基于事务语句模板的数据库存取控制模型研究*)

钟 勇 秦小麟 包 磊
(佛山科学技术学院信息与教育技术中心 佛山 528000)
(南京航空航天大学信息科学与技术学院 南京 210016)

摘 要 传统的基于鉴于身份认证和存取控制的数据库安全机制存在一定的局限性,如无法防止 SQL 注入、反馈信息利用、合法用户权限滥用等非法行为,本文提出了一个基于事务语句模板和序列的存取控制模型,该方法可以有效地防止这些非法行为,同时对模型实现及其应用环境做了阐述,最后分析了模型实现对数据库查询性能的影响。

关键词 存取控制,模板,数据库安全,安全模型

Database Access Control Based on Transaction Sentence Templates

ZHONG Yong QIN Xiao-Lin BAO Lei
(Information and Educational Technology Center, Foshan University, Foshan 528000)
(Information Science and Technology Institute, Nanjing University of Aeronautics and Astronautics, Nanjing 210016)

Abstract There is limitation on the traditional user identification and access control of database security mechanism, such as to the illegal actions of SQL injection, disassembly with error messages, misusing authorization. The paper presents an access control model based on transaction sentence templates and its sequence. The model can prevent database from these illegal actions effectively. The paper also discussed the implementation method of the model and its application environment. At the end of the paper, a performance analysis to the database query is reported.

Keywords Access control, Template, Database security, Security model

1 传统的存取控制方法面临的一些问题

存取控制是一种很重要的数据库安全机制,通过授予用户或角色对客体的一定操作来限制用户的行为,存取控制是数据库安全的重要保障。但随着数据库日益网络化的发展,一些来自网络的攻击,攻击者往往能窃取到合法的身份或权限,如利用密码嗅探(password sniffing)攻击者可能获得合法的用户账号和密码,利用会话骑劫(session hijacking)的攻击者可能伪装成合法的用户,以及合法数据用户对自己权限的滥用,使数据库面临着前所未有的安全困境,例如以下一些问题是传统的存取控制很难或无法解决的。

(1) SQL 注入问题

SQL 注入[1](SQL injection)技术是一种入侵者利用字符串技术伪造恶意 SQL 语句来欺骗应用程序的入侵手段,SQL 注入往往是由于应用程序的设计疏忽或不完善等问题导致的。例如现今流行的在互联网上使用的三层 B/S 体系结构中,用户不直接查询数据库而是通过应用层来代理用户的请求,如使用 SQL SERVER 数据库的 T-SQL 事务语句和 C# 伪码的代码片段[2]。

```
[SqlCommandMethod (CommandType. Text,"SELECT * FROM Users WHERE Username=@username AND Password=@Passwd")]
public static DataSet GetAnnouncements (SqlConnection connection, int moduleId)
{…functions}
```

图 1 登录语句片段

在 T-SQL 中,字符串表示成'***',符号';'是 SQL 语句的结束符,符号--表示单行注释。如果应用程序没有为变量@username 和@Password 执行适当的输入验证,恶意用户可能进行 SQL 注入攻击,例如用户在@username 变量中输入"admin'--",查询将执行"Select * from users where username ='admin'--";输入"'or 1=1",查询将执行"Select * from users where username ="or 1=1"。这些语句都可以使用户绕过正常的密码检查。甚至用户可以在@username 变量中输入"';drop table user--",查询将执行"Select * from users where username =";drop table user--",导致用户表删除。输入代码验证如符号替换、输入限定、长度限制等方法常用来对付 SQL 语句注入,但这些方法也存在一定的限制和问题如文[1]所示。

(2) 数据库错误反馈信息利用问题

错误反馈信息利用[3]是通过数据库的错误反馈来推断程序和数据库结构的一种技术,如在使用 ODBC 编程接口的用户验证代码中,用户在 username 字段中输入"admin'",查询将执行"Select * from users where username ='admin' and password=''"。执行该语句系统可能返回错误如图 2 所示。

```
Micvrosoft OLE DB Provider for ODBC Drivers error '80040e14'
[Microsoft][ODBC SQL Server Driver][SQL Server]Unclosed quotation mark before the character string 'aaa' AND password=''
```

图 2 ODBC 返回错误片段

反馈信息中"AND password =''"片段泄露了程序语句以及在数据库中存在 password 的表字段,利用这些信息入侵者可以进一步地推断数据库的表结构[3]。

