Subject: [PATCH 0/3] ext4: Extent code cleanup
Posted by Dmitriy Monakhov on Fri, 22 Jun 2007 12:14:42 GMT
View Forum Message <> Reply to Message

Hi, i've finaly found time to review and test ext4 patches.
And i have some fixes:
[1] One more one line fix for ext4-block-reservation patch. IMHO
    it is better to merge it with my previous block reservation fixes.

[2] Fix compilation with EXT_DEBUG

[3] Some code cleanup

Subject: [PATCH 1/3] ext4 block reservation fix3
Posted by Dmitry Monakhov on Fri, 22 Jun 2007 12:16:38 GMT
View Forum Message <> Reply to Message

If ext4_reserve_block has failed we have to drop quota.

Signed-off-by: Dmitry Monakhov <dmonakhov@openvz.org>
---
 fs/ext4/balloc.c | 2 +-
 1 files changed, 1 insertions(+), 1 deletions(-)

diff --git a/fs/ext4/balloc.c b/fs/ext4/balloc.c
index a9655f1..eacffd3 100644
--- a/fs/ext4/balloc.c
+++ b/fs/ext4/balloc.c
@@ -1467,7 +1467,7 @@ ext4_fsblk_t ext4_new_blocks(handle_t *handle, struct inode *inode,
  if (!(EXT4_I(inode)->i_state & EXT4_STATE_BLOCKS_RESERVED)) {
   *errp = ext4_reserve_blocks(sb, num);
   if (*errp)
-   return 0;
+   goto out;
   reserved = num;
  }

--
1.5.2

Subject: [PATCH 2/3] extent compilation fixes
Posted by Dmitry Monakhov on Fri, 22 Jun 2007 12:17:53 GMT
View Forum Message <> Reply to Message

Fix compilation with EXT_DEBUG, also fix leXX_to_cpu convertions.

```
Signed-off-by: Dmitry Monakhov <dmonakhov@openvz.org>
---
 fs/ext4/extents.c |   18 ++++++++++--------
 1 files changed, 10 insertions(+), 8 deletions(-)

diff --git a/fs/ext4/extents.c b/fs/ext4/extents.c
index 6f72dcb..12fe3d7 100644
--- a/fs/ext4/extents.c
+++ b/fs/ext4/extents.c
@@ -382,13 +382,14 @@ ext4_ext_binsearch_idx(struct inode *inode, struct ext4_ext_path
*path, int bloc
    r = m - 1;
   else
    l = m + 1;
-  ext_debug("%p(%u):%p(%u):%p(%u) ", l, l->ei_block,
-    m, m->ei_block, r, r->ei_block);
+  ext_debug("%p(%u):%p(%u):%p(%u) ", l, le32_to_cpu(l->ei_block),
+    m, le32_to_cpu(m->ei_block),
+    r, le32_to_cpu(r->ei_block));
  }

  path->p_idx = l - 1;
  ext_debug("  -> %d->%lld ", le32_to_cpu(path->p_idx->ei_block),
-    idx_block(path->p_idx));
+    idx_pblock(path->p_idx));

 #ifdef CHECK_BINSEARCH
  {
@@ -447,8 +448,9 @@ ext4_ext_binsearch(struct inode *inode, struct ext4_ext_path *path, int
block)
    r = m - 1;
   else
    l = m + 1;
-  ext_debug("%p(%u):%p(%u):%p(%u) ", l, l->ee_block,
-    m, m->ee_block, r, r->ee_block);
+  ext_debug("%p(%u):%p(%u):%p(%u) ", l, le32_to_cpu(l->ee_block),
+    m, le32_to_cpu(m->ee_block),
+    r, le32_to_cpu(r->ee_block));
  }

  path->p_ext = l - 1;
@@ -580,7 +582,7 @@ static int ext4_ext_insert_index(handle_t *handle, struct inode *inode,
   if (curp->p_idx != EXT_LAST_INDEX(curp->p_hdr)) {
    len = (len - 1) * sizeof(struct ext4_extent_idx);
    len = len < 0 ? 0 : len;
-   ext_debug("insert new index %d after: %d. "
+   ext_debug("insert new index %d after: %llu. "
```

```
      "move %d from 0x%p to 0x%p\n",
      logical, ptr, len,
      (curp->p_idx + 1), (curp->p_idx + 2));
@@ -591,7 +593,7 @@ static int ext4_ext_insert_index(handle_t *handle, struct inode *inode,
   /* insert before */
   len = len * sizeof(struct ext4_extent_idx);
   len = len < 0 ? 0 : len;
-  ext_debug("insert new index %d before: %d. "
+  ext_debug("insert new index %d before: %llu. "
      "move %d from 0x%p to 0x%p\n",
      logical, ptr, len,
      curp->p_idx, (curp->p_idx + 1));
@@ -791,7 +793,7 @@ static int ext4_ext_split(handle_t *handle, struct inode *inode,
   BUG_ON(EXT_MAX_INDEX(path[i].p_hdr) !=
     EXT_LAST_INDEX(path[i].p_hdr));
   while (path[i].p_idx <= EXT_MAX_INDEX(path[i].p_hdr)) {
-   ext_debug("%d: move %d:%d in new index %llu\n", i,
+   ext_debug("%d: move %d:%llu in new index %llu\n", i,
      le32_to_cpu(path[i].p_idx->ei_block),
      idx_pblock(path[i].p_idx),
      newblock);
--
1.5.2
```

