DE LA RECHERCHE À L'INDUSTRIE

cea

www.cea.fr

Lecture on Parallel Filesystems

Lustre (3/5)

Jacques-Charles Lafoucriere

ENSIIE| 2018

DNE

What is DNE?

Distributed Namespace Environment

- Multiple MDT for a single FS
- Multiple MDS can serve a single FS
- FS root is always on MDT0

- Namespace can be distributed over MDT on sub-tree basis
  - Each MDT host a full sub-tree

- A directory can be distributed over multiple MDT
  - Such directory is named a striped directory

DNE Advantages

Multiple MDT

- More IOps globally
- Allow dedicating MDT to some subtree
  - Localize load
  - Fault isolation

Striped directory

- Larger directory
- More IOps in a single directory

How to Create a Lustre With Multiple MDT?

Create a new MDT

```
# mkfs -t lustre --fsname=test --mgsnode=MGS_NID --mdt --index=1 /dev/blockdevice8
```

Start it and add it to the configuration

```
# mount -t lustre /dev/blockdevice8 /mnt/mdt1
```


Creating a Sub-directory on a given MDT

Create a new directory

```
$ lfs mkdir -i 1 dir1
$ lfs setdirstripe -i 1 dir1
```

Check a directory stripe

```
$ lfs getdirstripe dir1
lmv_stripe_count: 0 lmv_stripe_offset: 1 lmv_hash_type: none
```

Creating a Sub-directory on a given MDT (cont.)

Change an existing directory

```
No way
```

Check how files are created

```
$ lfs getstripe -m FILE
1
```

How can I find information on mdt striping rules/result?

```
$ lfs find DIR --mdt-index 1
```


Creating a Striped Directory

Create Directory

```
$ lfs mkdir -c 2 sdir
```

Check the result

```
$ lfs getdirstripe sdir
lmv_stripe_count: 2 lmv_stripe_offset: 0 lmv_hash_type: fnv_1a_64
mdtidx           FID[seq:oid:ver]
     0           [0x200000400:0x2:0x0]
     1           [0x300000401:0x2:0x0]
```

Using a Striped Directory

After creating 10 files

```
$ lfs find .  --mdt-index 0
.
./file7
./file1
./file9
./file5
./file3
$ lfs find .  --mdt-index 1
./file8
./file6
./file0
./file4
./file2
```

```
$ lfs getstripe --mdt-index file*
1
0
1
0
1
0
1
0
1
0
```


DoM

DoM: Data on MDT

Objective

- Improves small file IO

How

- Place small files directly on the MDT

Positive Consequence

- Avoid OST being affected by small random IO

DoM Implementation

Based on PFL

- 1st component is on MDT
- Others are placed on OST's if needed
  - Case if "small file" growths too much
  - OST's component are created on demand

DoM File Creation

Create PFL file

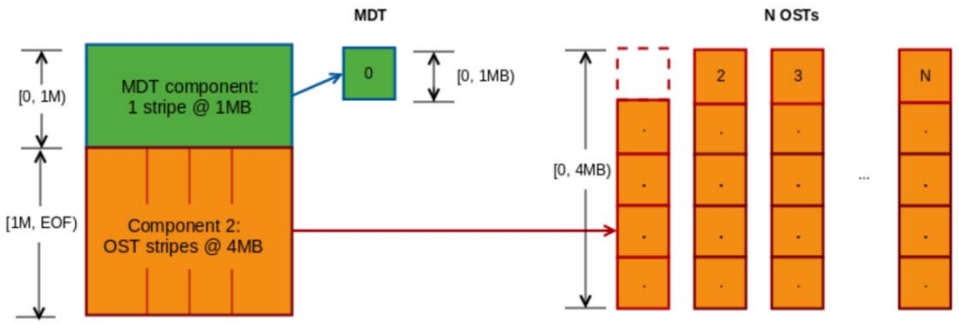
```
$ lfs setstripe -E 1M -L mdt -E -1 fom0
```

Get result

```
$ lfs getstripe fom0
```


DoM File Creation: a more complex example

Create PFL file




```
$ lfs setstripe -E 1M -L mdt -E EOF -S 4M -c -1 fom2
```

DoM File Creation: a more complex example (cont.)

Get layout

```
$ lfs getstripe fom2
fom2
  lcm_layout_gen:    2
  lcm_mirror_count:  1
  lcm_entry_count:   2
    lcme_id:             1
    lcme_mirror_id:      0
    lcme_flags:          init
    lcme_extent.e_start: 0
    lcme_extent.e_end:   1048576
      lmm_stripe_count:  0
      lmm_stripe_size:   1048576
      lmm_pattern:       mdt
      lmm_layout_gen:    0
      lmm_stripe_offset: 0
```

```
lcme_id:             2
lcme_mirror_id:      0
lcme_flags:          0
lcme_extent.e_start: 1048576
lcme_extent.e_end:   EOF
  lmm_stripe_count:  -1
  lmm_stripe_size:   4194304
  lmm_pattern:       raid0
  lmm_layout_gen:    0
  lmm_stripe_offset: -1
```


DoM Limitations

To avoid filling MDT

- Max size of a DoM component is limited by lfs setstripe 1GB
- MDT also limits to a tunable: dom_stripesize

```
# lctl set_param -n lod.MDT0000.dom_stripesize=2M
# lctl conf_param test-MDT0000.lod.dom_stripesize=2M
```

- Setting dom_stripesize to 0 will forbid use of this MDT for DoM
- To get some parameter value

```
# lctl get_param lod.test-MDT0000*.dom_stripesize
lod.test-MDT0000-mdtlov.dom_stripesize=2097152
# cat /proc/fs/lustre/lod/test-MDT0000-mdtlov/dom_stripesize
2097152
```

File Level Redundancy

What is a File Level Redundancy?

