动态推理控制

顾海星[1],毛宇光[1,2]
(1.南京航空航天大学 信息科学与技术学院,江苏 南京 210016;
2.南京大学 计算机软件新技术国家重点实验室,江苏 南京 210093)

摘 要:防止未授权的用户从可读取的安全等级较低的数据中推理出安全等级较高的数据是多级关系数据库达到安全的必要保证。由于数据库中元组、属性、元素之间的相互关联性,多级关系数据库存在着推理通道。它的存在对信息的安全造成很大威胁。主要论述了多级安全数据库系统的推理通道的来源,分析了目前在多级安全数据库系统中推理问题的成果。在此基础上,提出了一种动态控制推理通道的方法并给出了相应算法。

关键词:安全数据库;推理控制;敏感信息
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2008)01-0030-03

Dynamic Inference Control

GU Hai-xing[1], MAO Yu-guang[1,2]
(1. College of Information Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China;
2. State Key Lab. for Novel Software Technology, Nanjing University, Nanjing 210093, China)

Abstract: Preventing unauthorized users from inferring higher level data from the authorized lower level data is the necessary assurance of the security of the database. Because of the relationship among the tuples, the attributes and the elements, inference becomes an important problem of the secure database. The existence of it brings hidden troubles of the information security. Mainly discusses how the inference channel comes from and analyses the method for handling this problem now. Brings forward a dynamic method and gives the algorithm.

Key words: secure database; inference control; sensitive information

0 引 言

早在1980年,数据库研究者们就已开始对安全数据库的推理控制问题进行研究。研究发现安全级较低的用户通过不同形式的推理可以访问高安全级的数据。最常见的表现形式是从一系列的查询结果中进行判断、推理。举例来说,假设项目名和支持此项目的公司这两个属性是在高安全级别上,但是,用户可以通过查询项目的会议名、参加会议的人员和与会人员的工作单位推断出项目名和支持它的公司。推理通道中还有一个比较复杂的问题是单独一个用户无法完成推理,但是大量的用户进行串谋(分工合作)后推理就变得容易了。

推理通道控制技术大致可以分为两类:静态控制和动态控制。静态控制是指在设计阶段发现推理通道,通过修改设计、提高某些敏感数据的安全级[1,2]。这种方法的缺点是可能导致给属性或原始数据分配过高的安全级,削弱数据的可用性;动态控制是指在查询期间消除推理通道,如果检测到推理通道就拒绝查询或修改查询,以保证敏感信息安全[3]。这种基于查询的推理控制代价高,且仍会导致敏感信息的泄漏(从拒绝的查询来推理敏感信息)。

文中提出了一种动态控制推理通道的方法并给出了算法。利用此算法可以很好地保护敏感信息、解决推理问题,同时又能保证高访问效率。

1 初步知识

用户对信息的访问控制是通过赋予用户和被访问信息一定权限来实现的,用户只能访问权限内的各种信息,把分配给用户和信息的各种权限称为安全级。为保护敏感数据,MAC(Mandatory Access Control)必须保证主体的安全级大于或等于客体的安全级,即 λ

收稿日期:2007-03-16
基金项目:国家重点基础研究规划项目(G1999032701)
作者简介:顾海星(1983-),男,江苏启东人,硕士研究生,研究方向为安全数据库;毛宇光,副教授,博士后,研究方向为数据库理论、特种数据库及多值逻辑。

(subjects) ≥λ(object)。这里的客体可以是表、元组、属性等,主体可以是访问客体的人或程序。

定义 1　推理通道(inference channel):InfCh(H) = {$X \in U\alpha \mid \lambda(X) < \lambda(H) \wedge \mathrm{INFER}(X \rightarrow H) > \varepsilon \vee \exists Z \in U\alpha[\lambda(Z) < \lambda(H) \wedge \mathrm{INFER}((Z \cup \{X\}) \rightarrow H) > (\mathrm{INFER}(Z \rightarrow H) + \varepsilon)]$}。其中 $U\alpha$ 表示数据集合;INFER($X \rightarrow H$) > ε 为推理函数。推理通道包含两部分内容:"或"的前半部分表示由低安全级的 X 可以推出较高安全级的 H;"或"的后半部分表示低安全级的 Z 自身并不能推出 H,但与 X 结合后,能推出较高安全级的 H[4]。