Subject: [PATCH 3/3] ext4: extent macros cleanup
Posted by Dmitry Monakhov on Fri, 22 Jun 2007 12:18:31 GMT
View Forum Message <> Reply to Message

```
- Replace math equation to it's macro equivalent
- make ext4_ext_grow_indepth() indexes/leaf correct

Signed-off-by: Dmitry Monakhov <dmonakhov@openvz.org>
---
 fs/ext4/extents.c |   11 +++++++----
 1 files changed, 7 insertions(+), 4 deletions(-)

diff --git a/fs/ext4/extents.c b/fs/ext4/extents.c
index 12fe3d7..1fd00ac 100644
--- a/fs/ext4/extents.c
+++ b/fs/ext4/extents.c
@@ -375,7 +375,7 @@ ext4_ext_binsearch_idx(struct inode *inode, struct ext4_ext_path *path,
int bloc
  ext_debug("binsearch for %d(idx):  ", block);

  l = EXT_FIRST_INDEX(eh) + 1;
- r = EXT_FIRST_INDEX(eh) + le16_to_cpu(eh->eh_entries) - 1;
+ r = EXT_LAST_INDEX(eh);
```

```
  while (l <= r) {
   m = l + (r - l) / 2;
   if (block < le32_to_cpu(m->ei_block))
@@ -440,7 +440,7 @@ ext4_ext_binsearch(struct inode *inode, struct ext4_ext_path *path, int
block)
  ext_debug("binsearch for %d:  ", block);

  l = EXT_FIRST_EXTENT(eh) + 1;
- r = EXT_FIRST_EXTENT(eh) + le16_to_cpu(eh->eh_entries) - 1;
+ r = EXT_LAST_EXTENT(eh);

  while (l <= r) {
   m = l + (r - l) / 2;
@@ -922,8 +922,11 @@ static int ext4_ext_grow_indepth(handle_t *handle, struct inode *inode,
  curp->p_hdr->eh_max = cpu_to_le16(ext4_ext_space_root_idx(inode));
  curp->p_hdr->eh_entries = cpu_to_le16(1);
  curp->p_idx = EXT_FIRST_INDEX(curp->p_hdr);
- /* FIXME: it works, but actually path[0] can be index */
- curp->p_idx->ei_block = EXT_FIRST_EXTENT(path[0].p_hdr)->ee_block;
+
+ if (path[0].p_hdr->eh_depth)
+   curp->p_idx->ei_block = EXT_FIRST_INDEX(path[0].p_hdr)->ei_block;
+ else
+   curp->p_idx->ei_block = EXT_FIRST_EXTENT(path[0].p_hdr)->ee_block;
  ext4_idx_store_pblock(curp->p_idx, newblock);

  neh = ext_inode_hdr(inode);
--
1.5.2
```

