Subject: Re: [patch 5/9] unprivileged mounts: allow unprivileged bind mounts
Posted by Karel Zak on Wed, 09 Jan 2008 13:25:45 GMT
View Forum Message <> Reply to Message

On Wed, Jan 09, 2008 at 01:45:09PM +0100, Jan Engelhardt wrote:
>
> On Jan 8 2008 20:08, Miklos Szeredi wrote:
> >> On Tue, 2008-01-08 at 12:35 +0100, Miklos Szeredi wrote:
> >> > +static int reserve_user_mount(void)
> >> > +{
> >> > +	int err = 0;
> >> > +
> >> > +	spin_lock(&vfsmount_lock);
> >> > +	if (nr_user_mounts >= max_user_mounts && !capable(CAP_SYS_ADMIN))
> >> > +		err = -EPERM;
> >> > +	else
> >> > +		nr_user_mounts++;
> >> > +	spin_unlock(&vfsmount_lock);
> >> > +	return err;
> >> > +}
> >>
> >> Would -ENOSPC or -ENOMEM be a more descriptive error here?
> >
> >The logic behind EPERM, is that this failure is only for unprivileged
> >callers. ENOMEM is too specifically about OOM. It could be changed
> >to ENOSPC, ENFILE, EMFILE, or it could remain EPERM. What do others
> >think?
>
> ENOSPC: No space remaining on device => 'wth'.
> ENOMEM: I usually think of a userspace OOM (e.g. malloc'ed out all of your
> 32-bit address space on 32-bit processes)
> EMFILE: "Too many open files"
> ENFILE: "Too many open files in system".
>
> ENFILE seems like a temporary winner among these four.

I see "EMFILE", it's still supported by the latest mount(8).

> Back in the old days, when the number of mounts was limited in Linux,
> what error value did it return? That one could be used.

Copy & past from mount-0.99.2:

```
/* Mount failed, complain, but don't die.  */
switch (mnt_err)
  {
  case EPERM:
    if (geteuid() == 0)
```

```
 error ("mount: mount point %s is not a directory", node);
      else
 error ("mount: must be superuser to use mount");
      break;
    case EBUSY:
      error ("mount: wrong fs type, %s already mounted, %s busy, "
 "or other error", spec, node);
      break;
    case ENOENT:
      error ("mount: mount point %s does not exist", node); break;
    case ENOTDIR:
      error ("mount: mount point %s is not a directory", node); break;
    case EINVAL:
      error ("mount: %s not a mount point", spec); break;
    case EMFILE:
      error ("mount table full"); break;
    case EIO:
      error ("mount: %s: can't read superblock", spec); break;
    case ENODEV:
      error ("mount: fs type %s not supported by kernel", type); break;
    case ENOTBLK:
      error ("mount: %s is not a block device", spec); break;
    case ENXIO:
      error ("mount: %s is not a valid block device", spec); break;
    case EACCES:
      error ("mount: block device %s is not permitted on its filesystem", spec);
      break;
    default:
      error ("mount: %s", strerror (mnt_err)); break;
    }


  Karel

--
 Karel Zak  <kzak@redhat.com>
```