Subject: [PATCH] diskquota: 32bit quota tools on 64bit architectures
Posted by Anonymous Coward on Thu, 19 Oct 2006 12:30:37 GMT
View Forum Message <> Reply to Message

OpenVZ Linux kernel team has discovered the problem
with 32bit quota tools working on 64bit architectures.
In 2.6.10 kernel sys32_quotactl() function was replaced by sys_quotactl() with
the comment "sys_quotactl seems to be 32/64bit clean, enable it for 32bit"
However this isn't right. Look at if_dqblk structure:

```
struct if_dqblk {
        __u64 dqb_bhardlimit;
        __u64 dqb_bsoftlimit;
        __u64 dqb_curspace;
        __u64 dqb_ihardlimit;
        __u64 dqb_isoftlimit;
        __u64 dqb_curinodes;
        __u64 dqb_btime;
        __u64 dqb_itime;
        __u32 dqb_valid;
};
```

For 32 bit quota tools sizeof(if_dqblk) == 0x44.
But for 64 bit kernel its size is 0x48, 'cause of alignment!
Thus we got a problem.
Attached patch reintroduce sys32_quotactl() function,
that handles the situation.

Signed-off-by: Vasily Tarasov <vtaras@openvz.org>
Acked-by: Dmitry Mishin <dim@openvz.org>

In OpenVZ technology 32 bit Virtual Environments over
64 bit OS are common, hence we have customers, that complains on this bad quota
behaviour:

```
# /usr/bin/quota
quota: error while getting quota from /dev/sda1 for 0: Success
```

