计算机科学 2002Vol. 29№. 12

数据库安全的研究与进展*)

Survey of Security in Database

王德强 张 锐 谢 立
(南京大学计算机系 南京 210093)
(南京大学软件新技术国家重点实验室 南京 210093)

宋 娇
(南京大学数学系 南京 210093)

Abstract Being an important part of Information System, Database Security has absorbed much attention for decades. First, we bring forth the Database Security's contents, properties of data: sensitivity, integrity and availability. Several access control models to guarantee these properties are described in this paper. Due to the challenges brought by advanced DBMSs such as OODBMS and ADBMS, we also discuss some security topics in these fields. At the end of this paper, the encryption control and inference control are reviewed together with some trend of the database security technology in the future.

Keywords Database security, DAC, MAC, RBAC, MLR, OODBS, ADBMS, Inference control, Encryption control

1 引言

计算机的产生逐渐将人类带进了信息化技术的时代,但是高风险性使得信息系统的安全性问题随着信息化进程而变得日益敏感和重要。数据库技术是组成信息系统的核心技术之一,因此,研究数据库的安全是很重要的。

数据库安全的威胁是数据的泄密和非法修改,也就是对数据的三种属性的破坏:可用性、保密性和完整性[1]。可用性是指合法用户可以随时访问他被授权可以访问的数据;保密性是指数据信息不被泄密;完整性是指数据信息不被非法修改或者破坏。

为了保证数据库中数据的安全性,有几种安全控制方法得到了应用,如存取控制、加密控制和推论控制等。采用存取控制的系统都对系统中的对象进行了两种划分,一是客体(object),数据信息的载体,如,数据库记录信息、表信息等;另一个主体(subject),存取数据信息的对象,如数据库用户。存取控制就是在主体存取客体的时候,对主体进行合法性检查,也就是对主体的权限进行检查,确保数据不被非法查询和修改。加密控制对存储的数据进行加密,防止数据被读取。推论控制防止用户通过各种推论获取非授权数据信息。

存取控制模型是研究的重点,除了自主存取控制(DAC)、强制存取控制(MAC)、基于角色的存取控制(RBAC)等传统信息安全模型的研究外,目前,多级关系数据库模型以及随之而来的多级事务问题得到了许多研究者的关注。

随着数据库技术的发展,出现了各种不同的数据库系统,如面向对象的数据库管理系统,主动数据库系统等等,这些不同的数据库系统都对安全性问题提出了不同的挑战[9]。比如,由于面向对象复杂的对象结构层次,在权限的传播控制方面就比关系型数据库复杂得多。

加密控制和推论控制也是数据库安全的两个重要方面[16]。虽然加密控制是一个独立的学科,但它在数据库方面有着自己独特的特点。

由于数据库安全涉及的范围较广,本文将简单地描述DAC、MAC、RBAC、加密控制和推论控制的基本思想;重点介绍多级数据库模型、多级事务和新一代数据库的安全问题。

2 存取控制模型

存取控制就是在主体存取客体的时候,对主体进行合法性检查,也就是对主体的权限进行检查,确保数据不被非法查询和修改。存取控制是数据库研究的重点。设计一个好的存取控制模型,需要借鉴一些基本的原则[9]:

- *最小特权和最大特权原则* 根据最小特权原则,主体在实施某一特定行动时,不应该使用不需要的权利。这样,即使这个主体被恶意的利用,破坏也只在有限的范围内进行。最大特权原则的意思则是,主体能够访问他应该访问的所有的客体(体现了数据的可用性原则)。
- *开放系统和封闭系统原则* 在一个开放的系统中,对一个用户来说,所有的存取控制,如果没有被显示的否定,都是被允许的。也就是说,缺省情况就是允许所有的存取操作。与此相反,在封闭系统中,缺省情况下所有的存取操作都是被禁止的。当一个系统的安全性占有十分重要的地位的时候,显然封闭系统优于开放系统。
- *集中管理和分散管理原则* 管理原则解决谁来负责存取控制模型中的权限的维护和管理。在集中管理原则下,单个的授权实体(或者组)负责所有的管理任务。分散管理就是管理任务由多个实体来负责。

