
Subject: [patch -mm 11/17] user namespace: add user_namespace ptr to vfsmount
Posted by [Cedric Le Goater](#) on Tue, 05 Dec 2006 10:28:03 GMT
[View Forum Message](#) <> [Reply to Message](#)

From: Serge E. Hallyn <serue@us.ibm.com>

Add user_namespace ptr to vfsmount, and define a helper to compare it to the task's user_ns.

Signed-off-by: Serge E. Hallyn <serue@us.ibm.com>

```
fs/namespace.c      | 4 ++++
include/linux/mount.h | 2 ++
include/linux/sched.h | 12 ++++++++
3 files changed, 18 insertions(+)
```

Index: 2.6.19-rc6-mm2/fs/namespace.c

=====

--- 2.6.19-rc6-mm2.orig/fs/namespace.c

+++ 2.6.19-rc6-mm2/fs/namespace.c

@@ -25,6 +25,7 @@

#include <linux/security.h>

#include <linux/mount.h>

#include <linux/ramfs.h>

+ #include <linux/user_namespace.h>

#include <asm/uaccess.h>

#include <asm/unistd.h>

#include "pnode.h"

@@ -56,6 +57,8 @@ struct vfsmount *alloc_vfsmnt(const char

struct vfsmount *mnt = kmem_cache_alloc(mnt_cache, GFP_KERNEL);

if (mnt) {

memset(mnt, 0, sizeof(struct vfsmount));

+ mnt->mnt_user_ns = current->nsproxy->user_ns;

+ get_user_ns(mnt->mnt_user_ns);

atomic_set(&mnt->mnt_count, 1);

INIT_LIST_HEAD(&mnt->mnt_hash);

INIT_LIST_HEAD(&mnt->mnt_child);

@@ -88,6 +91,7 @@ EXPORT_SYMBOL(simple_set_mnt);

void free_vfsmnt(struct vfsmount *mnt)

{

+ put_user_ns(mnt->mnt_user_ns);

kfree(mnt->mnt_devname);

kmem_cache_free(mnt_cache, mnt);

}

Index: 2.6.19-rc6-mm2/include/linux/mount.h

=====

--- 2.6.19-rc6-mm2.orig/include/linux/mount.h

```

+++ 2.6.19-rc6-mm2/include/linux/mount.h
@@ -21,6 +21,7 @@ struct super_block;
struct vfsmount;
struct dentry;
struct mnt_namespace;
+struct user_namespace;

#define MNT_NOSUID 0x01
#define MNT_NODEV 0x02
@@ -53,6 +54,7 @@ struct vfsmount {
    struct list_head mnt_slave; /* slave list entry */
    struct vfsmount *mnt_master; /* slave is on master->mnt_slave_list */
    struct mnt_namespace *mnt_ns; /* containing namespace */
+ struct user_namespace *mnt_user_ns; /* namespace for uid interpretation */
    int mnt_pinned;
};

```

Index: 2.6.19-rc6-mm2/include/linux/sched.h

```

=====
--- 2.6.19-rc6-mm2.orig/include/linux/sched.h
+++ 2.6.19-rc6-mm2/include/linux/sched.h
@@ -83,6 +83,8 @@ struct sched_param {
#include <linux/timer.h>
#include <linux/hrtimer.h>
#include <linux/task_io_accounting.h>
+#include <linux/nsproxy.h>
+#include <linux/mount.h>

#include <asm/processor.h>

@@ -1589,6 +1591,16 @@ extern int cond_resched(void);
extern int cond_resched_lock(spinlock_t * lock);
extern int cond_resched_softirq(void);

+static inline int task_mnt_same_uid(struct task_struct *tsk,
+    struct vfsmount *mnt)
+{
+ if (tsk->nsproxy == init_task.nsproxy)
+ return 1;
+ if (mnt->mnt_user_ns == tsk->nsproxy->user_ns)
+ return 1;
+ return 0;
+}
+
+/*
+ * Does a critical section need to be broken due to another
+ * task waiting?:

```

--

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>
