第5章 操作系统安全

5.1 操作系统安全概论

5.1.1 操作系统安全的基本概念

1. 安全功能与安全保证

操作系统的安全功能和安全保证是评估系统安全性的关键因素。安全功能评估系统的安全策略和机制是否符合等级要求，安全保证通过设计、实现自身安全和管理等方面的描述，确保系统提供的安全功能符合要求。安全操作系统必须同时考量安全功能和安全保证，根据安全需求确定总体安全保护等级，并满足相应的安全功能和保证要求。面向威胁的操作系统应注重实现抵御威胁的安全策略和机制，并重视安全保证。因此，设计面向威胁的安全系统时，需要综合考量安全功能和安全保证两方面。

2. 可信软件和不可信软件

一般来说，软件可以分为三大可信类别。

(1) 可信的：软件保证能安全运行，但是系统的安全仍依赖于对软件的无错操作。

(2) 良性的：软件并不确保安全运行，但由于使用了特权或对敏感信息的访问权，因而必须确信它不会蓄意地违反规则。良性软件的错误被视为偶然性的，而且这类错误不会影响系统的安全。

(3) 恶意的：软件来源不明，从安全的角度出发，该软件必须被视为恶意的，即被认为将对系统进行破坏。

安全操作系统内的可信软件通常是指首先由可信人员根据严格的标准开发出来，然后通过先进的软件工程技术(如形式化模型设计与验证)证明的软件。可信软件只是与安全相关的，并且位于安全周界内的那部分软件，这部分软件的故障会对系统安全造成不利影响。良性软件与安全无关，且位于安全周界之外，这些软件对维持系统的运行也许是必需的，但不会破坏系统的安全。

3. 主体与客体

在一个操作系统中，每一个实体组件可以是主体或客体，或者既是主体又是客体。

主体包括用户、用户组和进程等主动实体。用户分为一般用户和特殊用户(如管理

员、安全员和审计员)。用户在系统中需要唯一标识和身份验证。几乎所有事件请求都由用户发起,并通过进程处理。进程既是用户的客体,也是访问对象的主体。操作系统进程分为用户进程和系统进程。用户进程运行应用程序,满足用户计算需求;系统进程处理用户请求。

客体是被动实体。在操作系统中,客体可以是存储在特定记录介质上的数据信息(通常以文件系统格式存储),也可以是进程。进程(包括用户进程和系统进程)具有双重身份。运行的进程为某一用户服务,处理其事件请求,成为该用户的客体,或为另一进程的主体(另一进程则是该用户的客体)。

4. 安全策路和安全模型

安全策略是指有关管理、保护和发布敏感信息的法律、规定和实施细则。例如,可以将安全策略定义为:系统中的用户和信息被划分为不同的层次,一些层次的级别比另一些级别高;当且仅当主体的级别高于或等于客体的级别,主体才能读访问客体;当且仅当主体的级别低于或等于客体的级别,主体才能写访问客体。

安全模型是对安全策略所表达的安全需求的简单、抽象和无歧义的描述,它为安全策略和安全策略实现机制的关联提供了一种框架。安全模型描述了对某个安全策略需要用哪种机制来满足,而模型的实现则描述了如何把特定的机制应用于系统中,从而实现某一特定安全策略所需的安全保护。

5. 参照监视器

参照监视器是一个抽象概念,它表现的是一种思想。安德森(J. P. Anderson)把参照监视器的具体实现称为引用验证机制,它是实现参照监视器思想的硬件和软件的组合。安全策略所要求的访问判定以抽象访问控制数据库中的信息为依据,访问判定是安全策略的具体表现。访问控制数据库包含由主体访问的客体及其访问方式的信息。数据库是动态的,它随着主体和客体的产生或删除及其权限的修改而改变。参照监视器的关键需求是控制从主体到客体的每一次访问,并将重要的安全事件存入审计文件之中。

6. 安全内核

安全内核是指系统中与安全性实现有关的部分,包括引用验证机制、访问控制机制、授权机制和授权管理机制等部分。因此一般情况下,趋向于把参照监视器的概念和安全内核方法等同起来。