*)基金项目:航空科学基金(编号:02F52033),江苏省高技术项目(编号:BG2005-005)资助。钟 勇 讲师,博士研究生,主要研究方向数据库安全、网络安全。秦小麟 教授,博士生导师,主要研究方向安全数据库、空间数据库、时间数据库、GIS 等。包 磊 博士研究生,主要研究方向为空间数据库、时空数据库。

(3) 合法用户权限滥用问题

对合法用户如某个会计突然将自己每月工资增加一万元,在正常情况下这是不允许的,由于牵涉到应用语义,现存的存取控制无法表达这些语义。大多数商用数据库通过约束、规则或触发器等手段来表达应用语义约束,但无法表达更灵活的策略,比如高级会计师可以增加 5000 元的金额,一般会计师只能增加 2500 元的限额,现在这些限制只能通过嵌入在应用程序中来实现,应用程序使安全规则固化而难以修改且难以统一管理。

本文提出一个基于模板的存取控制方法来解决这些问题,本方法已应用在我们开发的基于能力的人力资源管理系统 CHRMS 中,也将用在我们正在开发的 DBMS 原型系统 NHSDB 中,作为数据库滥用检测的一部分。

2 基于模板的存取控制模型

针对以上一些存取控制所遇到的问题,我们提出使用模板授权的方式来防止这些问题。模板是用户使用数据库事务语句的固定模式,通过模板授权的用户只能按照模板规定的形式来操作数据库,为简单起见,本文在示例中使用 SQL Server 的 T-SQL。

2.1 模板定义

定义 1 模板 tmpl 是四元组〈OP, F, T, C〉, OP 是 SQL 语句的语句类型,F 是属性集,T 是表集合,C 是查询条件集。

例如对 SELECT 语句"select 编号,姓名,地址 from 雇员表 where 年龄>20 and 年龄<60"的模板形式见表 1。

表 1

类型(OP)	SELECT
属性集 (F)	编号, 姓名, 地址
表集 (T)	雇员表
条件集 (C)	(年龄>%数字%) ∧ (年龄<%数字%)

对 INSERT、UPDATE、DELETE 等语句模板也可以有不同的形式,如对 UPDATE 语句,其 F 集可以是属性赋值集,如对更新语句"update 雇员表 set 地址='北京' where 年龄>20 and 年龄<60"的模板见表 2。

表 2

类型(OP)	update
属性集 (F)	地址=%文本%
表集 (T)	雇员表
条件集 (C)	(年龄>%数字%) ∧ (年龄<%数字%)

为叙述简单起见,本文以 SELECT 语句模板为例。

2.2 模板执行序列

在事务或应用模块中,模板之间存在一定的执行顺序,如下类似于正则表达式的方式来表示事务模板的执行序列。

()将多条执行语句组合成单个执行单元。

{}表示前面执行单元的执行次数,例如(ts){1,3}表示 ts 可能执行一次到三次。

* 表示前面的执行单元执行零次或多次。

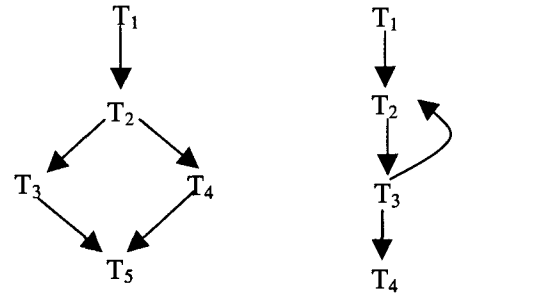
+ 表示前面的执行单元执行一次或多次。

? 表示前面的执行单元执行零次或一次。

|表示执行前面或后面的单元,如单元(ab|cd)表示执行单元 ab 或 cd。

△ 终止符号,表示事务流程可以在符号处终止执行。

如图 3 所示,应用模块或事务 t 的执行序列表示为 $S_t = T_1T_2(T_3|T_4)T_5$,应用模块或事务 r 的执行序列表示为 $S_r = T_1(T_2T_3)+T_4$。