2.1 自主存取控制模型

自主存取控制(Discretional Access Control, DAC)就是允许客体的所有者自主的将客体的存取权限授予给其它的主体。这是一种封闭的、分散的管理模型。DAC 模型很早就被提出,其中以 HRU 存取矩阵模型最为典型[10]。

DAC 中存在三个基本的元素:主体、客体和权限。权限包括读、写、删除、创建、执行等。DAC 模型就是将主体、客体和权限这三者联系起来。从形式化角度讲,我们可以使用一个权限矩阵(如表 1 所示,也称为存取矩阵)来完成这个功能。某个

*)本课题得到国家"863"资助(NO:863-301-6-4)。王德强 博士生,研究方向:信息系统的安全。张 锐 硕士研究生,研究领域:安全数据库。宋 娇 硕士研究生,研究领域:运筹学与最优化。谢 立 教授,博士生导师,研究领域:信息系统安全,网络安全等。

主体是否有某个客体的权限，都由客体的属主(owner)决定。

表 1 存取矩阵

客体 / 主体	O_1	…	O_j	…	O_m
S_1	$M[S_1,O_1]$		$M[S_1,O_j]$		$M[S_1,O_m]$
S_i	$M[S_i,O_1]$		$M[S_i,O_j]$		$M[S_i,O_m]$
S_n	$M[S_n,O_1]$		$M[S_n,O_j]$		$M[S_n,O_m]$

$M[S_i,O_j]$：主体 S_i 对客体 O_j 的权限

DAC 对主体存取信息的控制是基于对主体的鉴别和权限矩阵中的存取规则的，其优点是灵活，可适用于不同类型的信息系统，如操作系统和数据库管理系统。它的缺点也很明显，特别是对于那些安全性要求较高的系统。在 DAC 中，虽然每个存取要求是受控的，但这种由授权定义的存取限制容易被旁路，即，某个主体允许读某部分数据，当他将这部分数据读出后，无需经由数据的属主认可，就可将这些数据传递给别人，不管他是有意还是无意。由于对信息的间接传播不加控制，自主控制很容易遭到 Trojan 木马的攻击[12]。而下面描述的强制存取控制策略就可以很好的解决这个问题。

2.2 强制存取控制模型

强制存取控制分配给主体和客体不同的安全级别，安全级别集是一个有序的集合(比如，绝密(TS)＞机密(S)＞秘密(C)＞公开(P))，主体的安全级别反映了该主体不将敏感信息给不持有允许安全级别主体的置信度，客体的安全级别反映了存取在客体内信息的敏感度。我们用函数 F 来表示主体 S 或者客体 O 到安全级别 L 的映射，即 F:S∪U→L。在实施存取控制的时候，采用了两个规则：

1. 简单安全规则(也被称为读安全规则)。如果主体 S 拥有对客体 O 读的权限，那么 $F(S)\geqslant F(O)$。

2. *-规则(也被称为写安全规则)。如果主体 S 拥有对客体 O 写的权限，那么 $F(O)\geqslant F(S)$。

采用了这两个规则，信息流动就只是单向的，即信息始终从低安全等级的客体流向高安全等级的主体，信息的间接传播得以控制。

有的 MAC 模型，如 MAC 中最典型的 Bell-LaPadula 模型使用的是上面的规则。但是，从规则 2 可见，低级别的主体可以更新高级别的客体，这可能造成客体的完整性的破坏。所以，Biba 模型[13]对 BLP 模型就进行了修改，将规则 2 中的 $F(O)\geqslant F(S)$ 改为 $F(O)=F(S)$，这样就可以限制低安全等级的主体写高安全等级的客体。Dion 模型对上面两个模型进行了综合，该模型提出了一个既保护数据安全性和又保护数据完整性的强制存取策略[17]。

2.2.1 强制存取控制模型在数据库中的应用-多级数据库(Multi-level DBMS)模型

象 BLP 这样的 MAC 模型，仅仅是对信息系统安全提出的一种构架。应用于数据库中，又有新的问题出现，最典型的就是多实例问题。

对于数据库来说，主体的级别一般是指用户；而客体的粒度变化就很多，如模式、表、字段、元组、元素等。粒度的不同影响了实际系统的可用性。在多级数据模型中，SeaView 采用的多种粒度共存的模式；在 Sandhu 的 MLR 模型中，采用的是元素粒度。不管在什么粒度情况下，MLD 都有多实例的问题。

