Self-Extending Layout (SEL) Operations Manual, v0.1

Author: Vitaly Fertman

The Lustre Self-Extending Layout (SEL) feature is an extension of the Progressive File Layout which allows MDS to change the defined PFL layout dynamically. MDS is monitoring the used space on OSTs and SEL lets it to swap the current file OSTs when they are low on space, this allows to avoid ENOSPC problems for SEL files when applications are writing to them.

SEL creates a file layout where each declared component is split on disk to 2 parts, in fact individual PFL components: a small "extendable" component and a large "extension" component. The extension (or SEL) component is a new component type which is always unassigned, and when a write reaches this unassigned space (and the client calls MDS to have it instantiated) MDS makes a decision if to grant some space to the extendable component (the granted region moves from the head of the extension component to the tail of the extendable component) and therefore allow the file to continue on the same OSTs; or (in case of low on space of at least one of the current OSTs) to modify the layout to switch to a new component on new OSTs. In particular, it lets IO automatically to spill over to a large HDD OST pool while using a small SSD OST pool.

The default extension policy modifies the layout in the following ways:

1. Extension: continue on the same OSTs – used when low on space condition does not occur on any of the current OSTs; a particular extent is granted to the extendable component;
2. Spill over: switch to next component OSTs – it is used only for not the last component when at least 1 of the current OSTs is low on space; the start of the next component is moved to the start of the SEL component, which is removed in its turn.
3. Repeating: switch to other OSTs within the same pool – it is used only for the last component when at least 1 of the current OSTs is low on space; a new component with the same layout is created but instantiated on different OSTs from the pool;
4. Forced extension: continue with the current OSTs despite the low on space condition – it is used only for the last component and if the component has just passed the low on space condition for the instantiation but immediately fails this condition for the extension within the same operation – spillover is impossible and there is no sense in the repeating.

SEL does not resolve all ENOSPC issues, in particular, it does not apply to non-SEL files or pre-existing files which keep writing to the same OSTs, layout is extended in some chunks what means it is still possible to run out of space while using already granted layout. But if applied as a default layout policy with appropriate settings, it should help explicitly avoid ENOSPC issues.

New commands are introduced to the Lustre tools to operate SEL files, below are the new commands and examples for the lfs setstripe, lfs find, lfs getstripe

Note
Using SEL files does not require clients to understand the SEL format of already created files, only the MDS supposed is needed which is introduced in Lustre 2.13. However, old clients will have some limitations as the Lustre tools will not support it.

1. lfs setstripe

lfs setstripe command is extended to create SEL files.

1.1. Create a SEL file

lfs setstripe -E end1 [STRIPE_OPTIONS] ... <directory|file>

STRIPE OPTIONS:
--extension-size, --ext-size, -z <ext_size>

While declaring another component, the stripe option "—extension-size" turns it to a pair of components: extendable and extension ones.

Example 1.

```
# ./lustre/utils/lfs setstripe -E 1G -z 64M -E -1 -z 256M /mnt/lustre/file
```

This command creates 2 pairs of extendable and extension components:
[0, 64M], SEL [64M, 1G], [1G, 1G], SEL [1G, EOF].
As the 1st component of a PFL file is instantiated, it is extended to the extension size, whereas the 3rd component is left 0-length.

```
# ./lustre/utils/lfs getstripe /mnt/lustre/file
/mnt/lustre/file
  lcm_layout_gen:    4
  lcm_mirror_count:  1
  lcm_entry_count:   4
    lcme_id:             1
    lcme_mirror_id:      0
    lcme_flags:          init
    lcme_extent.e_start: 0
    lcme_extent.e_end:   67108864
      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: 0, l_fid: [0x100000000:0x6:0x0] }

    lcme_id:             2
    lcme_mirror_id:      0
    lcme_flags:          extension
    lcme_extent.e_start: 67108864
    lcme_extent.e_end:   1073741824
      lmm_stripe_count:  0
      lmm_extension_size: 67108864
      lmm_pattern:       raid0
      lmm_layout_gen:    0
      lmm_stripe_offset: -1

    lcme_id:             3
```

```
  lcme_mirror_id:     0
  lcme_flags:         0
  lcme_extent.e_start: 1073741824
  lcme_extent.e_end:   1073741824
    lmm_stripe_count:  1
    lmm_stripe_size:   1048576
    lmm_pattern:       raid0
    lmm_layout_gen:    0
    lmm_stripe_offset: -1

  lcme_id:            4
  lcme_mirror_id:     0
  lcme_flags:         extension
  lcme_extent.e_start: 1073741824
  lcme_extent.e_end:   EOF
    lmm_stripe_count:  0
    lmm_extension_size: 268435456
    lmm_pattern:       raid0
    lmm_layout_gen:    0
    lmm_stripe_offset: -1
```

As you can see the SEL components are marked by the “extension” flag and have lmm_extension_size field with the specified extension size.

Set default SEL layout to an existing directory

Similar to PFL, it is possible to define a default SEL layout for a directory. After that, all the files created will inherit this layout by default.

