

# RE: hugepage: Strict page reservation for hugepage inodes

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-02/msg09876.html>

---

- *From:* "Zhang, Yanmin" <[yanmin.zhang@xxxxxxxxx](mailto:yanmin.zhang@xxxxxxxxx)>
  - *Date:* Tue, 28 Feb 2006 17:21:16 +0800
- 

-----Original Message-----

From: David Gibson [<mailto:david@xxxxxxxxxxxxxxxxxxxxxxxx>]

Sent: 2006t2B8å 17:15

To: Zhang, Yanmin

Cc: Andrew Morton; William Lee Irwin; linux-kernel@xxxxxxxxxxxxxxxx

Subject: Re: hugepage: Strict page reservation for hugepage inodes

On Tue, Feb 28, 2006 at 04:53:49PM +0800, Zhang, Yanmin wrote:

[YM] Consider this scenario of multi-thread:  
One process has 2 threads. The process mmmaps a hugetlb area with 1 huge page and there is a free huge page. Later on, the 2 threads fault on the huge page at the same time. The second thread would fail, and WARN\_ON check is triggered, then the second thread is killed by function hugetlb\_no\_page.

That's why this patch *\*must\** go after my other patch which serializes the allocation->instantiation path.

[YM] Sorry, I didn't see other patches.

```
+int
hugetlb_extend_reservation(struct
hugetlbf_inode_info *info,
+ unsigned long atleast)
+{
+ struct inode *inode =
+ &info->vfs_inode;
+ struct address_space
+ *mapping =
+ inode->i_mapping;
```

RE: hugepage: Strict page reservation for hugepage inodes

```
+ unsigned long idx;
+ unsigned long
change_in_reserve = 0;
+ struct page *page;
+ int ret = 0;
+
+
spin_lock(&hugetlb_lock);
+
read_lock_irq(&inode->i_mapping->tree_lock);
+
+ if
(info->prereserved_hpages
>= atleast)
+ goto out;
+
+ /* prereserved_hpages
stores the number of pages
already
+ * guaranteed (reserved or
instantiated) for this inode.
+ * Count how many extra
pages we need to reserve. */
+ for (idx =
info->prereserved_hpages;
idx < atleast; idx++) {
+ page =
radix_tree_lookup(&mapping->page_tree,
idx);
+ if (!page)
+ /* Pages which are already
instantiated don't
+ * need to be reserved */
+ change_in_reserve++;
+ }
```

[YM] Why always to go through the page cache?  
prereserved\_hpages and  
reserved\_huge\_pages are protected by hugetlb\_lock.

Erm.. sorry, I don't see how that helps. We need to go through the  
page cache to see which pages have already been instantiated, and so  
don't need to be reserved.

[YM] The huge pages beyond info->prereserved\_hpages are always not allocated.

—

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

RE: hugepage: Strict page reservation for hugepage inodes

RE: hugepage: Strict page reservation for hugepage inodes

Please read the FAQ at <http://www.tux.org/lkml/>