
Subject: Re: [PATCH] diskquota: 32bit quota tools on 64bit architectures

Posted by [Andrew Morton](#) on Tue, 19 Jun 2007 19:33:16 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Mon, 18 Jun 2007 12:21:47 +0400

Vasily Tarasov <vtaras@openvz.org> wrote:

> From: Vasily Tarasov <vtaras@openvz.org>

>

> This patch should be applied after Arnd Bergmann's patch,

> that introduces new compat types:

> <http://lkml.org/lkml/2007/6/15/98>

>

> 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_ityme;

> __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 this and related situations.

>

> Signed-off-by: Vasily Tarasov <vtaras@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.

Only x86_64 and ia64 are fixed. Would it be correct to assume that the other CONFIG_COMPAT architectures also need to be fixed?

```
> --- linux-2.6.22-rc4-fixed/arch/x86_64/ia32/ia32entry.S.orig 2007-06-14 15:55:24.000000000
+0400
> +++ linux-2.6.22-rc4-fixed/arch/x86_64/ia32/ia32entry.S 2007-06-14 16:22:52.000000000
+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-2.6.22-rc4-fixed/arch/ia64/ia32/ia32_entry.S.orig 2007-06-14 15:55:24.000000000
+0400
> +++ linux-2.6.22-rc4-fixed/arch/ia64/ia32/ia32_entry.S 2007-06-14 16:22:52.000000000 +0400
> @@ -304,7 +304,7 @@ 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-2.6.22-rc4-fixed/fs/quota.c.orig 2007-06-14 15:55:26.000000000 +0400
> +++ linux-2.6.22-rc4-fixed/fs/quota.c 2007-06-18 12:00:56.000000000 +0400
> @@ -10,12 +10,14 @@
> #include <linux/slab.h>
> #include <asm/current.h>
> #include <asm/uaccess.h>
> +#include <asm/compat.h>
> #include <linux/kernel.h>
> #include <linux/security.h>
> #include <linux/syscalls.h>
> #include <linux/buffer_head.h>
> #include <linux/capability.h>
> #include <linux/quotaops.h>
> +#include <linux/types.h>
>
> /* Check validity of generic quotactl commands */
> static int generic_quotactl_valid(struct super_block *sb, int type, int cmd, qid_t id)
> @@ -384,3 +386,119 @@ asmlinkage long sys_quotactl(unsigned in
```

```

>
> return ret;
> }
> +
> + #if defined(CONFIG_X86_64) || defined(CONFIG_IA64)
> + /*
> + * This code works only for 32 bit quota tools over 64 bit OS (x86_64, ia64)
> + * and is necessary due to alignment problems.
> + */
> + struct compat_if_dqblk {
> +     compat_u64 dqb_bhardlimit;
> +     compat_u64 dqb_bsoftlimit;
> +     compat_u64 dqb_curspace;
> +     compat_u64 dqb_ihardlimit;
> +     compat_u64 dqb_isoftlimit;
> +     compat_u64 dqb_curinodes;
> +     compat_u64 dqb_btime;
> +     compat_u64 dqb_itime;
> +     compat_uint_t dqb_valid;
> + };
> +
> + /* XFS structures */
> + struct compat_fs_qfilestat {
> +     compat_u64 dqb_bhardlimit;
> +     compat_u64 qfs_nblks;
> +     compat_uint_t qfs_nextents;
> + };
> +
> + struct compat_fs_quota_stat {
> +     __s8 qs_version;
> +     __u16 qs_flags;
> +     __s8 qs_pad;
> +     struct compat_fs_qfilestat qs_uquota;
> +     struct compat_fs_qfilestat qs_gquota;
> +     compat_uint_t qs_incoredq;
> +     compat_int_t qs_btlimlimit;
> +     compat_int_t qs_itimelimit;
> +     compat_int_t qs_rtbtimelimit;
> +     __u16 qs_bwarnlimit;
> +     __u16 qs_iwarnlimit;
> + };
> +
> + asmlinkage long sys32_quotactl(unsigned int cmd, const char __user *special,
> +     qid_t id, void __user *addr)
> + {
> +     unsigned int cmds;
> +     struct if_dqblk __user *dqblk;
> +     struct compat_if_dqblk __user *compat_dqblk;

```

```

> + struct fs_quota_stat __user *fsqstat;
> + struct compat_fs_quota_stat __user *compat_fsqstat;
> + compat_uint_t data;
> + u16 xdata;
> + long ret;
> +
> + cmds = cmd >> SUBCMDSHIFT;
> +
> + switch (cmds) {
> + case Q_GETQUOTA:
> +     dqblk = compat_alloc_user_space(sizeof(struct if_dqblk));
> +     compat_dqblk = addr;
> +     ret = sys_quotactl(cmd, special, id, dqblk);
> +     if (ret)
> +         break;
> +     if (copy_in_user(compat_dqblk, dqblk, sizeof(*compat_dqblk)) ||
> +         get_user(data, &dqblk->dqblk_valid) ||
> +         put_user(data, &compat_dqblk->dqblk_valid))
> +         ret = -EFAULT;
> +     break;
> + case Q_SETQUOTA:
> +     dqblk = compat_alloc_user_space(sizeof(struct if_dqblk));
> +     compat_dqblk = addr;
> +     ret = -EFAULT;
> +     if (copy_in_user(dqblk, compat_dqblk, sizeof(*compat_dqblk)) ||
> +         get_user(data, &compat_dqblk->dqblk_valid) ||
> +         put_user(data, &dqblk->dqblk_valid))
> +         break;
> +     ret = sys_quotactl(cmd, special, id, dqblk);
> +     break;
> + case Q_XGETQSTAT:
> +     fsqstat = compat_alloc_user_space(sizeof(struct fs_quota_stat));
> +     compat_fsqstat = addr;
> +     ret = sys_quotactl(cmd, special, id, fsqstat);
> +     if (ret)
> +         break;
> +     ret = -EFAULT;
> +     /* Copying qs_version, qs_flags, qs_pad */
> +     if (copy_in_user(compat_fsqstat, fsqstat,
> +         offsetof(struct compat_fs_quota_stat, qs_uquota)))
> +         break;
> +     /* Copying qs_uquota */
> +     if (copy_in_user(&compat_fsqstat->qs_uquota,
> +         &fsqstat->qs_uquota,
> +         sizeof(compat_fsqstat->qs_uquota)) ||
> +         get_user(data, &fsqstat->qs_uquota.qfs_nextents) ||
> +         put_user(data, &compat_fsqstat->qs_uquota.qfs_nextents))
> +         break;

```

```
> + /* Copying qs_gquota */
> + if (copy_in_user(&compat_fsqstat->qs_gquota,
> + &fsqstat->qs_gquota,
> + sizeof(compat_fsqstat->qs_gquota)) ||
> + get_user(data, &fsqstat->qs_gquota.qfs_nextents) ||
> + put_user(data, &compat_fsqstat->qs_gquota.qfs_nextents))
> + break;
> + /* Copying the rest */
> + if (copy_in_user(&compat_fsqstat->qs_incoredq,
> + &fsqstat->qs_incoredq,
> + sizeof(struct compat_fs_quota_stat) -
> + offsetof(struct compat_fs_quota_stat, qs_incoredq)) ||
> + get_user(xdata, &fsqstat->qs_iwarnlimit) ||
> + put_user(xdata, &compat_fsqstat->qs_iwarnlimit))
> + break;
> + ret = 0;
> + break;
> + default:
> + ret = sys_quotactl(cmd, special, id, addr);
> + }
> + return ret;
> +}
> +#endif
```