应用模块或事务 t 的执行序列　　应用模块或事务 r 的执行序列

图 3 执行序列示例

2.3 模板规范

(1) 模板属性集 F 和表集 T

属性集中的属性包括直接查询的属性和查询表达式中的属性,即对查询表达式 $E=f(field_1,\cdots,field_n)$,则属性 $field_1,\cdots,field_n$ 也属于 F,例如对查询"select 产品名称,日期,数量 * 价格 as 销售额 from 销售表,产品表",F={产品名称,日期,数量,价格}。表集 T 是查询所涉及到的关系表名,如上述查询表集 T={销售表,产品表}。

(2) 条件集 C 的规范

步骤 1:将条件集中的属性取值以其类型代替,表示为"%T%",其中 T 是属性类型。如数字表示成"%数字%",日期表示成"%日期%"。

步骤 2:对嵌套查询的处理。对带有比较谓词和运算符的嵌套查询如图 3 的 select 语句,对简化的处理可以简单地去除内嵌的 SQL 语句,以数据类型代替。在需要更细微入侵检测政策和环境当中,使用"[E,L]"的格式代替"%T%",其中 E 是规范后的查询谓词或比较运算符,L 是指向子查询模板的链接。图 4 是模板嵌套的示例。

步骤 3:领域知识的处理。在预先得到的领域知识或通过对训练集的学习,可以得到 C 集中属性的取值范围,使用 E={$e_1,\cdots,e_n$}表示,e_i 是属性的取值或取值范围。对文本格式使用正则表达式来限制属性格式。例如对使用 Email 地址的查询"select 编号,姓名,地址 from 雇员表 where Email=nanhang@163.net"的查询语句,可得到模板〈OP={select},F={学号,姓名},T={学生表},C={Email=\w+@((\w+[.]?)+)}〉;对取值范围使用[ ]、( )、[ ]、() 符号进行限定。如进一步限定查询对只能查询学生表中 18 到 25 岁或 30 岁以上的计算机系或信息管理系的学生,模板为〈 OP={select}, F={学号,姓名},T={学生表},C={系名=计算机系|信息管理系∧年龄={[18~25],[30-*]}∧Email=\w+@((\w+[.]?)+)}〉。在步骤 4 中,条件集 C 转换成合取范式,省略 C 集中的合取符号∧,也可以得到表 3 的模板形式。

步骤 4:利用分配律将条件集 C 转换成规范的合取范式。如条件集 $A_1 \wedge A_2 \vee A_3 \rightarrow (A_1 \vee A_3) \wedge (A_2 \vee A_3)$。

(3) 模板和序列的精简

在应用中,产生过多的模板和序列,对模板的查找和匹配如影响了数据库的性能,在略微放松安全性的情况下,可以对模板进行精简和合并,减少模板的数量,下列模板中的关系可以用来进行精简和合并。

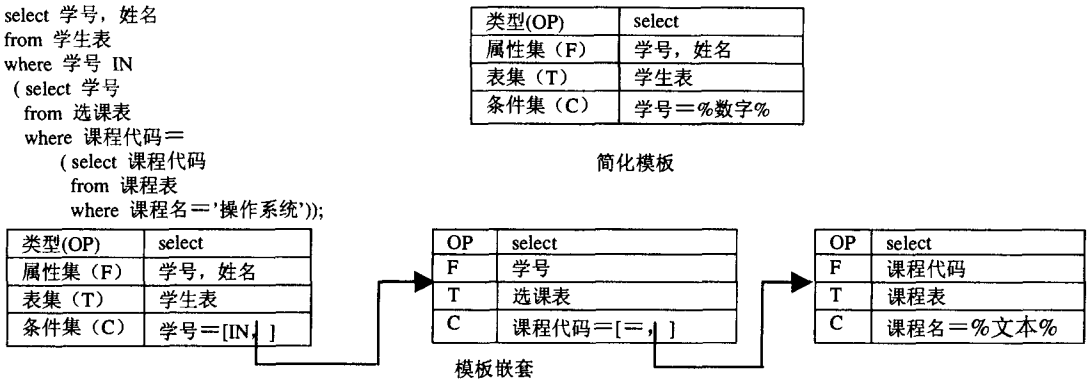


图 4　嵌套查询处理

表 3

OP	select
F	学号，姓名
T	学生表
C	系名=计算机系 | 信息管理系，年龄={[18-25],[30-*]}，Email=\w+@((\w+[.]?)+)}

