维普资讯 http://www.cqvip.com

计算机科学2002Vol. 29№. 2

增强型安全数据库系统中存储过程的安全性讨论

The Discussions on the Security of Stored Procedures in Security Subsystem of DBMS without Kernel Changed

陆晓华 吕 磊 黄 皓 谢俊元

(南京大学计算机科学与技术系 软件新技术国家重点实验室 南京210093)

Abstract Today the security subsystem of DBMS without kernel changed is of special importance. But the stored procedure, an important part of DBMS, may have some security holes. A procedure may be constructed by a group of SQL statements of different access types, each of which requires a strict check. But that is not so easy as the procedures are stored in the form of p-code in database, additional with the subsystem independent from the DBMS. As a result, we can hardly call a procedure without violating MAC rules before special measures taken. Here, we detail to make a procedure conform to MAC rules, and also, many restrictions emerge.

Keywords Security Subsystem of DBMS without Kernel Changed, Mandatory Access Control(MAC), Stored Procedure

一、引言

随着信息技术、特别是网络应用的飞速发展,对于高科技、军事、金融等需要高安全性的信息领域中的用户来说,原来的那些符合 TCSEC/TDI 的 C 类标准[2]数据库管理系统,已不能满足新的安全要求,需要提高数据库的安全性,达到以强制访问控制为基本特征的 B 类安全标准[2]。

强制访问控制的安全策略通常用贝尔-拉帕丢拉模型[1]表示。采用强制访问控制的系统中,实体分为主体与客体。数据库中的主体即为用户,在访问控制粒度为表的数据库中的客体有表、视图等。主体和客体都应具有一个安全级别;安全级别构成偏序关系。访问控制策略为"下读上写",即对已经标识过的主体和客体,高安全级别的主体可以"读"低安全级别的客体;低安全级别的主体可以"写"高安全级别的客体[2]。

在实现安全的数据库管理系统时,我们有时候把安全子系统作为一个独立的设施加在已有数据库管理系统之上,进行 MAC 检查,称之为增强型安全数据库系统。这种实现方式可以把一个普通的数据库不作较大的修改(甚至可以不知道源代码)就改造成安全的数据库。

在考虑实现安全数据库管理系统时,一方面我国自行研制的数据库管理系统本身还不够成熟,另一方面现有的大型信息系统大多已经使用国外产品;同时我们在进口国外安全数据库系统时受到诸多限制。因此我们以这种增强型安全子系统方式来提升现有数据库系统的安全性、实现安全数据库系统非常有用。

增强型安全数据库管理系统中,用户对数据库的请求(一般是 SQL 语句)将首先经过安全子系统的检查;通过安全检查的语句交给数据库去执行,否则将被拒绝。安全子系统必须有其独立的语法分析能力,以便取得每一条语句的主体、客体和操作类型,进行安全检查。

但是对于存储过程(本文中所说的存储过程,也包括了函数),将会出现一些问题。存储过程是数据库系统中一个非常有用的设施。它把控制结构和 SQL 语句无缝地集成到一起,具有强大的功能;特别是在客户机/服务器模式下,一组 SQL 语句可以捆绑在一个存储过程内作为一个单独的单位发往服务器,减少网络通讯。存储过程在创建时被编译成伪代码,并存储在数据库中;其中存取操作的权限检查也在编译时刻进行。每次调用时仅仅读取这些信息,从而提高了效率[4]。

存储过程在调用时仅仅提供了名字和一些可能的参数,安全子系统因独立于数据库底层实现,不可能把 MAC 检查的信息也编译进伪代码中;考虑到复杂性及效率,也不适宜在安全子系统中编译并保存存储过程的信息,在每次调用时再读取加以分析。这样就无法调用存储过程的相关信息,安全检查也无从做起了。