MLR 是一个比较完善的模型。在 MLR 中，表的一般模式是：$R(A_1,C_1,A_2,C_2,\cdots,A_i,C_i,\cdots,A_n,C_n,TC)$，R 表示模式名，$A_i$ 表示第 i 个字段名，C_i 表示第 i 个字段的安全级别。我们可以定义这样的一个表：Criminal(NAME，C_1，Crime，C_2，TI，C_3，TC)，TI(刑期的意思)字段元素的安全级别是 C_3；TC 表示整个元组的安全级别，它是划分不同主体视图的依据；NAME 字段是关键字。定义了 4 个安全级别，TS＞S＞C＞P。表 Criminal 内容如表 2 所示，因为这是系统实际存储的内容，所以称为系统视图。在 MAC 中，信息是单向流动的，不同的主体，它的视图(所能存取的信息)是不一样的，因此级别为 S 的主体王五是不能获取张三的信息。表 3 是王五的视图。如果王五现在想插入一条关键字为"张三"的元组。王五有这个权力，但他并不知道系统中有张三这个元组，这个时候，就产生了矛盾：系统如果忽略这个插入请求，就会让王五知道存在一个张三这样的重刑犯，这是一个典型的推论问题(参见推论控制)；如系统允许这个请求，但是，系统中不能存在两个关键字相同的元组。推论问题是不能接受的，所以，SEAVIEW 和 MLR 中都允许创建多个关键字相同的元组，这和传统 RDBMS 的语义是不一致的，这个问题就是多实例问题。

表 2 系统视图

NAME	CRIME	TI	TC
…			
张三 S	杀人 S	无期 TS	TS
李四 C	偷窃 C	一年 S	S
…			

表 3 王五的视图

NAME	CRIME	TI	TC
…			
李四 C	偷窃 C	一年 S	S
…			

解决多实例问题，就必须扩展传统数据库中关键字的概念。在 MLR 模型中，定义了外观主码 AK(apparent key)，AK 包含了一个或者多个字段。在单安全级别的系统中，AK 就是关键字。MLR 定义了完整性规则，说明了 AK 的作用：

规则 1(完整性规则)：

a) $A_i\in AK\Rightarrow t[A_i]\neq null$

b) $A_i,A_j\in AK\Rightarrow t[C_i]=t[C_i]$

c) $A_i\notin AK,A_j\in AK\Rightarrow t[C_i]\geqslant t[C_j]$　($t[C_i]$)表示字段 i 的安全级别的值。

条件 a 说明 AK 不能为空；条件 b 说明属于 AK 的字段的安全级别是一样的，之所以这样定义，是因为 AK 所对应的元组反映的是某个特定级别的所有主体的视图内容；条件 c 说明非 AK 字段的安全级别大于 AK 字段的安全级别。

MLR 通过定义下面的多实例完整性规则来解决多实例问题：

规则 2(多实例规则)：

$AK,TC\rightarrow C_i$

"→"表示函数依赖关系。这个规则的含义是，系统允许在一个关系中存在 AK 值相同的多个元组；但是，对于某个特定安全级别的用户来说，它只能接受至多一个。

另外，MLR 还定义了借数据完整性(Data-Borrow Integrity)规则。其意思是高安全级别主体可以看见低安全级别的数据，但是，它不能修改这些数据；如果它想这么做，它只能

把这些低级别信息借到它的视图空间进行修改。这个规则的目的是解决多级模型中数据的完整性问题，也就是不让低安全等级的主体写高安全级的客体。

2.2.2 *多级数据库事务的研究与发展* 事务是数据库关系中很重要的一个组成部分，多级数据库模型中，事务对安全有着很大的影响，所以，安全数据库的研究从一开始就没有停止过对事务的研究。

