
Subject: [PATCH] More vxdquot debugging for #341, #116, #177

Posted by [Alexey Dobriyan](#) on Mon, 13 Nov 2006 16:22:06 GMT

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

- * Extend ->origin into array for previous origin tracking.
- * Print last two origins in debugging message.
- * Also print i_mode, i_op, i_fop, ... of the offending inode.

```
diff --git a/fs/vxdquot.c b/fs/vxdquot.c
index e723ec7..c5148b8 100644
--- a/fs/vxdquot.c
+++ b/fs/vxdquot.c
@@ -549,10 +549,17 @@ static inline int vxquota_qlnk_is_empty(
    return qlnk->qmbk == __VZ_QUOTA_EMPTY;
}

+static inline void set_qlnk_origin(struct vx_quota_ilink *qlnk,
+ unsigned char origin)
+{
+ qlnk->origin[0] = qlnk->origin[1];
+ qlnk->origin[1] = origin;
+}
+
static inline void vxquota_qlnk_set_empty(struct vx_quota_ilink *qlnk)
{
    qlnk->qmbk = __VZ_QUOTA_EMPTY;
- qlnk->origin = VZ_QUOTAO_SETE;
+ set_qlnk_origin(qlnk, VZ_QUOTAO_SETE);
}

void vxquota_qlnk_init(struct vx_quota_ilink *qlnk)
@@ -560,7 +567,7 @@ void vxquota_qlnk_init(struct vx_quota_i
    memset(qlnk, 0, sizeof(*qlnk));
    INIT_LIST_HEAD(&qlnk->list);
    vxquota_qlnk_set_empty(qlnk);
- qlnk->origin = VZ_QUOTAO_INIT;
+ set_qlnk_origin(qlnk, VZ_QUOTAO_INIT);
}

void vxquota_qlnk_destroy(struct vx_quota_ilink *qlnk)
@@ -587,7 +594,7 @@ #if defined(CONFIG_VZ_QUOTA_UGID)
#endif
    if (qlnk->qmbk != NULL && qlnk->qmbk != VZ_QUOTA_BAD)
        qmbk_put(qlnk->qmbk);
- qlnk->origin = VZ_QUOTAO_DESTR;
+ set_qlnk_origin(qlnk, VZ_QUOTAO_DESTR);
}
```

```

/**
@@ -613,7 +620,7 @@ static void vzquota_qlnk_swap(struct vz_
    if (qb != __VZ_QUOTA_EMPTY && qb != VZ_QUOTA_BAD)
        list_add(&qli->list, &qb->dq_iline_list);
    INIT_LIST_HEAD(&qli->list);
- qli->origin = VZ_QUOTAO_SWAP;
+ set_qlnk_origin(qli, VZ_QUOTAO_SWAP);

    for (i = 0; i < MAXQUOTAS; i++) {
        qu = qli->qugid[i];
@@ -774,7 +781,7 @@ static void __vzquota_inode_init(struct
    vzquota_qlnk_init(INODE_QLNK(inode));
    inode->i_dquot[USRQUOTA] = (void *)~(unsigned long)NODQUOT;
}
- INODE_QLNK(inode)->origin = origin;
+ set_qlnk_origin(INODE_QLNK(inode), origin);
}

/**
@@ -789,7 +796,7 @@ static void vzquota_inode_drop(struct in
    vzquota_qlnk_init(&qlnk);
    inode_qmblock_lock(inode->i_sb);
    vzquota_qlnk_swap(&qlnk, INODE_QLNK(inode));
- INODE_QLNK(inode)->origin = VZ_QUOTAO_DRCAL;
+ set_qlnk_origin(INODE_QLNK(inode), VZ_QUOTAO_DRCAL);
    inode->i_dquot[USRQUOTA] = NODQUOT;
    inode_qmblock_unlock(inode->i_sb);
    vzquota_qlnk_destroy(&qlnk);
@@ -816,9 +823,10 @@ static int vzquota_inode_qmblock_set(struc
    struct vz_quota_iline *qlnk)
{
    if (qmblock == NULL) {
- printk(KERN_ERR "VZDQ: NULL in set, "
- "orig %u, dev %s, inode %lu, fs %s\n",
- INODE_QLNK(inode)->origin,
+ printk(KERN_ERR "VZDQ: NULL in set, orig {%u, %u}, "
+ "dev %s, inode %lu, fs %s\n",
+ INODE_QLNK(inode)->origin[0],
+ INODE_QLNK(inode)->origin[1],
        inode->i_sb->s_id, inode->i_ino,
        inode->i_sb->s_type->name);
    printk(KERN_ERR "current %d (%s), VE %d\n",
@@ -837,7 +845,7 @@ static int vzquota_inode_qmblock_set(struc
    return 1;
}
    vzquota_qlnk_swap(qlnk, INODE_QLNK(inode));
- INODE_QLNK(inode)->origin = VZ_QUOTAO_QSET;
+ set_qlnk_origin(INODE_QLNK(inode), VZ_QUOTAO_QSET);