安全内核方法是一种常用的建立安全操作系统的方法,可以避免通常设计中固有的安全问题。安全内核方法以指导设计和开发的一系列严格的原则为基础,能够极大地提高用户对系统安全控制的信任度。

安全内核是实现参照监视器概念的一种技术,其理论依据是:在一个大型操作系统中,只有其中的小部分软件用于安全目的。所以,在重新生成操作系统时,可使用安全相关软件来构建可信的操作系统内核,即安全内核。安全内核必须得到适当的保护,防止篡改,并确保没有任何违反安全内核访问控制检查的行为。此外,安全内核应尽可能精简以方便进行正确性验证。安全内核由硬件和介于硬件与操作系统之间的一层软件组成。软件和硬件都必须处于安全周界内,而操作系统和应用程序则位于周界之外。安全周界是

划分操作系统时维护系统安全相关和无关元素之间的想象边界。

7. 可信计算基

操作系统的安全依赖于一些具体实施安全策略的可信的软件和硬件。这些软件、硬件和负责系统安全管理的人员一起组成了系统的可信计算基(Trusted Computing Base，TCB)。具体来说，可信计算基由以下几个部分组成。

(1) 操作系统的安全内核。

(2) 具有特权的程序和命令。

(3) 处理敏感信息的程序，如系统管理命令等。

(4) 与 TCB 实施安全策略有关的文件。

(5) 其他有关的固件、硬件和设备。为使系统安全，这里要求系统的固件和硬件部分必须能可信地完成它们的设计任务，其原因在于固件和硬件故障可能引起信息的丢失、改变或产生违反安全策略的事件。这也是把安全操作系统中的固件和硬件作为 TCB 的一部分来看待的理由。

(6) 负责系统管理的人员。由于系统管理员的误操作或恶意操作也会引起系统的安全性问题，因此他们也被看作是 TCB 的一部分。系统安全管理员必须经过严格的培训，并慎重地进行系统操作。

(7) 保障固件和硬件正确的程序和诊断软件。在上面所列的 TCB 的各组成部分中，可信计算基的软件部分是安全操作系统的核心内容，它完成下述工作：内核的良好定义和安全运行方式；标识系统中的每个用户；保持用户到 TCB 登录的可信路径；实施主体对客体的访问控制；维持 TCB 功能的正确性；监视和记录系统中的有关事件。

5.1.2　操作系统常见的安全保护机制

1. 认证机制

身份认证是安全操作系统应具备的最基本功能，用于审查和证实用户身份以获取系统资源或进行数据传输。身份认证机制是大多数安全机制的基础，可分为内部和外部两种。

外部身份认证验证用户是否符合其所宣称的身份，通常使用账号和口令进行验证。账号可能是公开的，如电子邮件地址，而口令则只有拥有者或管理员可更改，以确保系统无法通过绕过认证来实施攻击。操作系统机制支持这种验证，以确保系统安全。

内部身份认证机制确保某进程不能表现为除了它自身以外的进程。若没有内部验证，某用户可以创建一个看上去属于另一个用户的进程，从而即使是最高效的外部验证机制也会因为把这个用户的伪造进程看成另一个合法用户的进程而被轻易地绕过。

身份认证根据其实现方式的不同可以分为三类：单向认证(One-way Authentication)、双向认证(Two-way Authentication)和可信第三方认证(Trusted Third-way Authentication)。每一种实现方式又可以根据不同的需求采用对称密钥密码或公开密钥密码来实现。

单向认证是通信的一方认证另一方的身份，如服务器在提供用户申请的服务以前，先要认证用户是否是这项服务的合法用户，但是不需要向用户证明自己的身份。双向认证

需要通信双方互相认证对方的身份。双方都要提供用户名和密码给对方，才能通过认证。信任的第三方认证也是一种通信双方相互认证的方式，但是认证过程必须借助于一个双方都能信任的第三方，一般而言可以是政府机构或其他可信赖的机构。