在多级数据库系统里，每一个事务和其它的数据一样都被分配了一个安全级别。由于事务将要对数据进行处理，所以，它应该被看成主体。它需要服从 MAC 的两个基本规则：简单安全规则（下读）和 *-规则（上写）。不过，在上面的多级模型的讨论中提到，为了解决多实例和数据完整性问题，禁止上写。所以，事务也被禁止向上写，只能同安全级别才能写。

我们尽管可以限制从高到低的、直接的非法信息流动，但是，这些限制并不能保证系统不受到损害，因为，系统可能受到一些间接的攻击，这就是隐蔽通道。不幸的是，使用传统的两阶段提交协议（2PL）和时间戳（TO）技术，很容易构造隐蔽通道。下面可以描述一个隐蔽通道的例子：

设想一个数据库，存有的数据分为两种类型：高级别和低级别的数据。所有低级别的数据对所有用户可见，而所有高级别的数据只有那些特权用户才能看见。根据安全策略规定，代表一般用户执行的事务只能存取低级的数据，代表特权用户的事务可以存取高级数据，也可以读低级数据。假设有两个协同做阴谋的事务 T_1 和 T_2，T_1 代表低级用户，T_2 由高级用户触发。某一时刻，T_2 读取一低级数据，这个时候，T_2 对这个数据进行加锁。然后，T_1 想修改这个数据，但是由于 T_2 已经对这个数据进行加锁，因此，T_1 被延迟。如果配合得当，T_1 就可以对这个延迟进行计数。这样，$T_2 \rightarrow T_1$ 就建立了一个有效的信息通道。

时间戳协议也难以避免隐蔽通道问题。假设 $ts(T_1) < ts(T_2)$，$ts(T)$ 表示 T 的时间戳。如果 T_1 试图去写 T_2 已经读过的数据 x，T_1 的写操作肯定被拒绝，因为 $rts(x) > ts(T_1)$，因此，T_1 被回滚。这样，协同做阴谋的两个事务就可以通过这个方式建成隐蔽通道。

不难发现，为了实现利用事务隐蔽通道，事务很容易被无限期延迟或被挂起。因此，有些事务面临饥饿（Starvation）的问题。多级数据库DBMS不仅要保证可串行化（Serializability），还要消除所有的非法信息通道，而且还要解决饥饿的问题。

在解决上述的事务问题上，商业数据库提出了不同的做法。Sybase Secure SQL Server 使用传统的两阶段提交协议，正如前面说的，这个提交协议是不安全的。安全 Oracle 使用了 2PL 和时间戳联合的技术。尽管这种方法是安全的，但是它不能产生一拷贝可串行化历史（one-copy serializable histories）。安全 Informix 为了解决隐蔽通道问题，允许一个低级事务对一个低级数据加写锁，不管是不是一个高级事务对这个数据加读锁了没有。这个高级事务仅仅是接到一个警告，告知它，它对那个低级事务的锁已经被破坏。这种方法尽管可解决隐蔽通道问题，但它难以保证数据的完整性。

不仅不同的事务间存在着安全问题，而且同一个多级事务内部也有安全问题，因为一个多级更新事务允许一个用户在多个级别读和写数据。为了实现这一点，就需要把这个事务拆分成多个单级事务，每一个事务符合 BLP 模型的简单安全规则和 *-规则。在 Jajodia 的文章定义了在一个多级事务中两个重要的数据依赖：一是对一个数据的低级别写，然后对这个数据进行高级别读；另一种是高级别的事务读一个数据，然后低安全级别的再读这个数据。Jajodia 进一步说明了，如果在一个事务中只有两种依赖中的一种，事物都可以安全的执行，但是，如果二者都有的话，安全问题就难以保证。有关具体的讨论可参见文[16]。

从上面对 MAC 的各种问题的讨论中，我们知道，MAC 不同于 DAC：它保证了数据信息的单向流动。在不可信代码中，即使有木马的存在，也不会造成信息的泄漏。强制存取策略的问题是它太严格，以至于在某些环境下是不能用的。例如，在一个商用系统中，并不是总能为用户分配一个安全级别，或为数据分配一个敏感级别。目前，一般的应用都是将多种策略（包括下面提高的 RBAC）结合在一起使用，南大苏富特公司研制的 Softbase 3.0 等采用的就是这样的方法。