Objective

- Increase hardware fault tolerance

How

- Mirror File Data on multiple OST's

Positive Consequence

- Double aggregate parallel read performance of a single file

FLR File Creation

Create a Mirrored file

```
$ lfs mirror create -N -o 1 -N -o 2 mir2
```


FLR File Creation (Cont.)

Check result

```
$ lfs getstripe mir2
mir2
  lcm_layout_gen:    2
  lcm_mirror_count:  2
  lcm_entry_count:   2
    lcme_id:             65537
    lcme_mirror_id:      1
    lcme_flags:          init
    lcme_extent.e_start: 0
    lcme_extent.e_end:   EOF
      lmm_stripe_count:  1
      lmm_stripe_size:   1048576
      lmm_pattern:       raid0
      lmm_layout_gen:    0
      lmm_stripe_offset: 1
      lmm_objects:
      - 0: { l_ost_idx: 1, l_fid:
[0x100010000:0x3:0x0] }
    lcme_id:             131074
    lcme_mirror_id:      2
    lcme_flags:          init
    lcme_extent.e_start: 0
    lcme_extent.e_end:   EOF
      lmm_stripe_count:  1
      lmm_stripe_size:   1048576
      lmm_pattern:       raid0
      lmm_layout_gen:    0
      lmm_stripe_offset: 2
      lmm_objects:
      - 0: { l_ost_idx: 2, l_fid:
[0x100020000:0x2:0x0] }
```

FLR Tips and Tricks

To control redundancy rules

- Associate mirrors with pools

Mirrors do not need to follow same striping

- 1st mirror striped over 2 OSTs HDD
- 2nd mirror striped over 4 OST HDD
- 3rd mirror striped over 1 OST SSD

Mirror states

- Init: component is allocated (has objects)
- Stale: component does not have up-to-date data
  - use lfs mirror resync to force sync
- Prefer: preferred component of RD/RW

FLR Tips and Tricks (cont.)

Mirror can be added after file creation

Mirror sync is today delayed

- Should be force by lfs mirror resync

Mirror can be detached from a file

- Detached layout can to put in a new file or destroyed

Mirror states can be checked

- $ lfs mirror verify

Mirrored file can be found

- $ lfs find

Lustre Parameters

Lustre Parameters Setting

How to get/set Lustre tuneable

- 2 interfaces
  - lctl set_param/get_param/conf_param
  - through /proc

```
# lctl set_params -n lov.test-MDT0000*.stripesize=2M
# echo 2M > /proc/fs/lustre/lov/test-MDT0000-mdtlov/stripesize

# lctl get_param lov.test-MDT0000*.stripesize
lov.test-MDT0000-mdtlov.stripesize=2097152
# cat /proc/fs/lustre/lov/test-MDT0000-mdtlov/stripesize
1048576
# lctl conf_param test-MDT0000.lov.stripesize=2M
```

Thank you for your attention

ENSIIE | 2018

Commissariat à l'énergie atomique et aux énergies alternatives
Centre DAM-Ile de France | 91297 Bruyères-le-Châtel Cedex
T. +33 (0)1 69 26 40 00 | F. +33 (0)1 69 26 70 86
Etablissement public à caractère industriel et commercial | RCS Paris B 775 685 019

Direction des applications militaires
Département sciences de la simulation et de l'information
Service informatique scientifique et réseaux

Lustre Glossary

DNE - Distributed Namespace Environment - feature to aggregate multiple MDTs (possibly on many MDS's) into a single filesystem namespace

IDIF - OST object ID In FID - specific FID range reserved for compatibility with pre-DNE OST objects

IGIF - Inode and Generation In FID - specific FID range reserved for compatibility from Lustre 1.x MDT inode objects

FID - File IDentifier - unique 128-bit identifier for every object within a single filesystem.

LMV - Logical Metadata Volume - client software layer that handles client (llite) access to multiple MDTs

LOD - Logical Object Device - MDS software layer that handles access to multiple MDTs and multiple OSTs

LOV - Logical Object Volume - client software layer that handles client (llite) access to multiple OSTs

MDC - MetaData Client - client software layer that interfaces to the MDS

MDD - Metadata Device Driver - MDS software layer that understands POSIX semantics for file access

MDS - MetaData Server - software service that manages access to filesystem namespace (inodes, paths, permission) requests from the client.

MDT - MetaData Target - storage device that holds the filesystem metadata (attributes, inodes, directories, xattrs, etc)

MGS - Management Server - service that helps clients and servers with configuration

MGT - Management Target - storage device that holds the configuration logs

OFD - Object Filter Device - OSS software layer that handles file IO

OSC - Object Storage Client - client software layer that interfaces to the OST

OSD - Object Storage Device - server software layer that abstracts MDD and OFD access to underlying disk filesystems like ldiskfs and ZFS

OSP - Object Storage Proxy - server software layer that interfaces from one MDS to the OSD on another MDS or another OSS

OSS - Object Storage Server - software service that manages access to filesystem data (read, write, truncate, etc)

OST - Object Storage Target - storage device that holds the filesystem data (regular data files, not directories, xattrs, or other metadata)

SFP 2018 Parallel Filesystems / Lustre | PAGE 26