于是我们考虑把存储过程作为一个客体。从安全角度考虑,可以只关心存储过程中的 SQL 语句,而可以忽略其中的控制语句。我们赋予存储过程一个与其中的 SQL 语句有关的安全级别,使得在调用时只要取到它的安全级别就能对每一条语句进行控制。然而事情并非如此简单,它有其特殊之处:它可能由多条 SQL 语句组成,每条语句中客体的安全等级不同,存取方式也不同(可能既有"读"操作又有"写"操作),而在 MAC 中不同的存取方式对应不同的控制策略("下读上写"),因此若像普通客体一样简单地给它赋一个安全级别,在进行访问控制时必将顾此失彼,使其中的某些语句违反强制访问控制的策略。

我们的工作就是要给存储过程设计一个合理的安全级别的形式,并制订确定其安全级别的方法,使得在创建存储过程时通过安全子系统的语法分析部分分析其结构,从而确定其安全级别。这样在调用时,不必关心其中有多少条、什么存取方式的语句,直接取得其安全级别,就能进行 MAC 检查了。

下文将首先简单介绍一下强制访问控制的一些概念和规则;然后详细讨论使存储过程的使用与多级安全策略不再冲突的措施,同时我们也能看到在这种情况下对存储过程的使用有很多限制;最后是结束语,对一些问题作了简单说明。

二、强制访问控制简介

通常,安全数据库管理系统的安全子系统采用的多级安全策略模型是贝尔-拉帕丢拉模型[1],实现强制访问控制。我们简单介绍强制访问控制的内容[3]。

1. 基本概念

主体:是系统的主动元素,能采取一系列的动作;数据库中的主体即为用户。

维普资讯 http://www.cqvip.com

客体：是系统中包含信息的被动元素，是动作的对象；数据库中的客体有表、视图等。

安全级别：由密级和范畴（集）组成。密级可以比较高低，是一个全序集中的元素；范畴是范畴名组成的集合。从而安全级别组成一个偏序关系。安全级别 Sa、Sb，其密级和范畴分别为 La、Lb 和 Ca、Cb。Sa≥Sb，当且仅当 La≥Lb，Ca⊇Cb，也称为 Sa 支配 Sb；Sa≤Sb，当且仅当 La≤Lb，Ca⊆Cb，也称为 Sa 被 Sb 支配；当 Ca 和 Cb 之间没有包含与被包含关系时 Sa 和 Sb 不可比较。不可比较的安全级别形成了一个“格”，如图1所示，H≥M1≥L，H≥M2≥L，M1与 M2不可比较。可比较的安全级别形成了一个链（H-M1-L 和 H-M2-L）；而不同的链形成了格。实际上，作为偏序集合，一个系统中安全级别一般都可形成这样一个格，其中最高的安全级别 MaxL：它的密级为最高的密级、它的范畴为所有范畴名组成的集合；最低的安全级别 MinL：它的密级为最低的密级、它的范畴为空集。

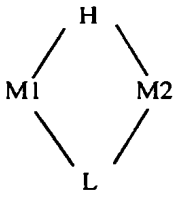

图1

标识：把一个安全级别赋予主体或客体的过程。分配给主体的安全级别也称为允许安全级（Clearance），反映了对主体不将敏感信息泄露给不持有相应允许安全级主体的置信度。分配给客体的安全级别也称为分级安全级（Classification），反映了存储在客体中的信息敏感度，也反映了未经授权向不允许存取该信息的主体泄露这些信息造成的潜在危害度。

2. 主体对客体的操作

读操作：读包含在客体中的信息。

写操作：向客体中加入信息。

读-写操作：有些操作既是读操作又是写操作，即通常所指的修改操作。

3. 规则

我们定义主体 S 的安全级别为 λ(S)，客体 O 的安全级别为 λ(O)。

（1）简单安全规则（Simple Security Property）：一个不可信主体 S 可以读一个客体 O，当且仅当 S 的安全级别支配 O 的安全级别，即 λ(S) ≥λ(O)。

