9.3 保护机制

如果有一个清晰的模型来制定哪些事情是允许做的，以及系统的哪些资源需要保护，那么实现系统安全将会简单得多。事实上很多安全方面的工作都是试图确定这些问题，到现在为止我们也只是浅尝辄止而已。我们将着重论述几个有普遍性的模型，以及增强它们的机制。

9.3.1 保护域

计算机系统里有许多需要保护的“对象”。这些对象可以是硬件（如CPU、内存段、磁盘驱动器或打印机）或软件（如进程、文件、数据库或信号量）。

每一个对象都有用于调用的单一名称和允许进程运行的有限的一系列操作。read和write是相对文件而言的操作；up和down是相对信号量而言的操作。

显而易见的是，我们需要一种方法来禁止进程对某些未经授权的对象进行访问。而且这样的机制必须也可以在需要的时候使得受到限制的进程执行某些合法的操作子集。如进程A可以对文件F有读的权限，但没有写的权限。

为了讨论不同的保护机制，很有必要介绍一下域的概念。域（domain）是一对（对象，权限）组合。每一对组合指定一个对象和一些可在其上运行的操作子集。这里权限（right）是指对某个操作的执行许可。通常域相当于单个用户，告诉用户可以做什么不可以做什么，当然有时域的范围比用户要更广。例如，一组为某个项目编写代码的人员可能都属于相同的一个域，以便于他们都有权读写与该项目相关的文件。

对象如何分配给域由需求来确定。一个最基本的原则就是最低权限原则（Principle of Least Authority，POLA），一般而言，当每个域都拥有最少数量的对象和满足其完成工作所需的最低权限时，安全性将达到最好。

图9-4给出了3种域，每一个域里都有一些对象，每一个对象都有些不同的权限（读、写、执行）。请注意打印机1同时存在于两个域中，且在每个域中具有相同的权限。文件1同样出现在两个域中，但它在两个域中却具有不同的权限。

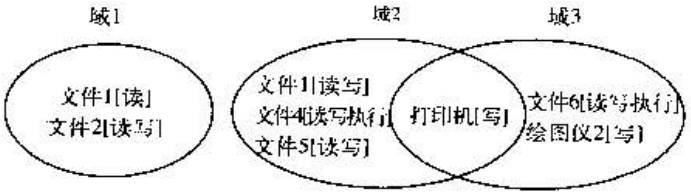

图9-4 三个保护域

任何时间，每个进程会在某个保护域中运行。换句话说，进程可以访问某些对象的集合，每个对象都有一个权限集。进程运行时也可以在不同的域之间切换。域切换的规则很大程度上与系统有关。

为了更详细地了解域，让我们来看看UNIX系统（包括Linux、FreeBSD以及一些相似的系统）。在UNIX中，进程的域是由UID和GID定义的。给定某个（UID，GID）的组合，就能够得到可以访问的所有对象列表（文件，包括由特殊文件代表的I/O设备等），以及它们是否可以读、写或执行。使用相同（UID，GID）组合的两个进程访问的是完全一致的对象组合。使用不同（UID，GID）值的进程访问的是不同的文件组合，虽然这些文件有大量的重叠。

而且，每个UNIX的进程有两个部分：用户部分和核心部分。当执行系统调用时，进程从用户部分切换到核心部分。核心部分可以访问与用户部分不同的对象集。例如，核心部分可以访问所有物理内存的页面、整个磁盘和其他所有被保护的资源。这样，系统调用就引发了域切换。

当进程把SETUID或SETGID位置于on状态时可以对文件执行exec操作，这时进程获得了新的有效UID或GID。不同的（UID，GID）组合会产生不同的文件和操作集。使用SETUID或SETGID运行程序也是一种域切换，因为可用的权限改变了。

一个很重要的问题是系统如何跟踪并确定哪个对象属于哪个域。从概念来说，至少可以预想一个大矩阵，矩阵的行代表域，列代表对象。每个方块列出对象的域包含的、可能有的权限。图9-4的矩阵如图9-5所示。有了矩阵和当前的域编号，系统就能够判断是否可以从指定的域以特定的方式访问给定的对象。

域的自我切换在矩阵模型中能够很容易实现，可以通过使用操作enter把域本身作为对象。图9-6再次显示了图9-5的矩阵，只不过把3个域当作了对象本身。域1中的进程可以切换到域2中，但是一旦切换后就不能返回。这种切换方法是在UNIX里通过执行SETUID程序实现的。不允许其他的域切换。