定义 2　属性包含关系。对于任意模板 $tmpl_1=\langle op_1, f_1, t_1, c_1\rangle$，$tmpl_2=\langle op_2, f_2, t_2, c_2\rangle$，如果 $op_1=op_2$，$t_1=t_2$，$c_1=c_2$ 且 $f_1\subset f_2$，则认为模板 $tmpl_1$ 和 $tmpl_2$ 存在属性包含关系，记为 $tmpl_1\subset^f tmpl_2$。如查询模板〈$OP=\{select\}$，$F=\{$姓名$\}$，$T=\{$学生表$\}$，$C=\{$系名=%文本%$\}\rangle\subset^f\langle OP=\{select\}$，$F=\{$学号，姓名$\}$，$T=\{$学生表$\}$，$C=\{$系名=%文本%$\}\rangle$。

具有属性包含关系的模板 $tmpl_1\subset^f tmpl_2$，对特定应用程序具体分析使用 $tmpl_2$ 代替 $tmpl_1$ 对安全性的影响。用 D_{tmpl} 表示模板 $tmpl$ 执行涉及到的所有数据集，P_{tmpl} 表示 $tmpl$ 对数据执行的权限总量，我们称不属于任何序列的语句为单语句。

(1) $tmpl_1$ 和 $tmpl_2$ 都是单语句。由于 $tmpl_2$ 比 $tmpl_1$ 包含更多的属性，则显然有 $D_{tmpl1}\subset D_{tmpl2}$，从而 $P_{tmpl1}\subset P_{tmpl2}$，执行 $tmpl_2$ 的权限包含执行 $tmpl_1$ 的权限，既然应用模块有执行 $tmpl_2$ 的权限，用 $tmpl_2$ 代替 $tmpl_1$ 不影响系统安全性。

(2) $tmpl_1$ 属于执行序列 s_1，$tmpl_2$ 为单语句。在执行序列中由于语句可能受到序列上下文的限制，其涉及到的数据集小于等于单语句涉及到的数据集。如下例事务中的 *Select* 语句如果在事务执行序列中执行的话，其涉及到的数据始终为零，如果应用程序只具有在该事务中执行该语句的权限，那么其实际权限远小于执行单语句所具有的权限。用 D_{tmpl1}'、P_{tmpl1}' 表示 $tmpl_1$ 模板在单语句上的数据集和权限，则有 $D_{tmpl1}\subseteq D_{tmpl1}'\subset D_{tmpl2}$，从而有 $P_{tmpl1}\subseteq P_{tmpl1}'\subset P_{tmpl2}$，用 $tmpl_2$ 代替 $tmpl_1$ 不影响系统安全性。

```
Begin tran
  Update 成绩 Set 成绩=60 From 成绩表 where 成绩<60
  Select 学号,姓名 from 成绩表 where 成绩<60
Commit tran
```

(3) $tmpl_1$ 是单语句，$tmpl_2$ 属于序列 s_2。用 D_{tmpl2}'、P_{tmpl2}' 表示 $tmpl_2$ 模板在单语句上的数据集和权限，则存在关系 $D_{tmpl2}\subseteq D_{tmpl2}'$、$D_{tmpl1}\subset D_{tmpl2}'$，在权限上 $P_{tmpl2}\subseteq P_{tmpl2}'$、$P_{tmpl1}\subset P_{tmpl2}'$，$P_{tmpl1}$ 无法和 P_{tmpl2} 作直接比较，用 $tmpl_2$ 代替 $tmpl_1$ 可能带来系统安全性上的损失。

(4) $tmpl_1$ 和 $tmpl_2$ 均属于某序列(包括属于相同序列)。用 D_{tmpl1}'、P_{tmpl1}'、D_{tmpl2}'、P_{tmpl2}' 分别表示 $tmpl_2$ 和 $tmpl_2$ 模板是单语句时的数据集和权限，则存在数据关系 $D_{tmpl1}\subseteq D_{tmpl1}'$、$D_{tmpl2}\subseteq D_{tmpl2}'$、$D_{tmpl1}'\subset D_{tmpl2}'$，权限关系 $P_{tmpl1}\subseteq P_{tmpl1}'$、$P_{tmpl2}\subseteq P_{tmpl2}'$、$P_{tmpl1}'\subset P_{tmpl2}'$，同样，$P_{tmpl1}$ 无法和 P_{tmpl2} 作直接比较，用 $tmpl_2$ 代替 $tmpl_1$ 可能带来系统安全性上的损失。

