Subject: [patch -mm 13/17] user namespace: implement shared mounts
Posted by Cedric Le Goater on Tue, 05 Dec 2006 10:28:05 GMT
View Forum Message <> Reply to Message

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

Implement shared and private userns mounts. Private userns mounts
are ones which no process in another user namespace can use.

Here is a sample mount program which only does a bind mount of arg1
onto arg2, but making the destination a private userns mount.

```
int main(int argc, char *argv[])
{
 int type;
 if(argc != 3) {
  fprintf(stderr, "usage: %s src dest", argv[0]);
  return 1;
 }

 fprintf(stdout, "%s %s %s\n", argv[0], argv[1], argv[2]);

 type = MS_PRIV_USERNS | MS_BIND;
 setfsuid(getuid());

 if(mount(argv[1], argv[2], "none", type, "") == -1) {
  perror("mount");
  return 1;
 }
 return 0;
}
```

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

```
 fs/namespace.c        |   16 ++++++++++++----
 fs/pnode.h            |    1 +
 include/linux/fs.h    |    1 +
 include/linux/mount.h |    1 +
 include/linux/sched.h |    2 ++
 5 files changed, 17 insertions(+), 4 deletions(-)
```

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
@@ -259,6 +259,8 @@ static struct vfsmount *clone_mnt(struct
   }
   if (flag & CL_MAKE_SHARED)
```

```
    set_mnt_shared(mnt);
+  if (flag & CL_PRIV_USERNS)
+   mnt->mnt_flags |= MNT_PRIV_USERNS;

   /* stick the duplicate mount on the same expiry list
    * as the original if that was on one */
@@ -370,6 +372,7 @@ static int show_vfsmnt(struct seq_file *
   { MNT_NOSUID, ",nosuid" },
   { MNT_NODEV, ",nodev" },
   { MNT_NOEXEC, ",noexec" },
+  { MNT_PRIV_USERNS, ",privuserns" },
   { MNT_NOATIME, ",noatime" },
   { MNT_NODIRATIME, ",nodiratime" },
   { 0, NULL }
@@ -901,11 +904,14 @@ static int do_change_type(struct nameida
 /*
  * do loopback mount.
  */
-static int do_loopback(struct nameidata *nd, char *old_name, int recurse)
+static int do_loopback(struct nameidata *nd, char *old_name, int recurse,
+       int uidns_share)
 {
  struct nameidata old_nd;
  struct vfsmount *mnt = NULL;
  int err = mount_is_safe(nd);
+ int flag = (uidns_share ? CL_PRIV_USERNS : 0);
+
  if (err)
   return err;
  if (!old_name || !*old_name)
@@ -924,9 +930,9 @@ static int do_loopback(struct nameidata

  err = -ENOMEM;
  if (recurse)
-  mnt = copy_tree(old_nd.mnt, old_nd.dentry, 0);
+  mnt = copy_tree(old_nd.mnt, old_nd.dentry, flag);
  else
-  mnt = clone_mnt(old_nd.mnt, old_nd.dentry, 0);
+  mnt = clone_mnt(old_nd.mnt, old_nd.dentry, flag);

  if (!mnt)
   goto out;
@@ -1409,6 +1415,8 @@ long do_mount(char *dev_name, char *dir_
   mnt_flags |= MNT_NOATIME;
  if (flags & MS_NODIRATIME)
   mnt_flags |= MNT_NODIRATIME;
+ if (flags & MS_PRIV_USERNS)
+  mnt_flags |= MNT_PRIV_USERNS;
```

```
  flags &= ~(MS_NOSUID | MS_NOEXEC | MS_NODEV | MS_ACTIVE |
       MS_NOATIME | MS_NODIRATIME);
@@ -1426,7 +1434,7 @@ long do_mount(char *dev_name, char *dir_
    retval = do_remount(&nd, flags & ~MS_REMOUNT, mnt_flags,
           data_page);
   else if (flags & MS_BIND)
-  retval = do_loopback(&nd, dev_name, flags & MS_REC);
+  retval = do_loopback(&nd, dev_name, flags & MS_REC, flags & MS_PRIV_USERNS);
   else if (flags & (MS_SHARED | MS_PRIVATE | MS_SLAVE | MS_UNBINDABLE))
    retval = do_change_type(&nd, flags);
   else if (flags & MS_MOVE)
Index: 2.6.19-rc6-mm2/fs/pnode.h
===================================================================
--- 2.6.19-rc6-mm2.orig/fs/pnode.h
+++ 2.6.19-rc6-mm2/fs/pnode.h
@@ -22,6 +22,7 @@
 #define CL_COPY_ALL   0x04
 #define CL_MAKE_SHARED    0x08
 #define CL_PROPAGATION    0x10
+#define CL_PRIV_USERNS    0x20

 static inline void set_mnt_shared(struct vfsmount *mnt)
 {
Index: 2.6.19-rc6-mm2/include/linux/fs.h
===================================================================
--- 2.6.19-rc6-mm2.orig/include/linux/fs.h
+++ 2.6.19-rc6-mm2/include/linux/fs.h
@@ -120,6 +120,7 @@ extern int dir_notify_enable;
 #define MS_PRIVATE (1<<18) /* change to private */
 #define MS_SLAVE (1<<19) /* change to slave */
 #define MS_SHARED (1<<20) /* change to shared */
+#define MS_PRIV_USERNS (1<<21) /* compare user namespaces for permission */
 #define MS_ACTIVE (1<<30)
 #define MS_NOUSER (1<<31)

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
@@ -34,6 +34,7 @@ struct user_namespace;
 #define MNT_SHARED 0x1000 /* if the vfsmount is a shared mount */
 #define MNT_UNBINDABLE 0x2000 /* if the vfsmount is a unbindable mount */
 #define MNT_PNODE_MASK 0x3000 /* propogation flag mask */
+#define MNT_PRIV_USERNS 0x4000  /* compare user namespaces for permission */

 struct vfsmount {
   struct list_head mnt_hash;
```

```
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
@@ -1596,6 +1596,8 @@ static inline int task_mnt_same_uid(stru
 {
  if (tsk->nsproxy == init_task.nsproxy)
   return 1;
+ if (!(mnt->mnt_flags & MNT_PRIV_USERNS))
+  return 1;
  if (mnt->mnt_user_ns == tsk->nsproxy->user_ns)
   return 1;
  return 0;

--
```