

Re: 2.6.2-rc2 nfsd+xfss spins in i_size_read()

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-01/7762.html>

From: Miquel van Smoorenburg (miquels_at_cistron.nl)

Date: 01/30/04

Date: Fri, 30 Jan 2004 21:21:55 +0100

To: Andrew Morton <akpm@osdl.org>

On Thu, 29 Jan 2004 07:25:21, Andrew Morton wrote:

> "Miquel van Smoorenburg" <miquels@cistron.nl> wrote:

>>

>> I have a Linux 2.6.2-rc2 NFS file server and another similar

>> box as client. Kernel is compiled for SMP (hyperthreading).

>> In a few seconds, the server locks up. It spins in

>> generic_fillattr(), apparently in the i_size_read() inline function.

>> Server responds to pings and sysrq, but nothing else.

>

> Is the EIP _always_ inside generic_fillattr()?

>

> If so then yes, your analysis look right. I'd say that the inode has been

> corrupted and the seqcount counter has assumed a non-even value. That

> will cause i_size_read() to lock up.

I added some extra code to i_size_read() and i_size_write(). First, some debugging code:

```
--- fs.h.orig 2004-01-30 21:10:28.000000000 +0100
```

```
+++ fs.h.v1 2004-01-30 21:11:19.000000000 +0100
```

```
@@ -425,6 +425,7 @@
```

```
    } u;
```

```
#ifdef __NEED_I_SIZE_ORDERED
```

```
    seqcount_t i_size_seqcount;
```

```
+ pid_t seq_pid; /* XXX */
```

```
#endif
```

```
};
```

```
@@ -450,6 +451,12 @@
```

```
    do {
```

```
        seq = read_seqcount_begin(&inode->i_size_seqcount);
```

```
        i_size = inode->i_size;
```

```
+#if 1 /* XXX HACK */
```

```
+ if ((++count & 65535) == 0) {
```

```
+ printk("i_size_read() seems to be looping - pid %d\n", inode->seq_pid);
```

```
+ mdelay(100);
```

```
+ }
```

```

+endif
    } while (read_seqcount_retry(&inode->i_size_seqcount, seq));
    return i_size;
#elif BITS_PER_LONG==32 && defined(CONFIG_PREEMPT)
@@ -467,9 +474,20 @@
static inline void i_size_write(struct inode *inode, loff_t i_size)
{
    #if BITS_PER_LONG==32 && defined(CONFIG_SMP)
+    #if 1 /* XXX */
+    inode->seq_pid = current->tgid;
+    write_seqcount_begin(&inode->i_size_seqcount);
+    inode->i_size = i_size;
+    write_seqcount_end(&inode->i_size_seqcount);
+    if (inode->i_size_seqcount.sequence & 1)
+    printk("i_size_write: pid %d: sequence is odd!\n",
+    current->tgid);
+    inode->seq_pid = 0;
+    #else
        write_seqcount_begin(&inode->i_size_seqcount);
        inode->i_size = i_size;
        write_seqcount_end(&inode->i_size_seqcount);
+    #endif
+endif
#elif BITS_PER_LONG==32 && defined(CONFIG_PREEMPT)
    preempt_disable();
    inode->i_size = i_size;

```

I then started the test that locks up the kernel, and it printed this:

```

i_size_write: pid 542: sequence is odd!
i_size_write: pid 543: sequence is odd!
i_size_write: pid 542: sequence is odd!

```

```

i_size_read() seems to be looping - pid 0
i_size_read() seems to be looping - pid 0
[this keeps on being printed and the kernel is locked up]

```

It took some time for the i_size_write messages to show up, and they were spaced 10-30 seconds apart, and during that time the server was still up - right until the first i_size_read message.

Then I added this patch:

```

--- fs.h.v1 2004-01-30 21:11:19.000000000 +0100
+++ fs.h 2004-01-30 20:16:35.000000000 +0100
@@ -426,6 +426,7 @@
#ifdef __NEED_I_SIZE_ORDERED
    seqcount_t i_size_seqcount;
    pid_t seq_pid; /* XXX */
+    spinlock_t i_size_lock;
#endif
};

```

```
@@ -475,6 +476,7 @@
{
#if BITS_PER_LONG==32 && defined(CONFIG_SMP)
#if 1 /* XXX */
+ spin_lock(&inode->i_size_lock);
  inode->seq_pid = current->tgid;
  write_seqcount_begin(&inode->i_size_seqcount);
  inode->i_size = i_size;
@@ -483,6 +485,7 @@
    printk("i_size_write: pid %d: sequence is odd!\n",
           current->tgid);
    inode->seq_pid = 0;
+ spin_unlock(&inode->i_size_lock);
#else
  write_seqcount_begin(&inode->i_size_seqcount);
  inode->i_size = i_size;
```

(and some code in fs/inode.c to initialize i_size_lock)

Guess what. No more debug output, no more lockups ... is there anything else I can do to debug this ? Because I'm not really sure what I'm doing, you see :)

Mike.

-

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@vger.kernel.org
More majordomo info at <http://vger.kernel.org/majordomo-info.html>
Please read the FAQ at <http://www.tux.org/lkml/>