3 基于角色的存取控制

基于角色存取控制（Role-Based Access Control，RBAC）的方法的基本思想来源于现实生活，例如，在公司里，有不同的角色、组员、经理、董事、项目主管等等，不同的角色有不同的权利和义务；同时，在公司的安全管理中，不同的角色安全功能也是不一样的。经理可以评定组员的业绩并加以更改，而组员则不能这样做。这种自然而然的思想早在上个世纪 70 年代就已经提出，但是只是到了 90 年代才引起了人们极大的兴趣，并且在这个领域取得了一定的进展[6]。

基于角色的存取控制有以下一些概念：

- 用户　就是一般意义上的人；
- 角色　用一般业务系统中的术语来说，实际上就是业务系统中的岗位、职位或者分工；
- 许可权　就是对系统中客体的某种存取权限；
- 会话　每个用户进入系统，都要和一个全局唯一的会话联系在一起，这个会话记录这个用户这次进入的一些信息，直到用户退出。

联系用户和许可权的中间是角色，见图 1。用户与特定的一个或多个角色相联系，角色与一个或多个访问许可权相联系，角色可以根据实际的工作需要生成或取消。用户每次登录系统都和一个会话联系，这个会话和一个角色相关联。如果一个用户有多个角色，他必须多次登录，由不同的会话和角色关联。除此之外，角色之间、许可权之间、角色和许可权之间定义了一些关系，比如角色间的层次性关系，也可以按需要定义各种约束，如定义出纳和会计这两个角色为互斥角色，即这两个角色不能分配给一个用户。

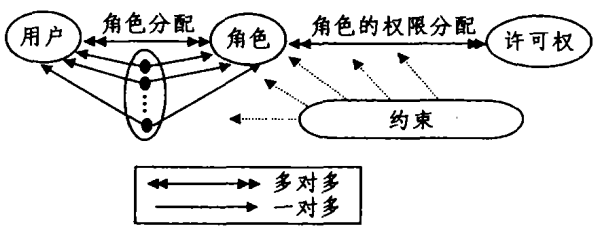

图 1　基于角色存取控制的概念实体关系图

基于角色存取控制是策略中立的，也就是说，经过一定的配置以后，它既可以实现 DAC 也可以实现 MAC。而且，因为角色的概念一般来说比较固定，所以角色的修改就少，这样，系统管理员需要做的大部分是维护角色和用户之间的映射，

管理任务减轻了不少。鉴于RBAC的这些优点，有些国际标准组织准备把角色的概念引入到标准SQL中。

但是，RBAC问题也不少。一方面，RBAC技术还不十分成熟，在角色配置的工程化、角色动态转换等方面还需要进一步研究。另一方面，采用RABC技术系统管理的前期工作是很大的。定义众多的角色和访问权限，以及它们之间的关系，而不出问题，是一件复杂的事情。

4 新一代数据库的安全与挑战

从80年代以来，DBMS领域研究和开发都集中在如何能够提供更多的功能以适应更多的应用领域。由于实际的需要，不同类型的数据库被提了出来，其中有面向对象数据库、主动数据库、联合数据库等。它们的出现既弥补了传统关系模型数据库的不足，同时也解决了一些传统数据库安全问题不能或者说不容易解决的问题，但是，它们也对数据库的安全性提出了新的要求[18]。

4.1 面向对象数据库系统的安全

面向对象数据库系统(OODBS)继承了面向对象编程的概念，如继承、多态和封装。面向对象模式最主要的吸引力是使用它可以对具有复杂结构和行为的实体进行建模。面向对象数据库系统研究也分为自主存取控制和强制存取控制。