具有不同安全特征的系统可以采用不同的处理方法，在很多场合，可以近似认为 $P_{tmpl1}\approx P_{tmpl1}'$。

定义 3　条件包含关系。对于任意模板 $tmpl_1=\langle op_1, f_1, t_1, c_1\rangle$，$tmpl_2=\langle op_2, f_2, t_2, c_2\rangle$，如果 $op_1=op_2$，$f_1=f_2$，$t_1=t_2$ 且 $c_2\subset c_1$，则认为模板 $tmpl_1$ 和 $tmpl_2$ 存在条件包含关系，记为 $tmpl_1\subset^c tmpl_2$。在这里注意条件是规范的合取范式，条件越少，返回的数据集越多。对条件 $c_1=a_1\wedge a_2\wedge\cdots\wedge a_n$ 和 $c_2=b_1\wedge b_2\wedge\cdots\wedge b_m$ 关系，可以使用真值表来定义包含关系，为了操作的方便性，简单定义为对任意 $a_i\in c_1(1\leqslant i\leqslant n)$，存在 $b_j\in c_2(1\leqslant j\leqslant m)$ 使得 $a_i=b_j$，则定义 $c_1\subset c_2$。如查询模板〈$OP=\{select\}$，$F=\{$学号$\}$，$T=\{$学生表$\}$，$C=\{$系名=%文本% $\wedge$年级=%整数%$\}\rangle\subset^c\langle OP=\{select\}$，$F=\{$姓名$\}$，$T=\{$学生表$\}$，$C=\{$系名=%文本%$\}\rangle$。

对模板 $t_{mpl1}\subset^c t_{mpl2}$，如果两个模板均属于单语句，则显然有 $P_{tmpl1}\subset P_{tmpl2}$，在涉及到序列时，条件包含关系具有与上述属性包含关系相同的性质。

定义 4　属性条件包含关系。对于任意模板 $tmpl_1=\langle op_1, f_1, t_1, c_1\rangle$，$tmpl_2=\langle op_2, f_2, t_2, c_2\rangle$，如果 $op_1=op_2$，$t_1=t_2$ 且 $f_1\subset f_2$，$c_2\subset c_1$，则认为模板 $tmpl_1$ 和 $tmpl_2$ 存在属性条件包含关系，记为 $tmpl_1\subset^{fc} tmpl_2$。

定义 5　条件范围包含关系。对于任意模板 $tmpl_1=\langle op_1, f_1, t_1, c_1\rangle$，$tmpl_2=\langle op_2, f_2, t_2, c_2\rangle$，如果 $op_1=op_2$，$f_1=f_2$，$t_1=t_2$ 且对条件 $c_1=a_1\wedge a_2\wedge\cdots\wedge a_n$ 和 $c_2=b_1\wedge b_2\wedge\cdots\wedge b_m$，如果对任意 $b_j\in c_2(1\leqslant j\leqslant m)$，存在相应的 $a_i\in c_1(1\leqslant i\leqslant n)$，有 a_i 的范围属于 b_j 的范围，$a_i\in b_j$，则认为 $tmpl_2$ 条件范围包含 $tmpl_1$，记作 $tmpl_1\subset^{\alpha} tmpl_2$。

精简规则 1：对任意应用模块 P，如果模板 $tmpl_1$ 是单语句，对任意该应用模块的授权模板 $tmpl_2$，如果存在关系 $tmpl_2\subset^f tmpl_1$ 或 $tmpl_2\subset^c tmpl_1$ 或 $tmpl_2\subset^{cf} tmpl_1$ 或 $tmpl_2\subset^{\alpha} tmpl_1$，则对应用模块 P，$tmpl_1$ 和 $tmpl_2$ 之间存在精简关系，可以用 $tmpl_1$ 来代替 $tmpl_2$。

精简规则 2：对任意模板序列 S，如果 S 中任意的模板 $tmpl$，都存在可以精简 $tmpl$ 的单语句 $tmpl'$，则可以精简序列 S。这是因为既然应用模块对序列 S 的任意语句都有单独执行的权限，也就是说应用模块有对这些语句任意组合执行的权限，自然也就包括 S 的排列，精简 S 不影响权限的分配。在实现中将 S 作为隐藏序列不参加匹配，但如果存在 $tmpl'$ 模板被删除的情形则激活 S。