Subject: Re: [PATCH 3/3] ext4: extent macros cleanup
Posted by Alex Tomas on Mon, 25 Jun 2007 11:06:33 GMT
View Forum Message <> Reply to Message

Acked-off-by: Alex Tomas <alex@clusterfs.com>

Dmitry Monakhov wrote:
> - Replace math equation to it's macro equivalent
> - make ext4_ext_grow_indepth() indexes/leaf correct
>
> Signed-off-by: Dmitry Monakhov <dmonakhov@openvz.org>
> ---
> fs/ext4/extents.c | 11 +++++++----
> 1 files changed, 7 insertions(+), 4 deletions(-)
>
> diff --git a/fs/ext4/extents.c b/fs/ext4/extents.c

> index 12fe3d7..1fd00ac 100644
> --- a/fs/ext4/extents.c
> +++ b/fs/ext4/extents.c
> @@ -375,7 +375,7 @@ ext4_ext_binsearch_idx(struct inode *inode, struct ext4_ext_path
*path, int bloc
> ext_debug("binsearch for %d(idx): ", block);
>
> l = EXT_FIRST_INDEX(eh) + 1;
> - r = EXT_FIRST_INDEX(eh) + le16_to_cpu(eh->eh_entries) - 1;
> + r = EXT_LAST_INDEX(eh);
> while (l <= r) {
> m = l + (r - l) / 2;
> if (block < le32_to_cpu(m->ei_block))
> @@ -440,7 +440,7 @@ ext4_ext_binsearch(struct inode *inode, struct ext4_ext_path *path, int
block)
> ext_debug("binsearch for %d: ", block);
>
> l = EXT_FIRST_EXTENT(eh) + 1;
> - r = EXT_FIRST_EXTENT(eh) + le16_to_cpu(eh->eh_entries) - 1;
> + r = EXT_LAST_EXTENT(eh);
>
> while (l <= r) {
> m = l + (r - l) / 2;
> @@ -922,8 +922,11 @@ static int ext4_ext_grow_indepth(handle_t *handle, struct inode
*inode,
> curp->p_hdr->eh_max = cpu_to_le16(ext4_ext_space_root_idx(inode));
> curp->p_hdr->eh_entries = cpu_to_le16(1);
> curp->p_idx = EXT_FIRST_INDEX(curp->p_hdr);
> - /* FIXME: it works, but actually path[0] can be index */
> - curp->p_idx->ei_block = EXT_FIRST_EXTENT(path[0].p_hdr)->ee_block;
> +
> + if (path[0].p_hdr->eh_depth)
> + curp->p_idx->ei_block = EXT_FIRST_INDEX(path[0].p_hdr)->ei_block;
> + else
> + curp->p_idx->ei_block = EXT_FIRST_EXTENT(path[0].p_hdr)->ee_block;
> ext4_idx_store_pblock(curp->p_idx, newblock);
>
> neh = ext_inode_hdr(inode);

Subject: Re: [PATCH 1/3] ext4 block reservation fix3
Posted by Alex Tomas on Mon, 25 Jun 2007 11:35:33 GMT
View Forum Message <> Reply to Message

Acked-off-by: Alex Tomas <alex@clusterfs.com>

thanks, Alex

Dmitry Monakhov wrote:
> If ext4_reserve_block has failed we have to drop quota.
>
> Signed-off-by: Dmitry Monakhov <dmonakhov@openvz.org>
> ---
> fs/ext4/balloc.c | 2 +-
> 1 files changed, 1 insertions(+), 1 deletions(-)
>
> diff --git a/fs/ext4/balloc.c b/fs/ext4/balloc.c
> index a9655f1..eacffd3 100644
> --- a/fs/ext4/balloc.c
> +++ b/fs/ext4/balloc.c
> @@ -1467,7 +1467,7 @@ ext4_fsblk_t ext4_new_blocks(handle_t *handle, struct inode *inode,
>   if (!(EXT4_I(inode)->i_state & EXT4_STATE_BLOCKS_RESERVED)) {
>   *errp = ext4_reserve_blocks(sb, num);
>   if (*errp)
> -   return 0;
> +   goto out;
>   reserved = num;
>   }
>