假定所有的推理通道已经在预查询阶段被识别出来。这样可以把所有的关注焦点集中到查询过程中的推理控制上。为了简单起见,文中只考虑多级安全数据库中同一安全级的用户。不同安全级用户的推理控制也是类似的。文中把存贮在数据库中的有信息的单元或者相互间的关系看作一个对象。以引言中的例子为例,"工程名"、"会议名"、"人员信息"和"公司名"都是对象。一个推理通道是指完成一个推理得到敏感信息所需的最少对象的集合。例如"工程名"、"会议名"、"人员信息"和"公司名"组成了一个长度为4的推理通道。

文中的思路来源于安全数据库分级,利用动态安全级来控制推理通道。方法可分为两个阶段:第一阶段是初始化。假设有一个推理通道的长度为 n,那么需要有 n 个密钥。也就是说密钥的个数与推理通道的长度在数值上相等。密钥的生成非常简单,不需要什么特殊的算法,因而可以保障较高的初始化速率。在初始化阶段,推理通道中的所有对象都被绑定一组密钥。这一阶段的动作只需执行一次。第二阶段就是在用户查询过程中的处理。用户不需要保存密钥,这杜绝了暴力破解密钥的可能性。处理思想是这样的:当一个密钥被用来访问某个对象后,其他的针对此对象的查询都使用同一个密钥。这通过删除与之联系的其他密钥来实现。

密钥是采用这样一种方法生成的:假设一个长度为 n 的推理通道中所有对象的最低安全级为 f,最高安全级为 s,同时假设用户的访问等级为 w,则生成的密钥前 $n-1$ 个是 f,最后一个大于 w 即可。

2　动态推理控制

本节主要描述动态安全级方法在三种不同情况下是如何解决推理控制问题的。

2.1　单推理通道

首先考虑数据库中只有一个推理通道的情况。m 标示推理通道的长度,推理通道中的对象分别为 O_1,…,O_m。U 代表数据库的某个使用者。每个 O_i 绑定的密钥集都一样,用 $K(O)$ 来表示。在这个简单模式下,密钥的总数为 m。假设推理通道中对象的最低安全级为 f,最高安全级为 s,用户的访问等级为 w,则密钥集可以为{$f, f, \cdots, f, w+1$}。

当用户尝试访问某个对象时,一个较小的密钥被选中。如果该密钥大于用户的访问等级则提示无权访问;否则,本对象其余的密钥全部删除,其余对象中均删除一个与被访问对象密钥相同的密钥。最终,有一个对象的密钥高于所有用户的访问等级,没有人可以访问它,把这个对象称之为"保留对象"[5]。一旦"保留对象"被确定,不论有多少用户进行合谋推理都是没有意义的。图 1 就是一个简单的例子。O_1, O_2, O_3 是一个推理通道中的三个对象。初始的时候每个对象都有三个密钥。当 O_1 被访问后,每个对象拥有的密钥变成了图 1 的下半部分。在这里假设访问的用户的安全级均为 C,这样当用户访问了三个对象中的任意两个后,剩下的一个必定是访问不了的。

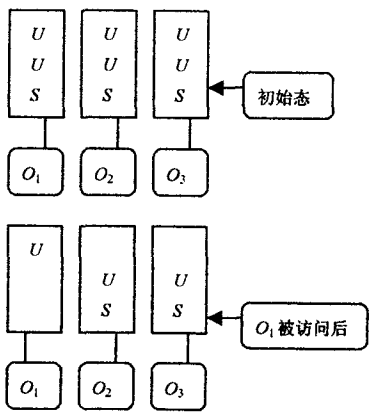

图 1　推理控制的例子

下面用算法来描述该模式。这个算法包含了一个密钥生成程序和查询处理算法。在此算法中,K 代表 m 个密钥的一个集合,f 代表用户的安全级。

算法 1　一个用户访问对象 O_i 的单推理通道控制

```
Key initialization:
  K(Oi) = K; i = 1,…,m
Query processing:
  Input: i;
Select a smaller key kj from K;
If kj > f then
    Output "access denied";
else
    K(Oi) = {kj};
    K(Os) = K(Os) \ {kj} for all s≠i;
    Deliver Oi to the user.
```