2. 访问机制

访问控制技术是用来管理用户对系统资源的访问的技术，包括以下三个任务。

(1) 授权。确定可给予哪些主体访问客体的权利。

(2) 确定访问权限(读、写、执行、删除、追加等访问方式的组合)。

(3) 实施访问权限。

访问控制一般涉及自主访问控制和强制访问控制两种形式。

(1) 自主访问控制。

自主访问控制是最常用的一类访问控制机制，用来决定一个用户是否有权访问一些特定客体的一种访问约束机制。在自主访问控制机制下，文件的拥有者可以按照自己的意愿精确指定系统中的其他用户对其文件的访问权。另外，自主也指对其他具有授予某种访问权利的用户能够自主地(可能是间接地)将访问权或访问权的某个子集授予另外的用户。UNIX、LINUX 以及 Windows 等操作系统都提供自主访问控制的功能。

在自主访问控制系统中，特权用户为普通用户分配的访问权限信息主要以访问控制表(Access Control List，ACL)、访问控制能力表(Access Control Capability Lists，ACCL)、访问控制矩阵(Access Control Matrix，ACM)三种形式来存储。

ACL 是以客体为中心建立的访问权限表，其优点在于实现简单，系统为每个客体确定一个授权主体的列表，大多数 PC、服务器和主机都使用 ACL 作为访问控制的实现机制。图 5-1 为 ACL 示例。其中，*R* 表示读操作，*W* 表示写操作，*Own* 表示管理操作。例如，对客体 1，乙只有读取权限，丙则有读/写操作的权限。

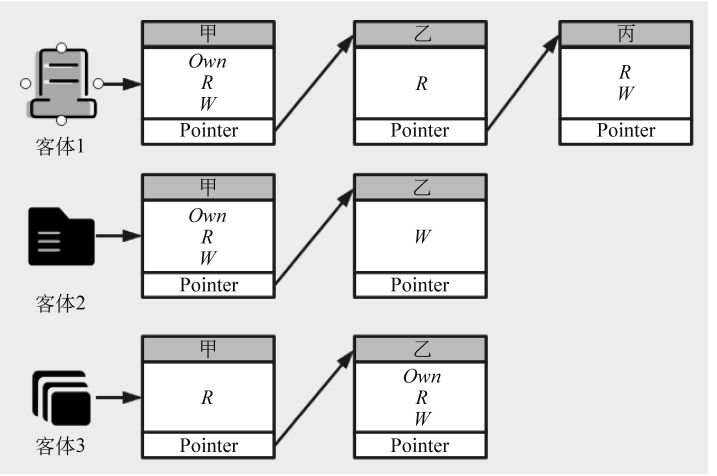

图 5-1　ACL 示例

图 5-2 为 ACCL 示例。ACCL 是以主体为中心建立的访问权限表。在这个例子中，

甲被赋予一定的访问控制能力，其具有的权限包括：对客体 1 拥有的访问权限集合为{*Own*，*R*，*W*}，对客体 2 拥有的只读权限{*R*}，对客体 3 拥有的读和写的权限{*R*，*W*}。