这个规则的目的是防止主体直接从不允许它存取的级别的客体中存取信息。

（2）星规则（Star Property）：一个不可信主体 S 可以写一个客体 O，当且仅当 O 的安全级别支配 S 的安全级别，即 λ(S) ≤λ(O)。

这个规则的目的是防止主体从一个较高级别的客体中读取信息然后写入一个较低级别的客体中，造成高级别的信息流入低级别的客体中，从而泄露给低级别的主体。

三、对存储过程的讨论

在这一部分，我们讨论这样一个问题：在数据库系统中对于一个存储过程应该赋予它什么样的安全级别，才能使它在执行时不违反强制访问控制的两个规则。我们假定其中涉及的各条 SQL 语句中的客体已经被标识过（即已经赋予了安全级别）。

为方便起见我们把 SQL 语句分为两种操作：“读”操作和“写”操作，例如 SELECT 属于读操作，INSERT 属于写操作，UPDATE 既是读操作又是写操作（那些既不是读操作也不是写操作的语句不在我们的考虑之列）。由此我们把一个存储过程中的每条语句都抽象成二元组（obj，oper）。obj 是该语句中的客体；oper 指操作，为读或者写。若该语句涉及多个客体，则可抽象成多个操作相同客体不同的二元组。

我们以下的讨论基于这样一个一致性规则：在 MAC 中，只要一个主体能执行一个存储过程中的所有语句，那么就不应对它执行这个存储过程作任何限制；反之，只要一个主体能执行一个存储过程，那么它就必须能执行其中的任何一条语句。

据此，我们可以找出所有能操作一个存储过程中的所有语句的主体的安全级别（无论系统中是否实际存在这样一个主体），来确定这个存储过程的安全级别，这个安全级别可以使得这些主体都可以执行这个存储过程。

下面我们针对一个存储过程的二元组中客体的安全级别，分两种情形进行讨论：

1）存储过程 P 的所有二元组（obj，oper）中的客体安全级别不构成格的情形

根据安全级别的定义，如果一些安全级别不构成格，那么它们一定构成了一个链，即所有这些安全级别均可以互相比较。

我们定义：存储过程 P 的读-级别 r-level，是所有形如（obj，“read”）的二元组的客体中最高的安全级别；存储过程 P 的写-级别 w-level，是所有（客体，“写”）二元组的客体中最低的安全级别。

下面我们根据存储过程的二元组中所含操作的类型，分三种情况来讨论。

（1）存储过程 P 的二元组（obj，oper）中不含写操作只含读操作。根据简单安全规则[1]：所有安全级别支配客体中最高安全级别（即为 r-level）的主体都能操作 P 中的所有语句，即对于所有满足 λ(S) ≥ r-level 的主体 S，都能操作 P 中的所有语句。这样我们可以把存储过程 P 的读-级别 r-level 作为这个存储过程的安全级别，并且满足“向下调用”规则：主体可以调用存储过程，当且仅当主体安全级别支配这个存储过程的安全级别。

如果不这样设置，就会违反一致性规则：存储过程 P 的级别高于它的 r-level，那么就会出现一些主体 S（满足 P 的级别＞λ(S) ≥r-level）可以执行所有其中的语句但不能执行 P；P 的级别低于它的读-级别，那么就会有一些主体 S（满足 P 的级别≤λ(S)＜r-level）不能执行 P 中的某些语句却可以执行 P。

（2）存储过程 P 的二元组（obj，oper）中不含读操作只含写操作。根据星规则[1]：所有安全级别被客体中最低安全级别（即为 w-level）支配的主体都能操作 P 中的所有语句。即对于所有满足 λ(S) ≤ w-level 的主体 S，都能操作 P 中的所有语句。这样我们可以把存储过程 P 的 w-level 作为这个 P 的安全级别，并且满足“向上调用”规则：主体可以调用存储过程，当且仅当这个存储过程的安全级别支配主体安全级别。

如果不这样设置，也将违反一致性规则：存储过程 P 的级别低于它的 w-level，那么就会出现一些主体（满足 P 的级别 ＜λ(S) ≤ w-level）可以执行所有其中的语句但不能执行

维普资讯 http://www.cqvip.com

P；存储过程P的级别高于它的w-level，那么就会有一些主体（满足P的级别≥λ(S) > w-level）不能执行P中的某些语句却可以执行P，出现“向下写”的结果。

