Subject: [patch 08/10] unprivileged mounts: make fuse safe
Posted by Miklos Szeredi on Wed, 16 Jan 2008 12:31:55 GMT
View Forum Message <> Reply to Message

From: Miklos Szeredi <mszeredi@suse.cz>

Don't require the "user_id=" and "group_id=" options for unprivileged mounts,
but if they are present, verify them for sanity.

Disallow the "allow_other" option for unprivileged mounts.

FUSE was designed from the beginning to be safe for unprivileged
users. This has also been verified in practice over many years, with
some distributions enabling unprivileged FUSE mounts by default.

However there are some properties of FUSE, that could make it unsafe
for certain situations (e.g. multiuser system with untrusted users):

- It is not always possible to use kill(2) (not even with SIGKILL) to
  terminate a process using a FUSE filesystem. However it is
  possible to use any of the following instead:
    o kill the filesystem daemon
    o use forced umounting
    o use the "fusectl" control filesystem

- As a special case of the above, killing a self-deadlocked FUSE
  process is not possible, and even killall5 will not terminate it.

- Due to the design of the process freezer, a hanging (due to network
  problems, etc) or malicious filesystem may prevent suspending to
  ram or hibernation to succeed. This is not actually unique to
  FUSE, as any hanging network filesystem will have the same affect.

If the above could pose a threat to the system, it is recommended,
that the '/proc/sys/fs/types/fuse/safe' sysctl tunable is not turned
on, and/or '/dev/fuse' is not made world-readable and writable.

Signed-off-by: Miklos Szeredi <mszeredi@suse.cz>

Index: linux/fs/fuse/inode.c
===
--- linux.orig/fs/fuse/inode.c 2008-01-16 13:24:52.000000000 +0100
+++ linux/fs/fuse/inode.c 2008-01-16 13:25:10.000000000 +0100
@@ -357,6 +357,19 @@ static int parse_fuse_opt(char *opt, str
  d->max_read = ~0;
  d->blksize = 512;

```
+ /*
+  * For unprivileged mounts use current uid/gid.  Still allow
+  * "user_id" and "group_id" options for compatibility, but
+  * only if they match these values.
+  */
+ if (!capable(CAP_SYS_ADMIN)) {
+  d->user_id = current->uid;
+  d->user_id_present = 1;
+  d->group_id = current->gid;
+  d->group_id_present = 1;
+
+ }
+
  while ((p = strsep(&opt, ",")) != NULL) {
   int token;
   int value;
@@ -385,6 +398,8 @@ static int parse_fuse_opt(char *opt, str
   case OPT_USER_ID:
    if (match_int(&args[0], &value))
     return 0;
+   if (d->user_id_present && d->user_id != value)
+    return 0;
    d->user_id = value;
    d->user_id_present = 1;
    break;
@@ -392,6 +407,8 @@ static int parse_fuse_opt(char *opt, str
   case OPT_GROUP_ID:
    if (match_int(&args[0], &value))
     return 0;
+   if (d->group_id_present && d->group_id != value)
+    return 0;
    d->group_id = value;
    d->group_id_present = 1;
    break;
@@ -596,6 +613,10 @@ static int fuse_fill_super(struct super_
  if (!parse_fuse_opt((char *) data, &d, is_bdev))
   return -EINVAL;

+ /* This is a privileged option */
+ if ((d.flags & FUSE_ALLOW_OTHER) && !capable(CAP_SYS_ADMIN))
+  return -EPERM;
+
  if (is_bdev) {
 #ifdef CONFIG_BLOCK
   if (!sb_set_blocksize(sb, d.blksize))

--
```

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers