Subject: [PATCH] ecryptfs ecryptfs_read_super path_lookup errh fix
Posted by Dmitriy Monakhov on Mon, 19 Feb 2007 15:57:21 GMT
View Forum Message <> Reply to Message

if path_lookup() return non zero code we don't have to worry about 'nd'
parameter, but ecryptfs_read_super does path_release(&nd) after path_lookup
has failed, and dentry counter becomes negative :) , this result in folowing BUG

ecryptfs_read_super: path_lookup() failed
BUG: Dentry f3f074c4{i=0,n=enc_dir} still in use (-1) [unmount of ext3 dm-5]
<<<<<<<<<<<<<<<<<<<<counter is negative here ^^^^^^
------------[cut here]------------
kernel BUG at fs/dcache.c:654!
invalid opcode: 0000 [#1]
SMP
last sysfs file: /fs/ecryptfs/version
Modules linked in: ecb ecryptfs bluetooth video e1000
CPU: 1
EIP: 0060:[<c01831a7>] Not tainted VLI
EFLAGS: 00210292 (2.6.20-mm2-g0cb8e157-dirty #10)
EIP is at shrink_dcache_for_umount_subtree+0x159/0x1fb
eax: 0000004b ebx: f3f074c4 ecx: c0122e78 edx: f7612070
esi: ffffffff edi: f3f0753c ebp: f4269cac esp: f4269c7c
ds: 007b es: 007b fs: 00d8 gs: 0033 ss: 0068
Process mount.ecryptfs (pid: 4236, ti=f4268000 task=f7612070 task.ti=f4268000)
Stack: c04f9dee f3f074c4 00000000 f3f0753c ffffffff c04fd617 f5161e30 00000000
00000000 f5161c00 c043b780 f4269d98 f4269cb8 c0183f5f f5161c00 f4269cd0
c0176166 f4269cc8 c022cb90 f705fcc0 f5161c00 f4269ce0 c017622c f5161c00
Call Trace:
[<c0105f70>] show_trace_log_lvl+0x1a/0x2f
[<c0106020>] show_stack_log_lvl+0x9b/0xaa
[<c01061e7>] show_registers+0x1b8/0x289
[<c01063d6>] die+0x11e/0x233
[<c042ed50>] do_trap+0x8a/0xa3
[<c0106979>] do_invalid_op+0x97/0xa1
[<c042eb14>] error_code+0x7c/0x84
[<c0183f5f>] shrink_dcache_for_umount+0x2d/0x3a
[<c0176166>] generic_shutdown_super+0x18/0xbe
[<c017622c>] kill_block_super+0x20/0x32
[<c01762ce>] deactivate_super+0x3f/0x51
[<c018724d>] mntput_no_expire+0x42/0x6b
[<c017a719>] path_release+0x20/0x23
[<f886387d>] ecryptfs_get_sb+0x45a/0x4ad [ecryptfs]
[<c0176361>] vfs_kern_mount+0x81/0xf1
[<c0176419>] do_kern_mount+0x30/0x42
[<c018847b>] do_mount+0x601/0x678
[<c0188561>] sys_mount+0x6f/0xa9
[<c0104f2c>] sysenter_past_esp+0x5d/0x99

 ======================
Code: 30 02 00 00 89 44 24 18 8b 45 ec 89 4c 24 14 89 74 24 10 89 7c 24 0c 89 5c 24 04 89 44 24 08 c7 04 24 ee 9d 4f c0 e8 b7 74 fa ff <0f> 0b eb fe 8b 73 30 39 de 75 04 31 f6 eb 03 f0 ff 0e 8d 43 48
EIP: [<c01831a7>] shrink_dcache_for_umount_subtree+0x159/0x1fb SS:ESP 0068:f4269c7c

This is easy to reproduce just try to mount ecryptfs to nonexisting lower path
mount -tecryptfs private/this_dir_not_exist root -ocipher=aes

Signed-off-by: Dmitriy Monakhov <dmonakhov@openvz.org>

```
diff --git a/fs/ecryptfs/main.c b/fs/ecryptfs/main.c
index 80044d1..812427e 100644
--- a/fs/ecryptfs/main.c
+++ b/fs/ecryptfs/main.c
@@ -487,7 +487,7 @@ static int ecryptfs_read_super(struct super_block *sb, const char
*dev_name)
  rc = path_lookup(dev_name, LOOKUP_FOLLOW, &nd);
  if (rc) {
   ecryptfs_printk(KERN_WARNING, "path_lookup() failed\n");
-  goto out_free;
+  goto out;
  }
  lower_root = nd.dentry;
  if (!lower_root->d_inode) {
```