Extension, spillover, repeating.

Suppose a SEL file is created:
```
# ./lustre/utils/lfs setstripe -E 1G -z 64M -E -1 -z 256M /mnt/lustre/file
```

i.e. with the following components: [0, 64M], SEL [64M, 1G], [1G, 1G], SEL [1G, EOF].

Each time an extension happens it is done on the specified extension size, i.e. for 1st and 2nd components on 64M and 256M correspondingly. Suppose there is an IO beyond the end of the 1st component (64M), the 1st component becomes [0, 128M], [0, 192M], etc on each extension; the SEL component is shrunk appropriately: [128M, 1G], [192M,1G], etc.

However, in case OST0 is low on space, a spillover happens, when the full region of the SEL component is added to the next component, e.g. after 2 successful extensions and spillover it looks like:

```
# ./lustre/utils/lfs getstripe  /mnt/lustre/file
/mnt/lustre/file
  lcm_layout_gen:    6
  lcm_mirror_count:  1
```

```
lcm_entry_count:   3
  lcme_id:           1
  lcme_mirror_id:    0
  lcme_flags:        init
  lcme_extent.e_start: 0
  lcme_extent.e_end:   201326592
    lmm_stripe_count:  1
    lmm_stripe_size:   1048576
    lmm_pattern:       raid0
    lmm_layout_gen:    0
    lmm_stripe_offset: 0
    lmm_objects:
    - 0: { l_ost_idx: 0, l_fid: [0x100000000:0x3:0x0] }

  lcme_id:           3
  lcme_mirror_id:    0
  lcme_flags:        init
  lcme_extent.e_start: 201326592
  lcme_extent.e_end:   469762048
    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:           4
  lcme_mirror_id:    0
  lcme_flags:        extension
  lcme_extent.e_start: 469762048
  lcme_extent.e_end:   EOF
    lmm_stripe_count:  0
    lmm_extension_size: 268435456
    lmm_pattern:       raid0
    lmm_layout_gen:    0
    lmm_stripe_offset: -1
```

As you can see, despite the fact the 3rd component was [1G, 1G] originally, while it is not instantiated, instead of getting extended backward, it is moved backward to the start of the previous SEL component (192M) and extended on its extension size (256M) from that position, thus it becomes [192M, 448M].

Now, suppose OST0 got enough free space back but OST1 is low on space, the following write to the last SEL component leads to a new component allocation before the SEL component, which repeats the previous component layout but is instantiated on free OSTs:

```
# ./lustre/utils/lfs getstripe  /mnt/lustre/file
/mnt/lustre/file
```

```
lcm_layout_gen:    8
lcm_mirror_count:  1
lcm_entry_count:   4
  lcme_id:             1
  lcme_mirror_id:      0
  lcme_flags:          init
  lcme_extent.e_start: 0
  lcme_extent.e_end:   201326592
    lmm_stripe_count:  1
    lmm_stripe_size:   1048576
    lmm_pattern:       raid0
    lmm_layout_gen:    0
    lmm_stripe_offset: 0
    lmm_objects:
    - 0: { l_ost_idx: 0, l_fid: [0x100000000:0x3:0x0] }

  lcme_id:             3
  lcme_mirror_id:      0
  lcme_flags:          init
  lcme_extent.e_start: 201326592
  lcme_extent.e_end:   469762048
    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:             7
  lcme_mirror_id:      0
  lcme_flags:          init
  lcme_extent.e_start: 469762048
  lcme_extent.e_end:   738197504
    lmm_stripe_count:  1
    lmm_stripe_size:   1048576
    lmm_pattern:       raid0
    lmm_layout_gen:    65535
    lmm_stripe_offset: 0
    lmm_objects:
    - 0: { l_ost_idx: 0, l_fid: [0x100000000:0x4:0x0] }

  lcme_id:             4
  lcme_mirror_id:      0
  lcme_flags:          extension
  lcme_extent.e_start: 738197504
  lcme_extent.e_end:   EOF
    lmm_stripe_count:  0
    lmm_extension_size: 268435456
```

```
lmm_pattern:       raid0
lmm_layout_gen:    0
lmm_stripe_offset: -1
```

lfs getstripe

lfs getstripe is extended to operate SEL files. When getting the striping of the file, extension components are marked by the “extension” flag and have lmm_extension_size field with the extension size specified at the time of the “lfs setstripe”.

--extension-size|--ext-size|-z

A new option is added to print the extension size in bytes. For composite files this is the extension size of the first extension component. If a particular component is identified by other options (--component-id, --component-start, etc), this component extension size is printed.

Example

```
# ./lustre/utils/lfs setstripe -E 1G -z 64M -E -1 -z 256M /mnt/lustre/file
# ./lustre/utils/lfs getstripe -z -I2 /mnt/lustre/file
67108864
```

lfs find

lfs find is extended to operate on SEL files. It can be used to search for the files that match the given SEL extension size.

--extension-size|--ext-size|-z [+-]ext-size[KMG]