Subject: Re: [PATCH 2/3] extent compilation fixes
Posted by Alex Tomas on Mon, 25 Jun 2007 11:38:12 GMT
View Forum Message <> Reply to Message

Acked-off-by: Alex Tomas <alex@clusterfs.com>

thanks, Alex

Dmitry Monakhov wrote:
> Fix compilation with EXT_DEBUG, also fix leXX_to_cpu convertions.
>
> Signed-off-by: Dmitry Monakhov <dmonakhov@openvz.org>
> ---
> fs/ext4/extents.c | 18 ++++++++++--------
> 1 files changed, 10 insertions(+), 8 deletions(-)
>
> diff --git a/fs/ext4/extents.c b/fs/ext4/extents.c
> index 6f72dcb..12fe3d7 100644
> --- a/fs/ext4/extents.c
> +++ b/fs/ext4/extents.c
> @@ -382,13 +382,14 @@ ext4_ext_binsearch_idx(struct inode *inode, struct ext4_ext_path
*path, int bloc
>   r = m - 1;
>   else
>   l = m + 1;

```
> -  ext_debug("%p(%u):%p(%u):%p(%u) ", l, l->ei_block,
> -    m, m->ei_block, r, r->ei_block);
> +  ext_debug("%p(%u):%p(%u):%p(%u) ", l, le32_to_cpu(l->ei_block),
> +    m, le32_to_cpu(m->ei_block),
> +    r, le32_to_cpu(r->ei_block));
>   }
>
>   path->p_idx = l - 1;
>   ext_debug("  -> %d->%lld ", le32_to_cpu(path->p_idx->ei_block),
> -    idx_block(path->p_idx));
> +    idx_pblock(path->p_idx));
>
>  #ifdef CHECK_BINSEARCH
>   {
> @@ -447,8 +448,9 @@ ext4_ext_binsearch(struct inode *inode, struct ext4_ext_path *path, int
block)
>     r = m - 1;
>    else
>     l = m + 1;
> -  ext_debug("%p(%u):%p(%u):%p(%u) ", l, l->ee_block,
> -    m, m->ee_block, r, r->ee_block);
> +  ext_debug("%p(%u):%p(%u):%p(%u) ", l, le32_to_cpu(l->ee_block),
> +    m, le32_to_cpu(m->ee_block),
> +    r, le32_to_cpu(r->ee_block));
>   }
>
>   path->p_ext = l - 1;
> @@ -580,7 +582,7 @@ static int ext4_ext_insert_index(handle_t *handle, struct inode *inode,
>    if (curp->p_idx != EXT_LAST_INDEX(curp->p_hdr)) {
>     len = (len - 1) * sizeof(struct ext4_extent_idx);
>     len = len < 0 ? 0 : len;
> -   ext_debug("insert new index %d after: %d. "
> +   ext_debug("insert new index %d after: %llu. "
>       "move %d from 0x%p to 0x%p\n",
>       logical, ptr, len,
>       (curp->p_idx + 1), (curp->p_idx + 2));
> @@ -591,7 +593,7 @@ static int ext4_ext_insert_index(handle_t *handle, struct inode *inode,
>    /* insert before */
>    len = len * sizeof(struct ext4_extent_idx);
>    len = len < 0 ? 0 : len;
> -  ext_debug("insert new index %d before: %d. "
> +  ext_debug("insert new index %d before: %llu. "
>      "move %d from 0x%p to 0x%p\n",
>      logical, ptr, len,
>      curp->p_idx, (curp->p_idx + 1));
> @@ -791,7 +793,7 @@ static int ext4_ext_split(handle_t *handle, struct inode *inode,
>    BUG_ON(EXT_MAX_INDEX(path[i].p_hdr) !=
>      EXT_LAST_INDEX(path[i].p_hdr));
```

```
>    while (path[i].p_idx <= EXT_MAX_INDEX(path[i].p_hdr)) {
> -   ext_debug("%d: move %d:%d in new index %llu\n", i,
> +   ext_debug("%d: move %d:%llu in new index %llu\n", i,
>       le32_to_cpu(path[i].p_idx->ei_block),
>       idx_pblock(path[i].p_idx),
>       newblock);
```