

# [PATCH 1/2] btrfs-progs: Fix printf format casting errors

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2008-03/msg12131.html>

---

- *From:* Alex Chiang <[achiang@xxxxxx](mailto:achiang@xxxxxx)>
  - *Date:* Mon, 31 Mar 2008 14:28:23 -0600
- 

From: Alex Chiang <[achiang@xxxxxx](mailto:achiang@xxxxxx)>  
MIME-Version: 1.0  
Content-Type: text/plain; charset=utf-8  
Content-Transfer-Encoding: 8bit

We get lots of warnings of the flavor:

```
utils.c:441: warning: format '%Lu' expects type 'long long unsigned int' but argument 2 has type 'u64'
```

And thanks to `-Werror`, the build fails. Clean up these printf's by properly casting the arg to the format specified.

Signed-off-by: Alex Chiang <[achiang@xxxxxx](mailto:achiang@xxxxxx)>

```
---  
ctree.c | 5 +++--  
disk-io.c | 7 ++++---  
extent-tree.c | 24 ++++++++-----  
extent_io.c | 3 ++-  
file-item.c | 8 ++++---  
inode-map.c | 3 ++-  
root-tree.c | 5 ++++--  
utils.c | 2 +-  
volumes.c | 14 ++++++++-----  
9 files changed, 45 insertions(+), 26 deletions(-)
```

```
diff --git a/ctree.c b/ctree.c  
index 88ebd9e..5311306 100644  
--- a/ctree.c  
+++ b/ctree.c  
@@ -237,8 +237,9 @@ int btrfs_cow_block(struct btrfs_trans_handle *trans,  
 }  
 */  
if (trans->transid != root->fs_info->generation) {  
- printk(KERN_CRIT "trans %Lu running %Lu\n", trans->transid,  
- root->fs_info->generation);  
+ printk(KERN_CRIT "trans %llu running %llu\n",  
+ (unsigned long long)trans->transid,  
+ (unsigned long long)root->fs_info->generation);
```

[PATCH 1/2] btrfs-progs: Fix printf format casting errors

```
WARN_ON(1);
}
if (btrfs_header_generation(buf) == trans->transid) {
diff --git a/disk-io.c b/disk-io.c
index 1afe5a6..8ee7716 100644
--- a/disk-io.c
+++ b/disk-io.c
@@ -84,7 +84,8 @@ static int csum_tree_block(struct btrfs_root *root, struct extent_buffer *buf,

if (verify) {
if (memcmp_extent_buffer(buf, result, 0, BTRFS_CRC32_SIZE)) {
- printk("checksum verify failed on %llu\n", buf->start);
+ printk("checksum verify failed on %llu\n",
+ (unsigned long long)buf->start);
return 1;
}
} else {
@@ -429,8 +430,8 @@ struct btrfs_root *open_ctree_fd(int fp, const char *path, u64 sb_bytenr)
fprintf(stderr, "No valid Btrfs found on %s\n", path);
return NULL;
}
- fprintf(stderr, "found Btrfs on %s with %Lu devices\n", path,
- total_devs);
+ fprintf(stderr, "found Btrfs on %s with %lu devices\n", path,
+ (unsigned long)total_devs);

if (total_devs != 1) {
ret = btrfs_scan_for_fsid(fs_devices, total_devs, 1);
diff --git a/extent-tree.c b/extent-tree.c
index 9696aab..b9cf92f 100644
--- a/extent-tree.c
+++ b/extent-tree.c
@@ -225,7 +225,8 @@ again:
out:
cache = btrfs_lookup_block_group(root->fs_info, search_start);
if (!cache) {
- printk("Unable to find block group for %Lu\n", search_start);
+ printk("Unable to find block group for %llu\n",
+ (unsigned long long)search_start);
WARN_ON(1);
}
return -ENOSPC;
@@ -680,7 +681,8 @@ static int lookup_extent_ref(struct btrfs_trans_handle *trans,
goto out;
if (ret != 0) {
btrfs_print_leaf(root, path->nodes[0]);
- printk("failed to find block number %Lu\n", bytenr);
+ printk("failed to find block number %llu\n",
+ (unsigned long long)bytenr);
BUG();
}
}
```

