## 2.4. Copying, Owning, and the Attenuation of Privilege

Two specific rights are worth discussing. The first augments existing rights and is called the **copy flag**; the second is the **own right**. Both of these rights are related to the principle of attenuation of privilege, which essentially says that a subject may not give away rights it does not possess.

### 2.4.1. Copy Right

The **copy right** (often called the **grant right**) allows the possessor to grant rights to another. By the principle of attenuation, only those rights the grantor possesses may be copied. Whether the copier must surrender the right, or can simply pass it on, is specific to the system being modeled. This right is often considered a flag attached to other rights; in this case, it is known as the **copy flag**.

EXAMPLE: In Windows NT, the copy flag corresponds to the "P" (change permission) right.

EXAMPLE: System R is a relational database developed by the IBM Corporation. Its authorization model [337, 426] takes the database tables as objects to be protected. Each table is a separate object, even if the same records are used to construct the table (meaning that two different views of the same records are treated as two separate objects). The users who access the tables are the subjects. The database rights are **read** entries, which define new views on an existing table; **insert**, **delete**, and **update** entries in a table; and **drop** (to delete a table). Associated with each right is a **grant option**; if it is set, the possessor of the privilege can grant it to another. Here, the grant option corresponds to a copy flag.

EXAMPLE: Let **c** be the copy right, and suppose a subject **p** has **r** rights over an object **f**. Then the following command allows **p** to copy **r** over **f** to another subject **q** only if **p** has a copy right over **f**.

```
command grant•r(p,f,q)
  if r in a[p,f] and c in a[p,f]
  then
    enter r into a[q,f];
end
```

EXAMPLE: If **p** does not have **c** rights over **f**, this command will not copy the **r** rights to **q**.

### 2.4.2. Own Right

The **own right** is a special right that enables possessors to add or delete privileges for themselves. It also allows the possessor to grant rights to others, although to whom they can be granted may be system- or implementation-dependent. The owner of an object is usually the subject that created the object or a subject to which the creator gave ownership.

EXAMPLE: On UNIX systems, the owner may use the **chown**(1) command to change the permissions that others have over an object. The semantics of delegation of owner ship vary among different versions of UNIX systems. On some versions, the owner cannot give ownership to another user, whereas on other versions, the owner can do so. In this case, the object cannot be later reclaimed. All power passes to the new owner.

### 2.4.3. Principle of Attenuation of Privilege

If a subject does not possess a right over an object, it should not be able to give that right to another subject. For example, if Matt cannot read the file **xyzzy**, he should not be able to grant Holly the right to read that file. This is a consequence of the principle of attenuation of privilege [280].

**Principle of Attenuation of Privilege.** A subject may not give rights it does not possess to another.

On most systems, the owner of an object can give other subjects rights over the object whether the owner has those rights enabled or not. At first glance, this appears to violate the principle. In fact, on these systems, the owner can grant itself any right over the object owned. Then the owner can grant that right to another subject. Lastly, the owner can delete the right for itself. So, this apparent exception actually conforms to the principle.

EXAMPLE: Suppose user **bishop** owns the file **/home/bishop/xyz** but does not have **read** permission on it. He can issue the following command to enable anyone to read the file, whether **bishop** can read it or not.

```
chmod go+r /home/bishop/xyz
```

If user **holly** tries to execute the same command, the system will reject the command, because **holly** cannot alter the permissions of a file she does not own. If she has **read** permission, she can copy the file and make the copy readable by everyone, thereby achieving the effect of making it world-readable.