4.1.1 OODBS中的DAC 面对对象概念的引入并不会对所有的安全概念造成影响，例如，用户的角色分配。但是，却会对局部的安全造成很大影响，比如，封装对存取模式的影响。封装使得对象间的访问只能通过方法进行，因此，除了传统的读、写和创建等存取模式外，系统还应该支持"执行方法"这个模式。一个方法可以调用其它的方法，所以，方法也应该被看成是主体。还有，继承也会对权限传播造成影响。如果我们授予给某个对象一权限，是否它的所有"子对象"都应该自动获得这些权限；或者是把这些权限传递给父对象。这些比MAC、DAC复杂得多的授权会大量的产生间接授权，OODBS应该提供策略来保证显式授权和间接授权间不发生冲突。在授权管理方面，数据对象的创建者对数据对象的存取权限传播负责。但是，问题在于，如果这个类实例并不是类的创建者自己所创建，类的创建者能否对类实例的权限传播加以限制。在粒度控制方面，也面临着类、对象实例和对象属性等几个不同粒度的选择。

在这个领域，提出的模型很多，其中的Orion，Iris比较有影响力。尽管如此，OODBS存取控制仍还处在研究的阶段，有很多的问题值得探讨。

4.1.2 OODBS中的MAC 和DAC一样，OODBS的MAC也是处在研究的阶段。MAC的研究主要集中在如何对存储在对象中数据分配安全级别。

一种简单的方式叫作单一级别对象，即对象内部，所有的对象属性和方法安全级别都是一样的。但是，这样做并不能反映现实生活，比如，文件这个对象，文件内容的安全级别显然比文件时间的安全级别高，由此，多级别对象模型被提了出来。但是，多级对象模型的存储是一个麻烦的问题，所以，如何通过单级对象存储多级对象是当前的一个研究方向。比如，有的研究者提出了将数据复制在不同的级上。这个方法的问题是，数据的一致性维护难度大。另外一种方法是，只保留一个真实的数据，需要用到该数据的时候，都使用索引。这种方法的问题是每次对象存取可能访问很多的对象，这对权限管理是一个大的负担。

在存取控制模型方面，利用对象封装的特点，Jajodia和Kogan提出了消息过滤的强制控制策略。Thurainsingham将一安全策略加到ORION模型中，提出了强制控制模型SORION模型。详细讨论可参见文[9]。

4.2 主动数据库的安全

主动的机制[18]是一种当前被看好的概念，这种概念已经被SYBASE和ORACLE数据所部分采用。主动数据库就是，除了提供传统关系数据库管理系统的所有模型外，主动数据库还允许用户定义各种各样的事件，当事件发生时，就执行用户事先定义好的代码或者规则。事件触发可以分为三种方式：

1. 立即模式 即触发这个事件的事务被挂起，直到这个事件执行完毕(也包括这个事件触发的其他事件)。

2. 延迟模式 即触发这个事件的事务不停止，直到达到提交点。在这个提交点前，这个事务触发的所有事件被排队，然后，在事务被提交前，这些触发的事件被执行。如果，所有的事件执行成功，这个事务被提交。

3. 并发模式 即被触发的事件被看成一个新的独立的事务，运行于系统之中。

显然，这些规则也是被保护的客体。谁能定义、修改或者删除规则应该在安全策略里面加以考虑。另外，在多级数据库中，对于一个规则来说，哪些事件它能看见，哪些事件它不能看见，这也是一个问题，因为触发一个规则的事务可能安全级别高，也可能安全级别低。

在自主存取模型下，主动数据库的安全研究做的不是很多。在Oracle和Sybase中，规则(在Sybase中也叫触发器)是一个特殊的对象，规则的创建者自动的成为对象的主人。它可以修改、删除该规则，但是他不能将这个规则的存取权限授予给别的用户。更多的工作是在MLS领域。文[14]对诸如规则对事件的存取关系提出了解决办法。

5 推论控制

推论问题并不仅仅只发生在数据管理系统，但在多级数据系统中，这一问题更加严重。推论问题就是用户通过他知道的信息或者低级别数据的信息，推断高级别(保密)、他不被授权访问的信息。比如，在一个上写下读的BLP系统中，用户向高级别表中插入数据，就可以判断在高级别数据中，该元组是否存在；又如，人口统计信息中，统计信息是低级别的数据，但是单个人的信息是高级别的。假设考察n个人的信息，一个用户掌握了n－1的人的统计信息，那么，和整个统计信息相比较，他就能很轻松地推断出剩下的那个人的个人信息。