[PATCH 1/2] btrfs-progs: Fix printf format casting errors

```
l = path->nodes[0];
@@ -1046,7 +1048,7 @@ static int do_chunk_alloc(struct btrfs_trans_handle *trans,

ret = btrfs_alloc_chunk(trans, extent_root, &start, &num_bytes, flags);
if (ret == -ENOSPC) {
-printf("space info full %Lu\n", flags);
+printf("space info full %llu\n", (unsigned long long)flags);
space_info->full = 1;
return 0;
}
@@ -1315,10 +1317,13 @@ static int __free_extent(struct btrfs_trans_handle *trans, struct btrfs_root
} else {
btrfs_print_leaf(extent_root, path->nodes[0]);
WARN_ON(1);
- printf("Unable to find ref byte nr %Lu root %Lu "
- " gen %Lu owner %Lu offset %Lu\n", bytenr,
- root_objectid, ref_generation, owner_objectid,
- owner_offset);
+ printf("Unable to find ref byte nr %llu root %llu "
+ " gen %llu owner %llu offset %llu\n",
+ (unsigned long long)bytenr,
+ (unsigned long long)root_objectid,
+ (unsigned long long)ref_generation,
+ (unsigned long long)owner_objectid,
+ (unsigned long long)owner_offset);
}
if (!found_extent) {
btrfs_release_path(extent_root, path);
@@ -1720,8 +1725,9 @@ int btrfs_alloc_extent(struct btrfs_trans_handle *trans,
update_block:
ret = update_block_group(trans, root, ins->objectid, ins->offset, 1, 0);
if (ret) {
- printf("update block group failed for %Lu %Lu\n",
- ins->objectid, ins->offset);
+ printf("update block group failed for %llu %llu\n",
+ (unsigned long long)ins->objectid,
+ (unsigned long long)ins->offset);
BUG();
}
return 0;
diff --git a/extent_io.c b/extent_io.c
index b663275..9071644 100644
--- a/extent_io.c
+++ b/extent_io.c
@@ -68,7 +68,8 @@ void extent_io_tree_cleanup(struct extent_io_tree *tree)
eb = list_entry(tree->lru.next, struct extent_buffer, lru);
if (eb->refs != 1) {
fprintf(stderr, "extent buffer leak: "
- "start %Lu len %u\n", eb->start, eb->len);
+ "start %llu len %u\n",
+ (unsigned long long)eb->start, eb->len);
```

[PATCH 1/2] btrfs-progs: Fix printf format casting errors

```
eb->refs = 1;
}
free_extent_buffer(eb);
diff --git a/file-item.c b/file-item.c
index 8b85f7a..4cbc2a5 100644
--- a/file-item.c
+++ b/file-item.c
@@ -136,7 +136,8 @@ int btrfs_insert_inline_extent(struct btrfs_trans_handle *trans,
err = ret;
btrfs_print_leaf(root, leaf);
printf("found wasn't inline offset %llu inode %llu\n",
- offset, objectid);
+ (unsigned long long)offset,
+ (unsigned long long)objectid);
goto fail;
}
found_size = btrfs_file_extent_inline_len(leaf,
@@ -386,8 +387,9 @@ found:
csum_result = btrfs_csum_data(root, data, csum_result, len);
btrfs_csum_final(csum_result, (char *)&csum_result);
if (csum_result == 0) {
- printf("csum result is 0 for inode %Lu offset %Lu\n",
- objectid, offset);
+ printf("csum result is 0 for inode %llu offset %llu\n",
+ (unsigned long long)objectid,
+ (unsigned long long)offset);
}

write_extent_buffer(leaf, &csum_result, (unsigned long)item,
diff --git a/inode-map.c b/inode-map.c
index a0925ea..d2970d4 100644
--- a/inode-map.c
+++ b/inode-map.c
@@ -72,7 +72,8 @@ int btrfs_find_free_objectid(struct btrfs_trans_handle *trans,
path = btrfs_alloc_path();
BUG_ON(!path);
search_start = root->last_inode_alloc;
- search_start = max(search_start, BTRFS_FIRST_FREE_OBJECTID);
+ search_start = max((unsigned long long)search_start,
+ BTRFS_FIRST_FREE_OBJECTID);
search_key.objectid = search_start;
search_key.offset = 0;

diff --git a/root-tree.c b/root-tree.c
index b639214..96c05cb 100644
--- a/root-tree.c
+++ b/root-tree.c
@@ -183,7 +183,10 @@ int btrfs_del_root(struct btrfs_trans_handle *trans, struct btrfs_root *root,
goto out;
if (ret) {
btrfs_print_leaf(root, path->nodes[0]);
```