```

```

return 0;
}

@@ -954,7 +962,7 @@ start:
    } else
        qmbblk = VZ_QUOTA_BAD;
    if (!vzquota_inode_qmbblk_set(inode, qmbblk, &qlnk)){/* success */
-   INODE_QLNK(inode)->origin = VZ_QUOTAO_ACT;
+   set_qlnk_origin(INODE_QLNK(inode), VZ_QUOTAO_ACT);
        break;
    }
}

@@ -1003,7 +1011,7 @@ start:

    if (vzquota_inode_qmbblk_set(inode, qmbblk, qlnk))
        goto start;
-   INODE_QLNK(inode)->origin = VZ_QUOTAO_DTREE;
+   set_qlnk_origin(INODE_QLNK(inode), VZ_QUOTAO_DTREE);
}

static void vzquota_det_qmbblk_recalc(struct inode *inode,
@@ -1058,7 +1066,7 @@ start:
set:
    if (vzquota_inode_qmbblk_set(inode, qmbblk, qlnk))
        goto start;
-   INODE_QLNK(inode)->origin = VZ_QUOTAO_DET;
+   set_qlnk_origin(INODE_QLNK(inode), VZ_QUOTAO_DET);
    return;

fail:
@@ -1074,11 +1082,21 @@ fail:
    tv.tv_usec -= tvo.tv_usec;
    if (tv.tv_sec < timeout)
        goto set;
-   printk(KERN_ERR "VZDQ: %s, orig %u,"
+   printk(KERN_ERR "VZDQ: %s, orig {%u, %u},"
        " dev %s, inode %lu, fs %s\n",
-   msg, INODE_QLNK(inode)->origin,
+   msg,
+   INODE_QLNK(inode)->origin[0],
+   INODE_QLNK(inode)->origin[1],
        inode->i_sb->s_id, inode->i_ino,
        inode->i_sb->s_type->name);
+   printk(KERN_ERR "i_count %u, ", atomic_read(&inode->i_count));
+   printk(KERN_ERR "i_mode %o, ", inode->i_mode);
+   printk(KERN_ERR "i_state %lx, ", inode->i_state);
+   printk(KERN_ERR "i_flags %x\n", inode->i_flags);
+   printk(KERN_ERR "i_op %p, vfs_empty_iops %p, "

```

```

+ "i_fop %p, i_mapping %p\n",
+ inode->i_op, &vfs_empty_iops,
+ inode->i_fop, inode->i_mapping);
    if (!cnt++) {
        printk(KERN_ERR "current %d (%s), VE %d,"
            " time %ld.%06ld\n",
@@ -1304,7 +1322,7 @@ finish:
    ret = vzquota_transfer_usage(inode, mask, &qlnk);
    if (!ret) {
        vzquota_qlnk_swap(&qlnk, INODE_QLNK(inode));
- INODE_QLNK(inode)->origin = VZ_QUOTAO_TRANS;
+ set_qlnk_origin(INODE_QLNK(inode), VZ_QUOTAO_TRANS);
    }
    out_unlock:
    vzquota_data_unlock(inode, &data);
@@ -1461,7 +1479,7 @@ int vzquota_on_qmbk(struct super_block
    if (!vzquota_inode_qmbk_set(inode, qmbk, &qlnk))
        break;
}
- INODE_QLNK(inode)->origin = VZ_QUOTAO_ON;
+ set_qlnk_origin(INODE_QLNK(inode), VZ_QUOTAO_ON);
    spin_unlock(&dcache_lock);

    if (!err) {
diff --git a/include/linux/vzquota_qlnk.h b/include/linux/vzquota_qlnk.h
index 03fb192..2788c41 100644
--- a/include/linux/vzquota_qlnk.h
+++ b/include/linux/vzquota_qlnk.h
@@ -19,7 +19,7 @@ struct vz_quota_ilink {
    struct vz_quota_master *qmbk;
    struct vz_quota_uqid *quqid[MAXQUOTAS];
    struct list_head list;
- unsigned char origin;
+ unsigned char origin[2];
};

#endif /* _VZDQUOTA_QLNK_H */

```