多级关系数据库模型解决上面所述的第一个问题，但是，第二个问题很难解决。系统很难定义用户到底可以察看多少个人的统计信息，n－1，n－2，…或者一个百分比，这个限度是难以把握的。针对这个问题，有的研究者提出了在保证统计信息基本正确的前提下，"歪曲"统计信息的办法。这样，攻击者可以获得近似正确的统计信息，但是很难由此推断个人信息。还有一种推论问题恰好和上面第二个问题相反，也就是，单个记录信息是低级别，但是统计信息是多级别的。例如，一个国家的国防力量，可能每个排的信息不重要，但是，把所有的排加在一起，那么这就暴露了一个国家的军事实力。

从上面的讨论可以看出，推论问题难以有一致的防范方法。但也有学者提出了一些普遍的概念，对推论控制加以定义。Denning和Morgenstern定义了如下的一个公式：

$$INFER(x \rightarrow y) = \frac{H(y) - Hx(y)}{H(y)}$$

H(y)表示 y 这个信息的不确定性，也就是说，推论出 y 这个信息的不可能性的概率是多少。Hx(y)表示在已知 x 信息的条件下 y 的不确定性。那么我们可以看出，INFER(x→y)表示 x 和 y 之间的相关性，这个值在 0 到 1 之间。如果这个值等于 0，表示我们不可能从 x 的信息中推论出任何东西来。如果这个值等于 1 表示给定 x 就能推出 y。

这个公式还是很有用的，因为它告诉我们推论不是一个绝对的问题，而是一个相对的问题。但是，它存在两个很大的问题：①我们非常难以——尽管不是没有可能——决定 Hx(y)的值；②我们也没有考虑做出推论的计算复杂度。

为了说明第二点，我们可以举如下的例子：通过尝试所有的密钥，我们可以从密文中推导出明文，也就是 Hx(y)=1，这个地方 x 表示明文，y 表示密文。但是，实际情况是，我们很难这样做。有关推论控制的详细讨论可参见文[15]。

6 加密控制

一般的数据库管理系统直接将数据以原始的形式存储在系统中，如果一个知道数据文件格式的专家有意的进行攻击，后果不堪设想。理想的情况是对数据进行加密存储。

数据库数据加密有它自身的特点。首先，它不能像一般数据那样，以整个数据文件为单位，从头到尾进行加密。数据库操作的特点要求可以从数据文件抽取任何记录。也就是，加密必须解决随机的从数据库文件中某一段数据开始解密的问题。同时，加密、解密是数据调入调出的必由之路。因此，低效的加、解密算法是不能满足数据库加密要求的。另外，数据库数据一般来说会存储无限长的时间，而数据库数据又是非常的大，如果要更改密钥，代价是很大的。这就导致了密钥时间上保持很久，增加了泄密的可能性。索引字段不能加密，因为在非明文状态下，根本不可能进行索引。也不能进行关系比较，因为被比较的字段 SQL 语句无法辨认。SQL 语言中的函数将对加密数据失去作用。

因此，总的来说，数据库加密存在很多的问题。一般的解决方案是仅仅是对数据库进行部分的加密，也就是对机密的信息进行加密。有关加密控制的详细讨论，可参见文[17]。

结论 数据库管理系统的安全研究取得了不错的进展，但是，应注意的是，有些问题还需要进一步的研究和讨论。

事务方面，目前很多的事务研究还仅仅是局限于严格的 ACID 事务要求。随着一些高级应用，如办公自动化，CAD/CAM，WEB 服务等，对数据库的要求，事务模型需要加以扩充。在新一代数据库方面的研究相对来说就更少了。对面向对象数据库来说，现在还缺乏一个一般的模型来描述存取控制。各种各样的模型虽然提出了很多，但是，如何验证这些模型的正确性，却成了一个问题。验证数据库管理模型是否安全也许是一个研究的方向。目前，对审计的研究也相对较少，如何使用审计来进行入侵检测，特别是在数据库中，还有待考虑。