[PATCH 1/2] btrfs-progs: Fix printf format casting errors

```
-printf("failed to del %Lu %u %Lu\n", key->objectid, key->type, key->offset);
+printf("failed to del %llu %u %llu\n",
+ (unsigned long long)key->objectid,
+ key->type,
+ (unsigned long long)key->offset);

}
BUG_ON(ret != 0);
diff --git a/Utils.c b/Utils.c
index 0a067e6..7d5a5d6 100644
--- a/Utils.c
+++ b/Utils.c
@@ -438,7 +438,7 @@ int btrfs_add_to_fsid(struct btrfs_trans_handle *trans,
memcpy(disk_super, super, sizeof(*disk_super));

- printf("adding device id %Lu\n", device.devid);
+ printf("adding device id %llu\n", (unsigned long long)device.devid);
btrfs_set_stack_device_id(dev_item, device.devid);
btrfs_set_stack_device_type(dev_item, device.type);
btrfs_set_stack_device_io_align(dev_item, device.io_align);
diff --git a/volumes.c b/volumes.c
index 7ab48b1..7127677 100644
--- a/volumes.c
+++ b/volumes.c
@@ -121,7 +121,7 @@ static int device_list_add(const char *path,
}
if (fs_devices->lowest_devid > devid) {
fs_devices->lowest_devid = devid;
- printf("lowest devid now %Lu\n", devid);
+ printf("lowest devid now %llu\n", (unsigned long long)devid);
}
*fs_devices_ret = fs_devices;
return 0;
@@ -151,7 +151,8 @@ int btrfs_open_devices(struct btrfs_fs_devices *fs_devices, int flags)
list_for_each(cur, head) {
device = list_entry(cur, struct btrfs_device, dev_list);
fd = open(device->name, flags);
-printf("opening %s devid %Lu fd %d\n", device->name, device->devid, fd);
+printf("opening %s devid %llu fd %d\n", device->name,
+ (unsigned long long)device->devid, fd);
if (fd < 0) {
ret = -errno;
goto fail;
@@ -195,7 +196,7 @@ int btrfs_scan_one_device(int fd, const char *path,
}
devid = le64_to_cpu(disk_super->dev_item.devid);
*total_devs = btrfs_super_num_devices(disk_super);
- printf("found device %Lu on %s\n", devid, path);
+ printf("found device %llu on %s\n", (unsigned long long)devid, path);
ret = device_list_add(path, disk_super, devid, fs_devices_ret);
```

[PATCH 1/2] btrfs-progs: Fix printf format casting errors

```
error_brelse:
@@ -639,7 +640,9 @@ again:
key.objectid,
calc_size, &dev_offset);
BUG_ON(ret);
-printf("alloc chunk size %Lu from dev %Lu\n", calc_size, device->devid);
+printf("alloc chunk size %llu from dev %llu\n",
+ (unsigned long long)calc_size,
+ (unsigned long long)device->devid);
device->bytes_used += calc_size;
ret = btrfs_update_device(trans, device);
BUG_ON(ret);
@@ -838,7 +841,8 @@ static int read_one_dev(struct btrfs_root *root,
devid = btrfs_device_id(leaf, dev_item);
device = btrfs_find_device(root, devid);
if (!device) {
- printf("warning devid %Lu not found already\n", devid);
+ printf("warning devid %llu not found already\n",
+ (unsigned long long)devid);
device = kmalloc(sizeof(*device), GFP_NOFS);
if (!device)
return -ENOMEM;
```

1.5.3.1.g1e61

---

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