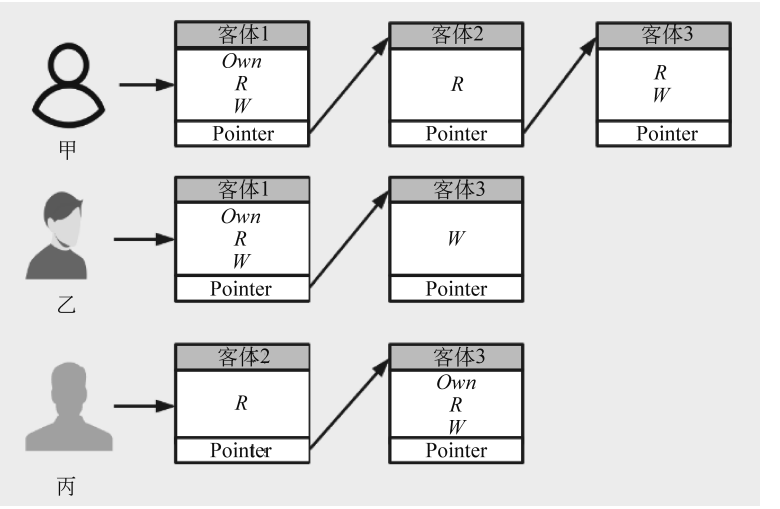

图 5-2　ACCL 示例

ACM 是通过矩阵形式表示主体用户和客体资源之间的授权关系的方法。表 5-1 为 ACM 示例。采用二维表的形式来存储访问控制策略，每一行为一个主体的访问能力描述，每一列为一个客体的访问控制描述，整个矩阵可以清晰地体现出访问控制策略。但如果主体和客体很多，那么 ACM 将会成几何级数增长，对于增长了的矩阵而言，会有大量的冗余空间，如主体丙和客体之间没有访问关系，但也存在着授权关系项。

表 5-1　ACM 示例

主　体	客　体		
	客　体　1	客　体　2	客　体　3
甲	*Own*，*R*，*W*	*R*	*R*，*W*
乙	*R*	*Own*，*R*，*W*	—
丙	*R*，*W*	—	*Own*，*R*，*W*

（2）强制访问控制。

在强制访问控制机制下，系统中的每个进程、每个文件、每个客体（消息队列、信号量集合和共享存储区）都被赋予相应的安全属性，这些安全属性是不能改变的，它由管理部门（如安全管理员）或由操作系统自动地按照严格的规则来设置，不像访问控制表由用户或程序直接或间接地修改。当一个进程访问一个客体时，调用强制访问控制机制，根据进程的安全属性和访问方式，比较进程的安全属性和客体的安全属性，从而确定是否允许进程对客体的访问。强制访问控制作为更强有力的安全保护方式，使用户不能通过意外事件和有意识的误操作逃避安全控制。因此强制访问控制用于将系统中的信息分密级和类进行管理，适用于政府部门、军事和金融等领域。

强制访问控制已经在许多基于安全内核的系统中得以实现，并转换到许多非内核化的操作系统中，这其中包括 HoneywellAf 的 Multics、DEC 的 SES/VMS 以及 Sperry 公司的 1100 操作系统。

3. 加密机制

加密是将信息编码成像密文一样难解形式的技术。加密的关键是要能高效地建立从根本上不可能被未授权用户解密的加密算法，以提高信息系统及数据的安全性和保密性，防止保密数据被窃取与泄密。图 5-3 为数据加密的一般模型。可以看出发送方意图将信息传递给接收方，为了保证安全，使用加密密钥将明文加密成密文，以密文的形式通过公共信道传输给接收方，接收方接收到密文后需要使用解密密钥将密文解密成明文，才能正确理解。第三者虽然可以在公共信道上得到密文，但不能理解其内容，即无法解密密文。在加密过程和解密过程中的两个密钥可以相同，也可以不同。在一个密码系统中除了解密密钥外，其余的加解密算法等都是公开的。一个密码体制安全的必要条件是无法穷举密钥搜索，即密钥空间非常大。

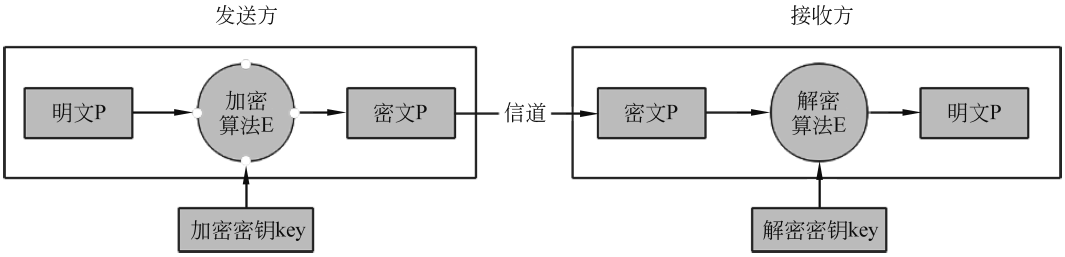

图 5-3　数据加密的一般模型

(1) 加解密算法分类。

根据不同的分类标准，密码体制有三种常用的分类：①根据密钥的使用方式不同，可分为对称密钥体制和非对称密码体制(也称为公钥密码体制)。②根据对明文和密文的处理方式和密钥的使用方式的不同，可将密码体制分成分组密码和序列密码体制。③根据加密算法在使用过程中是否变化，可将密码体制分成固定算法密码体制和变化算法密码体制。