The reason is caused above.

```
--- linux-2.6.18/arch/ia64/ia32/sys_ia32.c.quot32 2006-09-20 07:42:06.000000000 +0400
+++ linux-2.6.18/arch/ia64/ia32/sys_ia32.c 2006-10-19 11:17:50.000000000 +0400
@@ -2545,6 +2545,54 @@ long sys32_fadvise64_64(int fd, __u32 of
         advice);
 }
```

```
+asmlinkage long sys32_quotactl(unsigned int cmd, const char __user *special,
+       qid_t id, void __user *addr)
+{
+ long ret;
+ unsigned int cmds;
+ mm_segment_t old_fs;
+ struct if_dqblk dqblk;
+ struct if32_dqblk {
+  __u32 dqb_bhardlimit[2];
+  __u32 dqb_bsoftlimit[2];
+  __u32 dqb_curspace[2];
+  __u32 dqb_ihardlimit[2];
+  __u32 dqb_isoftlimit[2];
+  __u32 dqb_curinodes[2];
+  __u32 dqb_btime[2];
+  __u32 dqb_itime[2];
+  __u32 dqb_valid;
+ } dqblk32;
+
+ cmds = cmd >> SUBCMDSHIFT;
+
+ switch (cmds) {
+  case Q_GETQUOTA:
+   old_fs = get_fs();
+   set_fs(KERNEL_DS);
+   ret = sys_quotactl(cmd, special, id, &dqblk);
+   set_fs(old_fs);
+   memcpy(&dqblk32, &dqblk, sizeof(dqblk32));
+   dqblk32.dqb_valid = dqblk.dqb_valid;
+   if (copy_to_user(addr, &dqblk32, sizeof(dqblk32)))
+    return -EFAULT;
+   break;
+  case Q_SETQUOTA:
+   if (copy_from_user(&dqblk32, addr, sizeof(dqblk32)))
+    return -EFAULT;
+   memcpy(&dqblk, &dqblk32, sizeof(dqblk32));
+   dqblk.dqb_valid = dqblk32.dqb_valid;
+   old_fs = get_fs();
+   set_fs(KERNEL_DS);
+   ret = sys_quotactl(cmd, special, id, &dqblk);
+   set_fs(old_fs);
+   break;
+  default:
+   return sys_quotactl(cmd, special, id, addr);
+ }
+ return ret;
+}
+
```

```
 #ifdef NOTYET  /* UNTESTED FOR IA64 FROM HERE DOWN */

 asmlinkage long sys32_setreuid(compat_uid_t ruid, compat_uid_t euid)
--- linux-2.6.18/arch/ia64/ia32/ia32_entry.S.quot32 2006-09-20 07:42:06.000000000 +0400
+++ linux-2.6.18/arch/ia64/ia32/ia32_entry.S 2006-10-19 11:15:52.000000000 +0400
@@ -341,7 +341,7 @@ ia32_syscall_table:
  data8 sys_ni_syscall /* init_module */
  data8 sys_ni_syscall /* delete_module */
  data8 sys_ni_syscall /* get_kernel_syms */  /* 130 */
- data8 sys_quotactl
+ data8 sys32_quotactl
  data8 sys_getpgid
  data8 sys_fchdir
  data8 sys_ni_syscall /* sys_bdflush */
--- linux-2.6.18/arch/x86_64/ia32/ia32entry.S.quot32 2006-09-20 07:42:06.000000000 +0400
+++ linux-2.6.18/arch/x86_64/ia32/ia32entry.S 2006-10-18 10:05:53.000000000 +0400
@@ -526,7 +526,7 @@ ia32_sys_call_table:
  .quad sys_init_module
  .quad sys_delete_module
  .quad quiet_ni_syscall  /* 130  get_kernel_syms */
- .quad sys_quotactl
+ .quad sys32_quotactl
  .quad sys_getpgid
  .quad sys_fchdir
  .quad quiet_ni_syscall /* bdflush */
--- linux-2.6.18/arch/x86_64/ia32/sys_ia32.c.quot32 2006-09-20 07:42:06.000000000 +0400
+++ linux-2.6.18/arch/x86_64/ia32/sys_ia32.c 2006-10-19 11:00:18.000000000 +0400
@@ -915,3 +915,50 @@ long sys32_lookup_dcookie(u32 addr_low,
  return sys_lookup_dcookie(((u64)addr_high << 32) | addr_low, buf, len);
 }

+asmlinkage long sys32_quotactl(unsigned int cmd, const char __user *special,
+       qid_t id, void __user *addr)
+{
+ long ret;
+ unsigned int cmds;
+ mm_segment_t old_fs;
+ struct if_dqblk dqblk;
+ struct if32_dqblk {
+  __u32 dqb_bhardlimit[2];
+  __u32 dqb_bsoftlimit[2];
+  __u32 dqb_curspace[2];
+  __u32 dqb_ihardlimit[2];
+  __u32 dqb_isoftlimit[2];
+  __u32 dqb_curinodes[2];
+  __u32 dqb_btime[2];
+  __u32 dqb_itime[2];
+  __u32 dqb_valid;
```

```
+ } dqblk32;
+
+ cmds = cmd >> SUBCMDSHIFT;
+
+ switch (cmds) {
+  case Q_GETQUOTA:
+   old_fs = get_fs();
+   set_fs(KERNEL_DS);
+   ret = sys_quotactl(cmd, special, id, &dqblk);
+   set_fs(old_fs);
+   memcpy(&dqblk32, &dqblk, sizeof(dqblk32));
+   dqblk32.dqb_valid = dqblk.dqb_valid;
+   if (copy_to_user(addr, &dqblk32, sizeof(dqblk32)))
+    return -EFAULT;
+   break;
+  case Q_SETQUOTA:
+   if (copy_from_user(&dqblk32, addr, sizeof(dqblk32)))
+    return -EFAULT;
+   memcpy(&dqblk, &dqblk32, sizeof(dqblk32));
+   dqblk.dqb_valid = dqblk32.dqb_valid;
+   old_fs = get_fs();
+   set_fs(KERNEL_DS);
+   ret = sys_quotactl(cmd, special, id, &dqblk);
+   set_fs(old_fs);
+   break;
+  default:
+   return sys_quotactl(cmd, special, id, addr);
+ }
+ return ret;
+}
```