参 考 文 献

1 Sandhu R S, Samarati P. Access Control: Principles and Practice. IEEE Communications, vol. 32, no. 9, pp. 40～48[SS94]
2 Denning D E. Cryptography and Data Security. Addison-Wesley Publishing Company, 1983
3 Fernandez E B, Summers R C, Wood C. Database Security and Integrity. Addison-Wesley Publishing Company, 1981
4 Lampson W. A Note on the Confinement Problem. Comm. of ACM, Oct. 1973
5 National Computer Security Center. Polyinstantiation Issues in Multilevel Secure Database Management Systems: [NCSC Technical Report-005]. Volume 3/5, May 1996
6 Sandhu R S, Coyne E J, Feinstein H L, Youman C E. Role-Based Access Control Models. IEEE Computer, 1996, 29(2): 38～47
7 Denning D E, et al. The SeaView Security Model. In: Proc. IEEE Symposium on Security and Privacy, 1998. 218～233
8 Sandhu R, Chen Fang. The Multilevel Relational (MLR) Data Model. George Mason University, 1996
9 Baraani-Dastijerdi A, Pieprzyk J, Safavi-Naini R. Security In Databases: A Survey Study: [TR-96-02]. Department of Computer Science, The University of Wollongong, Wollongong, Australia, 1996
10 Harrison M, Ruzzo W, Ullman J. Protection in operating systems. Communications of the ACM, 1976, 19(8): 461～471
11 Denning D E. Database Security. Annual Reviews Inc., 1988, 3: 1～22
12 Biba K. Integrity Considerations for Secure Computer System: [ESD-TR-76-372]. Hanscom AFB, Mass. Air Force Electronic System Divison, 1977
13 Smith K. Execution Ordering for Multilevel Secure Rules. In: 4th Int. Workshop on Issues in Data Engineering, RIDE-ADS, Houston, TX, Feb. 1994. 98～104
14 National Computer Security Center. Inference and Aggregation Issues in Secure Database Management Systems: [NSCS Technical Report-005]. Volume 1/5, May 1990
15 Jajodia S, et al. Multilevel Secure Transaction Processing: Status and Prospects. In Database Security X: Status and Prospects, Chapman & Hall, London, 1997. 79～98
16 刘启原，刘怡编著. 数据库与信息系统安全. 科学出版社，2000
17 Rabitti F, Bertino E, Kim W, Woelk D. A Model of Authorization for Next-Generation Database Systems. ACM Transactions on Database Systems, 1991, 16(1): 88～131

（上接第 63 页）

代理缓存技术等。有关 Web Caching 尚存在着一些未解决问题，如动态和实时数据的缓存问题、影响置换算法性能的另外一些因素、Caching 对动态文档的作用、置换算法与检查文档一致性的算法之间的关系问题、多媒体代理缓存的特点等问题。针对这些问题，还可以做进一步的研究。

参 考 文 献

1 Danzig P. Network Appliance NetCache White Paper: Architecture and Deployment. http://www.netapp.com
2 Gwertzman J, Seltzer M. The case for geographical pushing caching. Fifth Annual Workshop on Hot Operating Systems, 1995
3 Melve I. Inter-cache communication protocols. IETF WREC Working Group Draft, 1999
4 Baeentsch M, et al. World web caching - the application level view of the Internet. IEEE Communications Magazine, 1997, 35(6)
5 Claffy K, Wessels D. ICP and the Squid Web Cache. 1997
6 Fan L, et al. Summary cache: A scalable wide-area web cache sharing protocol. Computer Sciences Dept. Univ. of Wisconsin Madison, 1998
7 Aggarwal C, Wolf J L, Yu P S. Caching on the World Wide Web. IEEE Trans. on Knowledge and Data Engineering, 1999, 11(1)
8 Zhang Z, et al, Video-Staging: Aproxy-Server-Based Approach to End-to-End Video Delivery over Wide-Area Networks. IEEE/ACM Trans. On Networking, 2000, 8(4)
9 Sen S, Rexford J, Towsley D. Proxy Prefix Caching for Multimedia Streams. IEEE Infocom 99
10 Rejaie R, Kangasharju J. Mocha: A Quality Adaptive Multimedia Proxy Cache for Internet Streaming. Proceedings of NOSSDAV, Port Jefferson, New York, June 2001