2.2　不相交的多推理通道

现在考虑无相交对象的多推理通道问题。不相交

是指两个推理通道之间没有相同的对象。解决的方法和2.1是类似的。为方便起见,用 C 代表数据库的一个推理通道:$C_1,\cdots,C_l$,推理通道 C_j 的长度用 m_j 表示,m_{max} 表示最长的那个推理通道的长度。可以假设所有推理通道的对象的最低安全级为 f,用户的访问等级为 t,则密钥集合 K 包含了 m_{max} 个密钥。采用如下的算法。首先,对于任一推理通道 C_j,K_j 是包含了 m_j-1 个 f,1 个 $t+1$ 密钥的集合。

算法 2 一个用户访问对象 O_i 的不相交的多推理通道控制

```
Key initialization:
Kj = {f,…,f,t+1}, j=1,2,…,l;
if Os∈Cj then K(Os) = Kj, for all possible s
Query processing:
    Input: i;
if find j such that Oi∈Cj then {
select the smaller key kp from K(Oi);
if kp>t then
  output "access denied" and quit;
else
    K(Oi) = kp;
  K(Os) = K(Os) \ {kp} for all Os∈Cj, s≠i;
  Deliver Oi to the user;
}
else
    output "information not found"
```

2.3 相交的多推理通道

最后,考虑情况比较复杂的相交的多推理通道。相交的多推理通道是指一些对象同时存在于两个或两个以上的推理通道中。相交的多推理通道又可以分为两种情况:一种情况是相交的对象不是任一通道的保留对象。在这种情况下,用户访问该相交对象和访问其他对象一样。另一种情况是相交的对象是一个保留对象,这样,用户的访问请求会被拒绝。

算法在下面描述。密钥集的初始化和 2.2 小节一样。因而使用的密钥总数为 m_{max}。对于相交的对象,密钥集的数目等于该对象在所有推理通道中出现的频率。用 $K_j(O_i)$ 代表对象 O_i 在通道 C_j 中的密钥集。假设所有对象的最低安全级为 f,用户访问安全级为 t。

算法 3 一个用户访问对象 O_i 的相交的多推理通道控制

```
Key initialization:
Kj = {f,…,f,t+1}, mj-1 个 f, j=1,…,l;
for All possible s do{
if Os∈Cj then Kj(Os) = Kj, j=1,…,l
}
Query processing:
    Input i:
if Oi∈Cj then {
select the smaller key kp from Kj(Oi);
if kp>t then
output "access denied" and quit;
else {
for every Cj such that Oi∈Cj do{
    while Os∈Cj and s≠i do {
    Kj(Os) = Kj(Os) \ {kp};
    If Kj(Os) = = {t+1} then
Ks(Os) = {t+1} for all r such that Os∈Cr;
            }
            }
    Deliver Oi to the user;
    }
  }
else
Output "information not found"
```

3 结束语

借鉴了文献[5]的部分思想并对它进行了大幅度的改进,使得密钥分配几乎不耗用机器时间,加快了访问效率。文中的算法中每个推理通道需要的密钥(也就是安全级)只有两个,在动态删除其他密钥时可以不用查询直接删除,这样查询效率有了大幅度的提高。由于本算法中保留对象是不定的,因而与静态提升安全级相比更加灵活。下一步的研究工作是如何处理用户通过不断地查询故意将某个对象锁住,阻止其他用户的访问。

参考文献:

[1] Zong T, Ozsoyoglu G. Controlling FD and MVD inferences in multilevel relational database system[J]. IEEE Trans on Knowledge and Data Engineering, 1991, 3(4): 474-485.

[2] Morgenstern M. Controlling logical inference in multilevel database systems[C]//Proc. of the IEEE Symp. on Security and Privacy. Oakland: [s.n.], 1998: 245-255.

[3] Stachour P, Thuraisingham B. Design of LDV: A multilevel secure relational database management system[J]. IEEE Trans on Knowledge and Data Engineering, 1990, 2(2): 190-209.

[4] 严和平,汪 卫,施伯乐.安全数据库的推理控制[J].软件学报,2006,17(4):750-758.

[5] Chen X, Wei R. A Dynamic Method for Handling the Inference Problem in Multilevel Secure Databases[C]//International Conference on Information Technology: Coding and Computing. Nevada: [s.n.], 2005: 751-756.