(3)存储过程P中既有读操作又有写操作。我们把所有二元组(obj,oper)分成读和写两个部分。由(1)知，对于所有读操作而言，主体S可以调用存储过程P，当且仅当λ(S) ≥ r-level；由(2)知，对于所有写操作而言，主体可以调用存储过程P，当且仅当λ(S) ≤ w-level。为了兼顾二者，我们把存储过程的级别设为一个区间，其上界UB(Upper Bound)为它的w-level，下界LB(Lower Bound)为它的r-level，其操作规则为：主体S可以调用存储过程P，当且仅当UB≥λ(S)≥LB。

对于这样配置安全级别的存储过程，我们发现其中的一些永远不能被任何主体调用。这与它的上下界即r-level与w-level的关系有关。如图2所示（图中S1,S2,S3表示三个安全级别，S1>S2>S3，“读”表示所有与读操作有关的客体级别，“写”表示所有与写操作有关的客体级别，P1、P2、P3和P4是具有这四种情况的存储过程）：

·对于存储过程P1，r-level = w-level 。这时只有具有这个安全级别的主体可以调用这个存储过程；

·对于存储过程P2，它的写-级别严格支配读-级别（这儿“严格支配”的意思是“支配并且不等”，下同），即w-level > r-level 。这时在一个安全级别区间内的所有主体S可以调用P2：w-level ≥λ(S) ≥ r-level；

·对于存储过程P3和P4，它们的写操作有关的客体安全级别不能支配所有读操作的客体级别，即w-level < r-level 。这两种存储过程永远不能被调用。

综合(1)(2)(3)可知，存储过程P的所有二元组(obj,oper)中的客体安全级别不构成格，则P的安全级别定义为一个级别区间：其上界UB为所有与写操作有关的客体的最低安全级别(w-level)，若没有写操作的客体，那么上界为所有安全级别的系统最高级别MaxL；下界LB为所有与读操作有关的客体的最高安全级别(r-level)，若没有读操作的客体，那么下界为所有安全级别的系统最低级别MinL。其调用规则为：一个主体S可以调用一个存储过程P，当且仅当S的安全级别支配P的下界并且被P的上界所支配，即UB≥λ(S)≥LB。

对于这样的定义，我们可以看到：(1)如果一个存储过程的下界严格支配它的上界（即LB > UB），那么这个存储过程就不可以被任何主体调用。(2)如果一个存储过程既没有读操作又没有写操作，则UB = MaxL，LB = MinL，那么这个存储过程可以被任何主体调用。

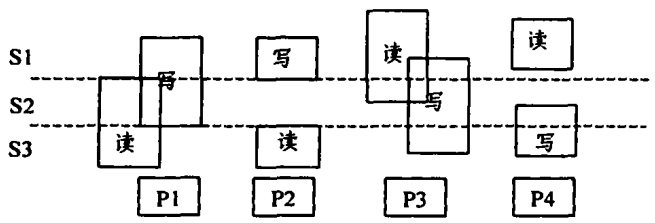

图2

2）存储过程P的所有二元组(obj,oper)中的客体安全级别构成格的情形

如前所述，安全级别形成“格”，即其中的某些级别不可比较（见图1）。从1的讨论中我们可以类似地得到：

(1)对于所有形如（客体，“读”）的二元组中的客体来说，我们要找到一个安全级别Level，使得满足λ(S) ≥ Level的所有主体S都恰好能“读”这些语句，而除此以外的主体都不能“读”所有这些客体（在1中Level即这些客体的最高级别r-level）。鉴于客体安全级别可能不可比较，我们构造一个这样的安全级别L：其密级为所有这些客体安全级别中的最高密级，范畴为所有这些客体安全级别中范畴的并集；如果没有读操作的客体，那么L=MinL。显然这个安全级别L能够满足我们的要求：它恰好支配所有读操作客体的安全级别，我们称L为所有读操作客体的安全级别的最小上限级别r-LUB (Least Upper Bound)。