(2) 经典密码算法。

经典密码采用的两种基本的技术，分别为代换技术和置换技术。现代加密算法大多是综合应用这两种技术来实现的，但基于的数学基础更加复杂。

代换是将明文字母替换成其他字母、数字或符号的方法。有四种类型的代换密码：①简单代换密码(或单字母密码)；②多名码代换密码；③字母代换密码；④多表代换密码。

置换技术中，明文的字母保持相同，但顺序被打乱。例如，纵行换位密码中，明文以固定的宽度水平地写在一张图表纸上，密文按垂直方向读出，解密时将密文按相同的宽度垂直地写在图表纸上，然后水平地读出明文。

4. 审计机制

一个系统的安全审计就是对系统中有关安全的活动进行记录、检查及审核。它的主要目的就是检测和阻止非法用户对计算机系统的入侵，并显示合法用法的误操作。审计作为事后追查的手段，为涉及系统安全的操作提供完整记录，帮助查询、定位和处理事故，并提供有效的追查事件和责任人的依据和支持。因此，审计是操作系统安全的一个重要方面。

审计事件是系统审计用户操作的最基本单位。系统将要求审计或可审计的用户动作都归纳成可区分、可识别、可标志和可记录的审计单位。审计机制定义了固定审计事件集，即必须审计的集合。用户可以通过设置自定义事件标准，要求审计特定事件。系统将监视用户操作，记录其落入用户事件集或系统固定审计事件集中的信息，并将其记录到审计日志文件中。否则，系统不会对该事件进行审计。

显然，审计过程会增大系统的开销。所以在实际设置中，应当选择最重要的事件进行审计，而不是设置过多的审计事件，以避免降低系统性能。审计员可以通过设置审计事件标准，确定哪些用户或事件需要审计，并将结果记录在审计日志文件中，也可以按要求打印审计结果的报表形式。

5.2　安全策略

安全策略为系统或系统集定义“安全”。安全策略在本质上可以是非形式化的，也可以是高度数学化的。本章主要介绍安全策略的基本知识，并讨论 BLP 模型、Biba 模型、Clark-Wilson 模型等几个不同类型的安全策略模型。

5.2.1　安全策略的概念与类型

安全策略是指有关管理、保护和发布敏感信息的法律、规定和实施细则，它将系统的状态分成两个集合：已授权的，即安全的状态集合；未授权的，即不安全的状态集合。

安全策略设置了可以定义安全系统的环境。在某种策略下安全的系统不一定在另一种策略下也是安全的。更确切地讲：安全系统是一种始于已授权状态但不能进入未授权状态的系统。

安全策略涉及保密性、完整性和可用性等问题的所有相关方面。

关于保密性，安全策略需要明确信息在何种状态下会泄露给未经授权的实体。这不仅包括权限的泄露，还包括没有权限的非法信息传输，即信息流。此外，安全策略必须考虑授权的动态变化，因此应包含时间元素。例如，公司的合同工在保密协议有效期内可以访问某些专利信息，一旦协议过期，他们将失去访问权限。这类安全策略通常被称为保密策略。

关于完整性，安全策略不仅要规定哪些已授权的方法可以改变信息，还要指明哪些已授权实体有权限进行这些更改。授权可以通过多种关系产生，并且可能受到外部影响的

限制。例如,在许多事务中,“职责分离”原则禁止一个实体单独完成一项事务。安全策略中描述数据可被修改的条件和方式的部分被称为完整性策略。

关于可用性,安全策略描述了必须提供的服务。它可以规定服务的参数和可访问范围,例如,浏览器可以下载网页但不能下载 Java 小程序。这种策略可能还涉及服务等级,例如服务器必须在 1 分钟内响应认证请求。可用性与服务质量(Quality of Service,QoS)直接相关。