从严格安全意义的形式验证来讲，对于存在精简关系的

模板 $tmpl_1$ 和 $tmpl_2$ 用 $tmpl_1$ 来代替 $tmpl_2$ 并非严格意义上的完全安全，上面我们已经证明，从应用模块对数据操作的权限 P 来讲，替代关系不影响应用模块对数据的授权权限，但从语句的角度来讲，除非 $tmpl_2$ 是单语句，否则替代关系放大了 $tmpl_2$ 在序列中的权限，可能这种放大了的权限将影响序列的下文执行，例如 $tmpl_2$ 是插入星期一至星期五的插入语句，下文依照插入的结果做其它查询，如果 $tmpl_2$ 放大到可以插入星期六和星期天，可能导致下文错误。如果要实施完全安全意义上的精简，只有两种情况，一是模板 $tmpl_1$ 和 $tmpl_2$ 都是单语句，二是精简规则 2 的情形。由于这种安全的影响是微乎其微的，在此不再详述。

(4) 模板合并

合并规则：对任意应用程序 P，对单语句模板 $tmpl_1=\langle op_1, f_1, tl, c_1\rangle$，$tmpl_2=\langle op_2, f_2, t_2, c_2\rangle$，如果 $op_1=op_2$，$f_1=f_2$，$t_1=t_2$ 且对条件集 c_1 和 c_2，除属性 E 取值范围外，$tmpl_1$ 和 $tmpl_2$ 具有相同的条件集，将 $tmpl_1$ 条件集中属性 $E=\{e_1,\cdots,e_n\}$ 与 $tmpl_2$ 条件集中属性 $E=\{e'_1,\cdots,e'_m\}$ 合并成 E 新的取值范围，以 E 新的取值范围代替 $tmpl_1$ 或 $tmpl_2$ 条件集中属性 E 的取值范围得到新模板 $tmpl_3$，使用新模板 $tmpl_3$ 代替 $tmpl_1$ 和 $tmpl_2$。

例如模板 $tmpl_1=\langle OP=\{select\}, F=\{学号\}, T=\{学生表\}, C=\{系名=\%文本\% \wedge 年级=\{[20,30]\}\}\rangle$ 和 $tmpl_2=\langle OP=\{select\}, F=\{学号\}, T=\{学生表\}, C=\{系名=\%文本\% \wedge 年级=\{[40,*]\}\}\rangle$ 合并成 $tmpl_2=\langle OP=\{select\}, F=\{学号\}, T=\{学生表\}, C=\{系名=\%文本\% \wedge 年级=\{[20,30],[40,]\}\}\rangle$。

模板合并只在单语句模板中进行，模板合并不影响系统的安全性。

2.4 模板存取控制对防止入侵和滥用的应用

对 SQL 注入的入侵，如在图 4 中，对任意用户 u，如果对应用模块或事务 t 的执行序列 s 与 S_t 不匹配 $s\not\subset S_t$，则认为用户 u 的执行序列出现偏差，系统将中止 u 的执行序列。例如对上述的用户姓名和密码查询“select * from users where Username=@username and Password=@ Passwd”，假设该应用程序的执行模板序列$\langle OP=\{select\}, F=\{*\}, T=\{users\}, C=\{username=.*, password=.*\}\rangle$，$\langle OP=\{select\}, F=\{学号,姓名\}, T=\{学生表\}, C=\{username=.*\}\rangle$。那么上述的 SQL 注入查询如“Select * from users where username ='admin'–”或“Select * from users where username =''or 1=1”由于不符合模板一将无法执行，查询“Select * from users where username ='';drop table user–”由于不满足模板的执行序列而无法执行。

对数据库错误反馈信息利用问题，由于模板授权系统截获错误或恶意查询，查询不将提交数据库，模板授权系统将屏蔽数据库错误反馈信息。通过模板可以限制和或授予事务语句的属性范围，限制用户的数据操作范围，因此也可以防止很大程度上的数据库滥用。

