Subject: Re: [PATCH 5/6] userns strict: hook ext2
Posted by serue on Tue, 05 Jun 2007 12:37:36 GMT
View Forum Message <> Reply to Message

Quoting Pavel Emelianov (xemul@sw.ru):

```
> Serge E. Hallyn wrote:
> >>From nobody Mon Sep 17 00:00:00 2001
> > From: Serge Hallyn <serue@us.ibm.com>
> > Date: Wed, 28 Mar 2007 15:06:47 -0500
> > Subject: [PATCH 5/6] userns strict: hook ext2
> >
> > Add a user namespace pointer to the ext2 superblock and inode.
> >
> > Signed-off-by: Serge E. Hallyn <serue@us.ibm.com>
> >
> > ---
> >
> >  fs/ext2/acl.c                  |    4 +++-
> >  fs/ext2/balloc.c               |   13 ++++++++++---
> >  fs/ext2/ialloc.c               |    5 +++++
> >  fs/ext2/inode.c                |    2 ++
> >  fs/ext2/ioctl.c                |   12 ++++++++----
> >  fs/ext2/super.c                |    4 ++++
> >  fs/ext2/xattr_trusted.c        |    3 ++-
> >  include/linux/ext2_fs_sb.h     |    1 +
> >  include/linux/user_namespace.h |   16 ++++++++++++++++
> >  9 files changed, 51 insertions(+), 9 deletions(-)
> >
>
> [snip]
>
> > diff --git a/fs/ext2/ialloc.c b/fs/ext2/ialloc.c
> > index 86a2f3b..34f16ab 100644
> > --- a/fs/ext2/ialloc.c
> > +++ b/fs/ext2/ialloc.c
> > @@ -17,6 +17,7 @@ #include <linux/sched.h>
> >  #include <linux/backing-dev.h>
> >  #include <linux/buffer_head.h>
> >  #include <linux/random.h>
> > +#include <linux/user_namespace.h>
> >  #include "ext2.h"
> >  #include "xattr.h"
> >  #include "acl.h"
> > @@ -133,6 +134,9 @@ void ext2_free_inode (struct inode * ino
> >   /* Do this BEFORE marking the inode not in use or returning an error */
> >   clear_inode (inode);
> >
> > + put_user_ns(inode->i_userns);
> > + inode->i_userns = NULL;
> > +
> >   if (ino < EXT2_FIRST_INO(sb) ||
> >       ino > le32_to_cpu(es->s_inodes_count)) {
> >    ext2_error (sb, "ext2_free_inode",
> > @@ -563,6 +567,7 @@ got:
> >   sb->s_dirt = 1;
> >   mark_buffer_dirty(bh2);
> >   inode->i_uid = current->fsuid;
> > + inode->i_userns = get_task_user_ns(current);
>
```

> We have all the ext2 (and in the next patch - the ext3) inodes
> attached to a particular user and prohibit access to the inodes
> belonging to other tasks' namespaces, don't we?

No. We provide access as though they were nobody/other. So for a large
chunk of the fs that would usually be read-only access.

> If so how can we allow the admin of the node to configure the
> root of a virtual server on the fly?

See below regarding using keys to set userns on an inode, but really the
idea here is that we would use the in-kernel keyring to store access
rights to other user namespaces, rather than try to do much with the
owners of a filesystem.

So the initial userns might be the owner of all of /usr, which has 755
perms, so all namespaces get read and execute permission. Root in one
user namespace might have a key saying (init_user_ns, uid 500), meaning
that root in that user namespace would have full access in the initial
user namespace as though it were uid 500. (Which presumably is who
started the user namespace)

> [snip]
>

```
> > diff --git a/fs/ext2/super.c b/fs/ext2/super.c
> > index 932579b..75ce9e8 100644
> > --- a/fs/ext2/super.c
> > +++ b/fs/ext2/super.c
> > @@ -29,6 +29,7 @@ #include <linux/smp_lock.h>
> >  #include <linux/vfs.h>
> >  #include <linux/seq_file.h>
> >  #include <linux/mount.h>
> > +#include <linux/user_namespace.h>
> >  #include <asm/uaccess.h>
> >  #include "ext2.h"
> >  #include "xattr.h"
> > @@ -125,6 +126,7 @@ static void ext2_put_super (struct super
> >     brelse (sbi->s_group_desc[i]);
> >   kfree(sbi->s_group_desc);
> >   kfree(sbi->s_debts);
> > + put_user_ns(sbi->s_resuidns);
> >   percpu_counter_destroy(&sbi->s_freeblocks_counter);
> >   percpu_counter_destroy(&sbi->s_freeinodes_counter);
> >   percpu_counter_destroy(&sbi->s_dirs_counter);
> > @@ -742,6 +744,7 @@ #endif
> >
> >   sbi->s_resuid = le16_to_cpu(es->s_def_resuid);
> >   sbi->s_resgid = le16_to_cpu(es->s_def_resgid);
> > + sbi->s_resuidns = get_task_user_ns(current);
> >
> >   set_opt(sbi->s_mount_opt, RESERVATION);
> >
> > @@ -990,6 +993,7 @@ failed_mount_group_desc:
> >   kfree(sbi->s_group_desc);
> >   kfree(sbi->s_debts);
> >  failed_mount:
> > + put_user_ns(sbi->s_resuidns);
> >   brelse(bh);
> >  failed_sbi:
> >   sb->s_fs_info = NULL;
>
> If we have a super block attached to a namespace, why not attach
> the inodes to the same namespace, not to the task opening the inode?
```

Well shoot - that is what I intended to do, and had done in some version. Must have gotten patches mixed up.

Yes, like you suggest, for "mostly userns-ignorant" filesystems that was my intent. userns-savvy filesystems could of course parse some userns xattr to set a userns. Not 100% sure how we'd do that, but I suppose when a userns started up it could register some secret key which is then placed in a table next to the userns *, and that key is what is stored in the xattr.

-serge