5.2.2 保密性策略

保密性策略强调对保密性的保护。保密性策略的重要性,一方面在于它所提供的保护作用,另一方面在于它在安全概念发展中所起的重要作用。

保密性策略,也称为信息流策略,旨在防止信息被未经授权的人泄露。相比之下,信息未经授权的更改是次要问题。例如,海军必须确保他们的军舰出航日期保密。如果出航日期发生变更,这种变化应能够从系统冗余和文书工作中被察觉。然而,如果敌军获知出航日期,该军舰可能会面临被击沉的危险。由于军事通信信道中存在大量冗余,可用性并不构成主要问题。

Bell-LaPadula 模型(简称 BLP 模型)是贝尔(Bell)和拉普杜拉(LaPadula)于 1973 年提出的对应于军事类型安全密级分类的计算机操作系统模型。BLP 模型是最早也是最常使用的计算机安全模型,它影响了许多其他模型的发展,甚至对于整个计算机安全技术的发展都有着很大程度的影响。可信计算机系统评估准则(Trusted Computer System Evaluation Criteria, TCSEC)中的很多内容就是围绕着 BLP 模型设计的。

Bell-LaPadul 模型对计算机安全建模的所有策略产生了深远影响,它是首个将实际系统属性转化为规则的数学模型。基于该模型,多个标准相继形成,其中包括美国国防部的可信计算机评估标准。

5.2.3 完整性策略

对于一个信息管理系统而言,如果它管理的数据可被正常发布,那么它就是在正常工作;如果这些数据可被随机改动,那么它就不是在正常工作。因此,这里的关键问题是完整性而不是保密性。完整性策略强调的不是保密性而是完整性,因为大多数商业和产业公司更关心的是信息的准确性,而不是信息的泄露问题。商业需求与军事需求的区别在于,商业需求更强调数据的完整性。利普纳(Lipner)总结了五项需求。

(1) 用户不能自行编写程序,必须使用现有的生产程序和生产数据库。

(2) 程序员在非生产系统上开发和测试程序。如需访问实际数据,需通过特定的处理过程获取,但只能在开发系统上使用这些数据。

(3) 在开发系统上的程序安装到生产系统上时,必须遵循特定的处理过程。

(4) 需要对需求(3)中的特定处理过程进行控制和审计。

(5) 管理员和审计员必须能够访问系统状态和生成的系统日志。

在军事环境中，对特定数据类别的访问许可和安全等级决定了对信息的不同访问能力。商业公司则很少依据访问许可来授予访问权限。如果某个人需要特定信息，通常就可以获得所需的信息。虽然 Bell-LaPadula 模型可以用来对这种情况进行建模，但这需要定义大量的数据类别和安全等级，从而增加了建模的复杂性。更为棘手的问题在于如何控制这些类别和安全等级的增长。在军事环境中，安全等级和类别是集中建立的，可以严格控制安全许可的数量；而在商业公司中，这些通常是分散建立的，因而不具备相同的控制能力。

1. Biba 模型

BLP 模型提供的是保证数据保密性的安全策略，是一个军事安全模型。但在商业应用中，人们关心的常常是数据的完整性。例如，在银行的存取款业务中，交易过程的非正常中断如果得不到正确处理，将会破坏客户和银行之间的数据完整性，使一方受到损失。又如对于一个库存控制系统来说，在正常工作时，它管理的数据可以正常发布，但是如果它管理的数据可以被随意改动，就不能真实地反映库存状况。大多数的商业和产业公司更关心的是信息的准确性，而不是信息的泄露问题。完整性模型强调的就是信息的完整性问题。

Biba 模型是由毕巴(Biba)等人在 1977 年提出的第一个完整性安全模型。Biba 模型借鉴了 BLP 模型中级别的概念，定义了完整性等级，应用类似于 BLP 模型的安全规则来保护信息的完整性。

