectre/Meltdown patches and slow fprof performance
    - Still need to track down

Future work

- Better detect sparse vs. compressed files
  - Using stat() (st_size – st_blocks*512)
  - Better to detect holes with SEEK_HOLE
    - Need to investigate Lustre compatibility (LU-10810?)
- Python v3 compatibility
- Better/updated documentation
- Wider adoption and community input

Thank you!