(2)对于所有形如（客体，“写”）的二元组中的客体来说，我们要找到一个安全级别Level，使得满足λ(S) ≤ Level的所有主体S都恰好能“写”这些语句，而除此以外的主体都不能“写”所有这些客体（在1中Level即这些客体的最低级别w-level）。鉴于客体安全级别可能不可比较，我们构造一个这样的安全级别L：其密级为所有这些客体安全级别中的最低密级，范畴为所有这些客体安全级别中范畴的交集；如果没有写操作的客体，那么L=MaxL。显然这个安全级别能够满足我们的要求：它恰好被所有写操作的客体级别所支配，我们称这个安全级别为所有写操作客体的安全级别的最大下限级别w-GLB(Greatest Lower Bound)。

我们定义：一个存储过程的安全级别为一个区间，上界UB为写客体的最大下限级别w-GLB，下界LB为读客体的最小上限级别r-LUB。一个主体可以调用存储过程，当且仅当这个主体的安全级别支配这个存储过程的下界，并且被其上界所支配，即UB≥λ(S)≥LB。

正如1中所述，这儿也不是所有的存储过程都可以被调用。我们通过讨论读安全级别的上下界，即读操作的最小上限级别r-LUB和写操作的最大下限级别w-GLB之间的关系，可以证明：只要存储过程中存在一个操作为读的客体与一个操作为写的客体，它们的安全级别不可比较，那么这个存储过程将不可被调用。证明如下：

假定操作为读的客体安全级别Mr，操作为写的客体安全级别Mw，Mr和Mw不可比较，则它们一定位于某个格中（如图3所示）。我们根据r-LUB和w-GLB的定义可知，有：L<Mr≤r-LUB；H>Mw≥w-GLB。r-LUB在左链上半部，w-GLB在右链的下半部，如果r-LUB和w-GLB可以比较，即它们要在同一个链中，则或者r-LUB≥ H，或者w-GLB≤L。前者有r-LUB≥H>w-GLB，后者有w-GLB≤L<r-LUB。因此r-LUB和w-GLB的关系为：r-LUB和w-GLB不可比较，或者r-LUB>w-GLB。这样存储过程的上界与下界不可比较，或者上界被下界所严格支配。则不存在一个主体的安全级别可以满足这个存储过程的调用规则，即这样的存储过程不可被调用。

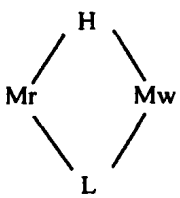

图3

那么在什么情况下既有读操作又有写操作的存储过程才可以被某些主体调用呢？（显然，只有读操作或只有写操作的

维普资讯 http://www.cqvip.com

存储过程一定可以被调用。)由上面的证明可知,所有的写操作的客体与所有读操作的客体应该都可以比较,尽管它们各自内部之间可能不可比较;又由 w-GLB≥r-LUB 可知,写操作的客体的安全级别应该支配读操作的客体的安全级别。这样我们可以确定,这样的存储过程中的客体的安全级别所形成的格如图4所示的三种情形,我们不再作进一步的说明了。

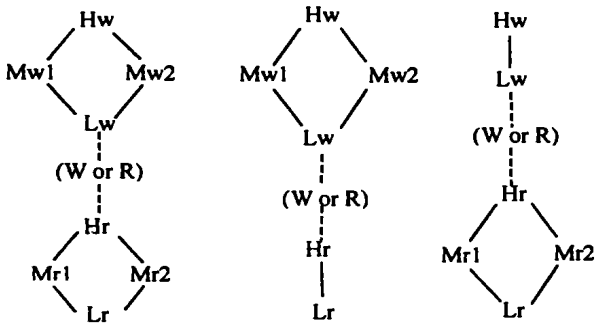

图4

综上所述,我们根据存储过程中的 SQL 语句来确定这个存储过程的安全级别,存储过程的安全级别定义为一个级别区间,凡是涉及读操作的客体的安全级别参与其下界的确定,凡是涉及写操作的客体的安全级别参与其上界的确定。考虑到可能有存储过程的嵌套,即一个存储过程的定义中又调用了存储过程,我们很自然地让子过程的下界参与父过程下界的确定,子过程的上界参与父过程上界的确定。具体地,我们把1和2统一起来可以得到如下的结论。