2. 克拉克-威尔逊模型

Biba 模型给出了完整性安全策略，可以有效地防止对数据的非授权修改，保证数据有效并不被完整性等级低的数据污染。但是 Biba 模型并没有考虑数据的一致性问题和事务处理本身的完整性问题。数据一致性就是数据满足给定的属性，就像在企业的财务数据中，设 D 是某账户今天到目前为止存入的金额总数，W 是今天到目前为止提取金额的总数，YB 是到昨天为止账户的金额总数，TB 是到今天到目前为止账户的金额总数，则一致性属性就是：D＋YB－W＝TB。事务处理的完整性也是很多商业应用中非常关心的，就像一个采购过程，从收到发票到确认支付，如果执行过程中缺乏一定的监督机制，就有可能会支付伪造发票。要想提高事务处理的安全性，一个可行的方法是职责分离机制，要求检验者和实现者不是一个人。这样要在事务处理过程中破坏数据，就必须最少有两个主体同时犯错误。Clark-Wilson 模型就是同时考虑了数据一致性和事务处理完整性的安全模型。

Clark-Wilson 模型是 1987 年由克拉克(Clark)和威尔逊(Wilson)共同开发的数据完整性模型。该模型的主要思想就是利用良性事务处理机制和任务分离机制来保证数据的一致性和事务处理的完整性。良性事务处理机制规定用户不能随意处理数据，即使是授权用户修改数据也要满足数据一致性的要求。任务分离机制将任务分成多个子集，由不同人完成以确保任务的安全性。为了保证主体只能以良性事务处理的方式访问客体，

Clark-Wilson 模型规定所有访问必须通过特定程序集合完成，并且这些程序必须保证有效性。Clark-Wilson 模型规定每个主体只能使用特定的程序集并分配执行权限以保证数据完整性。任何一个验证行为正确性的人不能同时也是被验证行为的执行人是任务分离机制的基本规则。

5.3 访问控制技术

5.3.1 自主访问控制

自主访问控制是应用得最为广泛的一类访问控制机制。采用该机制的系统允许客体的所有者或建立者控制和定义主体对客体的访问，即访问控制是基于拥有者的自由处理。这里的自主有两方面的含义：一方面是指用户可以自主地说明自己所拥有的资源允许系统中哪些用户以何种权限进行共享；另一方面是指对其他具有授权能力的用户，能够自主地将访问权或访问权的某个子集授予另外的用户。

需要自主访问控制保护的客体的数量取决于系统环境，几乎所有的系统在自主访问控制机制中都包括对文件、目录、IPC（Internet Process Connection）以及设备的访问控制。为了实现完备的自主访问控制，系统将访问控制矩阵相应的信息以某种形式保存在系统中。访问控制矩阵的每一行表示一个主体，每一列表示一个受保护的客体，矩阵中的元素表示该主体可以对客体进行的访问模式。目前，在操作系统中实现的自主访问控制机制一般都不是将矩阵完整地保存起来。因为矩阵中的许多元素常常为空，大量的空元素会造成存储空间的浪费，并且查找元素会耗费很多时间，造成访问控制的效率很低。实际上常常采用的方法是基于矩阵的行或列来表示访问控制信息。

1. 用户身份认证机制

Windows 系统内置支持用户身份验证（Authentication）和访问控制（Access Control）等安全机制，而身份验证是访问控制的基础。下面介绍与身份验证和访问控制相关的基本概念。

（1）用户账户（Account）。

用户账户是一种参考上下文，操作系统在这个上下文描述符中运行它的大部分代码。换种说法，所有的用户模式代码在一个用户账户的上下文中运行。即使是那些在任何人都没有登录之前就运行的代码（例如服务）也是运行在一个账户（特殊的本地系统账户 SYSTEM）的上下文中的。如果用户使用账户凭据（用户名和口令）成功通过了登录验证，之后他执行的所有命令都具有该用户的权限。于是，执行代码所进行的操作只受限于运行它的账户所具有的权限。

用户账户分为本地用户账户和域用户账户，本地用户账户用于访问本地计算机，只在本地进行身份验证，存在于本地账户数据库（Security Account Manager，SAM）中，域用户账户用于访问网络资源，存在于活动目录（Active Directory）中。

Windows 系统常用的账户如表 5-2 所示。