SISTEMAS OPERATIVOS

Recopilacion de conceptos clave

Chequeo de permisos

Del libro Kerrisk, "The Linux Programming Interface", pag. 297 y siguientes.

The rules applied by the kernel when checking permissions are as follows:

1. If the process is privileged, all access is granted.
2. If the effective user ID of the process is the same as the user ID (owner) of the file, then access is granted according to the *owner* permissions on the file. For example, read access is granted if the owner-read permission bit is turned on in the file permissions mask; otherwise, read access is denied.
3. If the effective group ID of the process or any of the process supplementary group IDs matches the group ID (group owner) of the file, then access is granted according to the *group* permissions on the file.
4. Otherwise, access is granted according to the *other* permissions on the file.

> In the kernel code, the above tests are actually constructed so that the test to see whether a process is privileged is performed only if the process is not granted the permissions it needs via one of the other tests. This is done to avoid unnecessarily setting the ASU process accounting flag, which indicates that the process made use of superuser privileges (Section 28.1).

The checks against owner, group, and other permissions are done in order, and checking stops as soon as the applicable rule is found. This can have an unexpected consequence: if, for example, the permissions for group exceed those of owner, then the owner will actually have fewer permissions on the file than members of the file's group, as illustrated by the following example:

```
$ echo 'Hello world' > a.txt
$ ls -l a.txt
-rw-r--r--   1 mtk    users    12 Jun 18 12:26 a.txt
$ chmod u-rw a.txt              Remove read and write permission from owner
$ ls -l a.txt
----r--r--   1 mtk    users    12 Jun 18 12:26 a.txt
$ cat a.txt
cat: a.txt: Permission denied    Owner can no longer read file
$ su avr                         Become someone else...
Password:
$ groups                         who is in the group owning the file...
users staff teach cs
$ cat a.txt                      and thus can read the file
Hello world
```

Similar remarks apply if other grants more permissions than owner or group.

Since file permissions and ownership information are maintained within a file i-node, all filenames (links) that refer to the same i-node share this information.

Linux 2.6 provides access control lists (ACLs), which make it possible to define file permissions on a per-user and per-group basis. If a file has an ACL, then a modified version of the above algorithm is used. We describe ACLs in Chapter 17.

Permission checking for privileged processes

Above, we said that if a process is privileged, all access is granted when checking permissions. We need to add one proviso to this statement. For a file that is not a directory, Linux grants execute permission to a privileged process only if that permission is granted to at least one of the permission categories for the file. On some