Subject: Re: Sv: Re: Infinite loop in __d_lookup ?
Posted by xemul on Wed, 21 May 2008 11:46:11 GMT
View Forum Message <> Reply to Message

Jakob Goldbach wrote:
> Hi Pavel (and others)
>
> Loop is in __d_lookup as trace show. Any ideas ?

Well. It this really happens, then we have a corrupted
chain of dentries. Let's try to catch this corruption
early.

Here's the debugging patch that checks the chain to be
consistent when entries are added/removed from it.

Thanks for your help, Jakob :)

I've added a BUG with this issue - please, continue
communication via bugzilla since now:
http://bugzilla.openvz.org/show_bug.cgi?id=895

> /Jakob
>
>
> [76893.524305] __d_lookup: Abort on 5000 loop iteration in a chain
> [76893.525411]
> [76893.525412] Call Trace:
> [76893.526538] [<ffffffff8020ae20>] show_trace+0xae/0x360
> [76893.527619] [<ffffffff8020b0e7>] dump_stack+0x15/0x17
> [76893.528677] [<ffffffff8029b343>] __d_lookup+0x13a/0x187
> [76893.529779] [<ffffffff8029105d>] do_lookup+0x2c/0x193
> [76893.530846] [<ffffffff80293122>] __link_path_walk+0xb07/0x10ac
> [76893.532066] [<ffffffff8029374e>] link_path_walk+0x87/0x140
> [76893.533230] [<ffffffff80293c76>] do_path_lookup+0x2d3/0x2f8
> [76893.534404] [<ffffffff802945e2>] __user_walk_fd+0x41/0x62
> [76893.535559] [<ffffffff80282a09>] sys_faccessat+0xf4/0x1b5
> [76893.536705] [<ffffffff80282add>] sys_access+0x13/0x15
> [76893.537873] [<ffffffff80209902>] system_call+0x7e/0x83
> [76893.538898] DWARF2 unwinder stuck at system_call+0x7e/0x83
> [76893.539964] Leftover inexact backtrace:
> [76893.540768]
> [76893.541202] __d_lookup: Abort on 5000 loop iteration in a chain

--- ./fs/dcache.c.ddebug2 2008-05-21 14:52:15.000000000 +0400
+++ ./fs/dcache.c 2008-05-21 15:10:06.000000000 +0400
@@ -1350,6 +1350,18 @@ static void __d_rehash(struct dentry * e

```
 {

   entry->d_flags &= ~DCACHE_UNHASHED;
+ if (!spin_is_locked(&dcache_lock)) {
+  printk(KERN_ERR "Dcache lock is not taken on add\n");
+  dump_stack();
+ } else if (list->first != NULL &&
+    list->first->pprev != &list->first) {
+  printk(KERN_ERR "Dcache chain corruption:\n");
+  printk(KERN_ERR "Chain %p --next-> %p\n",
+     list, list->first);
+  printk(KERN_ERR "First %p <-pprev- %p\n",
+     list->first, list->first->pprev);
+  dump_stack();
+ }
   hlist_add_head_rcu(&entry->d_hash, list);
 }

@@ -1443,6 +1455,32 @@ static void switch_names(struct dentry *
  * dcache entries should not be moved in this way.
  */

+void d_node_check(struct hlist_node *n)
+{
+ if (!spin_is_locked(&dcache_lock)) {
+  printk(KERN_ERR "Dcache lock is not taken on del\n");
+  dump_stack();
+ }
+
+ if (n->next != NULL &&
+    n->next->pprev != &n->next) {
+  printk(KERN_ERR "Dentry d_hash node corruption(m1):\n");
+  printk(KERN_ERR "Node %p --next-> %p\n",
+     n, n->next);
+  printk(KERN_ERR "Next %p <-pprev- %p\n",
+     n->next, n->next->pprev);
+  dump_stack();
+ }
+
+ if (*n->pprev != n) {
+  printk(KERN_ERR "Dentry d_hash node corruption(m2):\n");
+  printk(KERN_ERR "Node %p <-pprev- %p -> %p\n",
+     n, n->pprev, *n->pprev);
+  dump_stack();
+ }
+}
+EXPORT_SYMBOL(d_node_check);
+
```

```
 void d_move(struct dentry * dentry, struct dentry * target)
 {
  struct hlist_head *list;
@@ -1467,6 +1505,7 @@ void d_move(struct dentry * dentry, stru
  if (dentry->d_flags & DCACHE_UNHASHED)
   goto already_unhashed;

+ d_node_check(&dentry->d_hash);
  hlist_del_rcu(&dentry->d_hash);

 already_unhashed:
--- ./include/linux/dcache.h.ddebug2 2008-05-21 14:50:31.000000000 +0400
+++ ./include/linux/dcache.h 2008-05-21 15:09:03.000000000 +0400
@@ -203,10 +203,13 @@ extern spinlock_t dcache_lock;
  * __d_drop requires dentry->d_lock.
  */

+void d_node_check(struct hlist_node *n);
+
 static inline void __d_drop(struct dentry *dentry)
 {
  if (!(dentry->d_flags & DCACHE_UNHASHED)) {
   dentry->d_flags |= DCACHE_UNHASHED;
+  d_node_check(&dentry->d_hash);
   hlist_del_rcu(&dentry->d_hash);
  }
 }
```