Subject: [PATCH] seq_file: more atomicity in traverse()
Posted by Alexey Dobriyan on Fri, 01 Jun 2007 12:55:19 GMT
View Forum Message <> Reply to Message

Original problem: in some circumstances seq_file interface can present
infinite proc file to the following script when normally said proc file
is finite:

```
 while read line; do
  [do something with $line]
 done </proc/$FILE
```

bash, to implement such loop does essentially

```
 read(0, buf, 128);
 [find \n]
 lseek(0, -difference, SEEK_CUR);
```

Consider, proc file prints list of objects each of them consists of many
lines, each line is shorter than 128 bytes.

Two objects in list, with ->index'es being 0 and 1. Current one is 1,
as bash prints second object line by line.

Imagine first object being removed right before lseek().
traverse() will be called, because there is negative offset.
traverse() will reset ->index to 0 (!).
traverse() will call ->next() and get NULL in any usual iterate-over-list
code using list_for_each_entry_continue() and such. There is one object in
list now after all...
traverse() will return 0, lseek() will update file position and pretend
everything is OK.

So, what we have now: ->f_pos points to place where second object will be
printed, but ->index is 0. seq_read instead() of returning EOF, will start
printing first line of first object every time it's called, until enough
objects are added to ->f_pos return in bounds.

Fix is to update ->index only after we're sure we saw enough objects down
the road.

Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>

```
 fs/seq_file.c |   16 ++++++++++------
 1 file changed, 10 insertions(+), 6 deletions(-)
```

--- a/fs/seq_file.c

```
+++ b/fs/seq_file.c
@@ -177,21 +177,23 @@ EXPORT_SYMBOL(seq_read);

 static int traverse(struct seq_file *m, loff_t offset)
 {
- loff_t pos = 0;
+ loff_t pos = 0, index;
  int error = 0;
  void *p;

  m->version = 0;
- m->index = 0;
+ index = 0;
  m->count = m->from = 0;
- if (!offset)
+ if (!offset) {
+  m->index = index;
   return 0;
+ }
  if (!m->buf) {
   m->buf = kmalloc(m->size = PAGE_SIZE, GFP_KERNEL);
   if (!m->buf)
    return -ENOMEM;
  }
- p = m->op->start(m, &m->index);
+ p = m->op->start(m, &index);
  while (p) {
   error = PTR_ERR(p);
   if (IS_ERR(p))
@@ -204,15 +206,17 @@ static int traverse(struct seq_file *m, loff_t offset)
   if (pos + m->count > offset) {
    m->from = offset - pos;
    m->count -= m->from;
+   m->index = index;
    break;
   }
   pos += m->count;
   m->count = 0;
   if (pos == offset) {
-   m->index++;
+   index++;
+   m->index = index;
    break;
   }
-  p = m->op->next(m, p, &m->index);
+  p = m->op->next(m, p, &index);
  }
  m->op->stop(m, p);
```

return error;