Subject: Re: [PATCH] ecryptfs ecryptfs_read_super path_lookup errh fix
Posted by Dmitriy Monakhov on Mon, 19 Feb 2007 19:11:39 GMT
View Forum Message <> Reply to Message

Dmitriy Monakhov <dmonakhov@sw.ru> writes:

> if path_lookup() return non zero code we don't have to worry about 'nd'
> parameter, but ecryptfs_read_super does path_release(&nd) after path_lookup
> has failed, and dentry counter becomes negative :) , this result in folowing BUG
>
> ecryptfs_read_super: path_lookup() failed
> BUG: Dentry f3f074c4{i=0,n=enc_dir} still in use (-1) [unmount of ext3 dm-5]
> <<<<<<<<<<<<<<<<<<<<counter is negative here   ^^^^^^
> ------------[cut here]------------
> kernel BUG at fs/dcache.c:654!
> [<c01063d6>] die+0x11e/0x233
> [<c042ed50>] do_trap+0x8a/0xa3
> [<c0106979>] do_invalid_op+0x97/0xa1
> [<c042eb14>] error_code+0x7c/0x84
> [<c0183f5f>] shrink_dcache_for_umount+0x2d/0x3a

> [<c0176166>] generic_shutdown_super+0x18/0xbe
> [<c017622c>] kill_block_super+0x20/0x32
> [<c01762ce>] deactivate_super+0x3f/0x51
> [<c018724d>] mntput_no_expire+0x42/0x6b
> [<c017a719>] path_release+0x20/0x23
> [<f886387d>] ecryptfs_get_sb+0x45a/0x4ad [ecryptfs]
> [<c0176361>] vfs_kern_mount+0x81/0xf1
> [<c0176419>] do_kern_mount+0x30/0x42
> [<c018847b>] do_mount+0x601/0x678
> [<c0188561>] sys_mount+0x6f/0xa9
> [<c0104f2c>] sysenter_past_esp+0x5d/0x99
> =======================
> Code: 30 02 00 00 89 44 24 18 8b 45 ec 89 4c 24 14 89 74 24 10 89 7c 24 0c 89 5c 24 04 89
44 24 08 c7 04 24 ee 9d 4f c0 e8 b7 74 fa ff <0f> 0b eb fe 8b 73 30 39 de 75 04 31 f6 eb 03 f0 ff
0e 8d 43 48
> EIP: [<c01831a7>] shrink_dcache_for_umount_subtree+0x159/0x1fb SS:ESP 0068:f4269c7c
>
> This is easy to reproduce just try to mount ecryptfs to nonexisting lower path
> # mount -tecryptfs private/this_dir_not_exist root -ocipher=aes
This is fun but ecryptfs_read_super() code is more crappy whan it looks at
first blush.
1) After path_lookup succeed we dot't have any guarantie what it is DIR.
   This must be checked explicitly.
2) path_lookup can't return negative dentry, So inode check is useless.

Following patch is updated version and have to be applied instead of first patch
Signed-off-by: Dmitriy Monakhov <dmonakhov@openvz.org>

```
diff --git a/fs/ecryptfs/main.c b/fs/ecryptfs/main.c
index 80044d1..fc4a3a2 100644
--- a/fs/ecryptfs/main.c
+++ b/fs/ecryptfs/main.c
@@ -484,18 +484,12 @@ static int ecryptfs_read_super(struct super_block *sb, const char
*dev_name)
  struct vfsmount *lower_mnt;

  memset(&nd, 0, sizeof(struct nameidata));
- rc = path_lookup(dev_name, LOOKUP_FOLLOW, &nd);
+ rc = path_lookup(dev_name, LOOKUP_FOLLOW | LOOKUP_DIRECTORY, &nd);
  if (rc) {
   ecryptfs_printk(KERN_WARNING, "path_lookup() failed\n");
-  goto out_free;
+  goto out;
  }
  lower_root = nd.dentry;
- if (!lower_root->d_inode) {
-  ecryptfs_printk(KERN_WARNING,
```

```
-    "No directory to interpose on\n");
-  rc = -ENOENT;
-  goto out_free;
- }
   lower_mnt = nd.mnt;
   ecryptfs_set_superblock_lower(sb, lower_root->d_sb);
   sb->s_maxbytes = lower_root->d_sb->s_maxbytes;
```