·一个存储过程的安全级别可以由其中的 SQL 语句来确定,SQL 语句根据存取类型分为读语句和写语句。

存储过程安全级别的下界是这样的一个安全级别:其密级为所有读语句中客体安全级别和所有子过程安全级别的下界中的最高密级,其范畴为所有读语句中客体安全级别和所有子过程安全级别的下界中的所有范畴的并集;如果不存在读语句和子过程,那么下界应该是一个最小的安全级别,即密级最低,范畴为空。

存储过程安全级别的上界是这样的一个安全级别:其密级为所有写语句中客体安全级别和所有子过程安全级别的上界中的最低密级,其范畴为所有写语句中客体安全级别和所有子过程安全级别的上界中的所有范畴的交集;如果不存在写语句和子过程,那么上界应该是一个最大的安全级别,即密级最高,范畴为所有范畴名的集合。

·存储过程调用的安全策略为:一个主体可以调用一个存储过程,当且仅当这个主体的安全级别支配这个存储过程安全级别的下界,并且被这个存储过程安全级别的上界所支配。

·一些存储过程从强制访问控制的安全角度来看是不可调用的:任何不可信主体调用它都将违反强制访问控制的安全策略。这种存储过程其安全级别有如下的特征:下界严格支配上界,或者下界与上界不可比较;实际上前者有"向上读"或"向下写"的操作,后者则从别的范畴读或向别的范畴写,都违反了强制访问控制的安全策略。这样的一个存储过程可以作为一个特洛伊木马程序。

结束语 以上我们通过考察构成存储过程中 SQL 语句的安全性,定义了它的安全级别的形式及其确定方法、调用规则,从而可以使存储过程在增强型的安全数据库系统中作为一个独立的客体被主体调用,而不违背强制访问控制的规则。

我们可以看到存储过程在进行安全检查时效率提高了:不管存储过程中有多少条 SQL 语句,只需进行一次 MAC 检查,因而平均每条语句所需代价比不用存储过程实现同样语句所需的代价要小;并且语句越多,平均代价越小。

存储过程的安全级别取决于其中的 SQL 语句的客体。从强制访问控制角度考虑,执行一个存储过程应该和逐条执行其中的所有 SQL 语句具有相同效果。但是由于可能包含了控制结构,如条件语句等,存储过程执行时可能有些 SQL 语句并不被执行,因此存储过程的每一次调用,所需要的安全级别可能不同。从存储过程的某一次调用看,我们确定的安全级别可能有过于严格的限制,但却是在保证了每一次调用都是安全的条件下限制是最小的。

需要说明的是存储过程本身并不存储信息,它不是信息的载体,因此并不是一个一般的客体。但是存储过程可以通过读、写客体来存取数据,与数据或者说"信息"密切相关,并且这种读写操作具有一定的稳定性(存储在数据库中),因此我们把它当作一个特殊的客体也是有道理的。正如我们所看到的,它的这个特殊性决定了它的安全级别的形式与普通客体有很大区别,访问控制的策略也不同。

这里我们没有考虑数据库中客体的完整性,这在伯巴模型[5]中有详细描述。事实上若需进行完整性讨论,与我们这里采用的方法完全类似。

参 考 文 献

1 Bell, La Padula. Secure Computer Systems: Mathematical Foundations and Model. M74～244. Bedfrod, Mass.: The MITRE Corporation, 1975

2 Department of Defense, Trusted Computer System Evaluation Criteria, 1985

3 刘启原,刘怡. 数据库与信息系统的安全. 科学出版社,2000

4 Scott Urman 著,译友翻译组 译. Oracle8 PL/SQL 程序设计. 机械工业出版社,1998

5 Biba K. Integrity Consideration for Secure Computer System: [ESD-TR-76-372]. Hanscom AFB, Mass. Air Force Electronic System Division, 1997