3 系统实现和具体应用

用户查询序列 S 在提交到数据库之前，模板编译器首先将 S 编译成模板实例 S^i，模板实例是将查询规范成模板后将查询取值代入模板后所得，例如对查询“select * from users where Username='小王' and Password='abcd'”经编译后得到模板实例$\langle^i OP=\{select\}, F=\{学号,姓名\}, T=\{学生表\}, C=\{Username='小王', Password='abcd'\}\rangle$，模板识别器将实例与模板库中的授权模板匹配，如果实例没有匹配相应的授权模板，则取消 S 的提交，否则得到 S 的匹配模板序列 S^T，序列识别器将 S^T 与授权序列库进行匹配，如果存在匹配序列则向数据库提交查询序列 S，否则不提交 S(如果 S^T 中的任意模板均匹配单语句则不需要经过序列识别器而可以直接提交)。

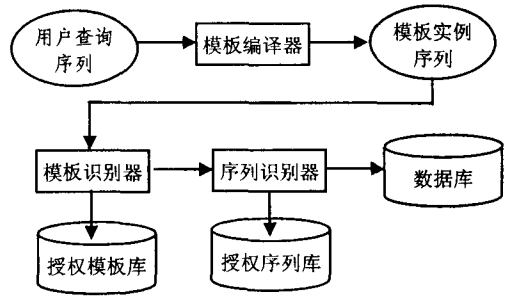


图 5 查询提交过程

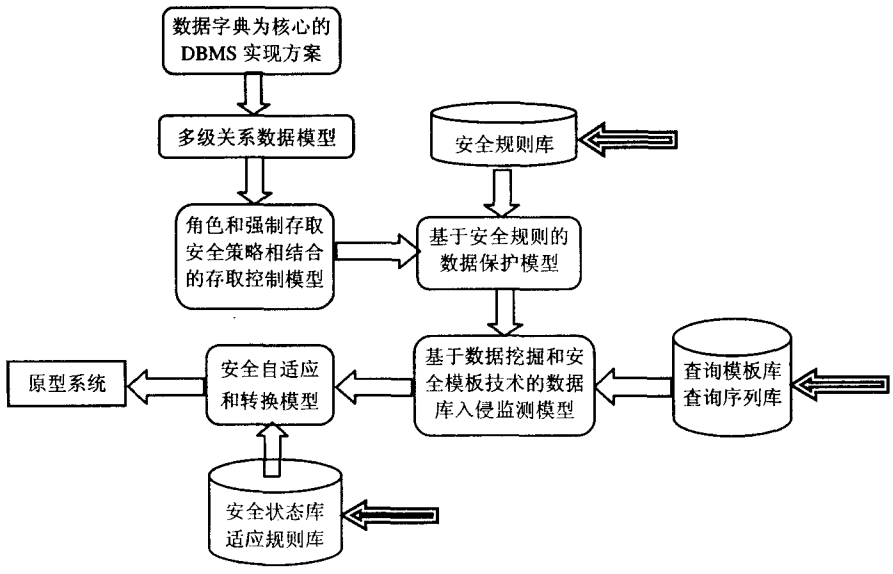


图 6 NHSDB 数据安全体系

基于事务语句模板的数据库存取控制方法应用在我们开发的基于能力的人力资源管理系统 CHRMS(Competence-based Human Resource Management System)中，基于能力的人力资源管理思想是彭剑锋，饶征等提出的一套以战略为导向的人力资源开发与管理体系[4]。CHRMS 系统使用三层的 B/S 结构，Visual studio. net 2003 作为开发环境，ASP. net 实现中间层，部分业务逻辑封装在自定义类中，以 SQL Server 2000 作为后台数据库服务器，图 5 的查询提交过程应用在中间层的自定义类中，下文对其性能做了测试。

本方法也将应用在我们正在开发的 NHSDB[5]（南航安全数据库）中，NHSDB 是一个基于多级关系数据模型，实现基于角色访问控制和基于属性标记强制安全策略相结合的 B 级数据库管理系统，该实现一个具有身份认证、存取控制、安全规则和入侵检测四层保护机制的自主版权数据库原型系统，具有入侵容忍和安全状态自适应能力，面对不同的安全形势调节数据库的安全等级和数据保护状态。基于事务语句模板的方法作为 NHSDB 入侵检测的一部分，我们将偏离用户查询模板库和用户查询序列库的查询作为入侵的征兆，并使用数据挖掘的方法学习和生成用户查询模板库和用户查询序列库，NHSDB 的安全体系如图 6 所示。

4 性能分析

用户查询在提交前，需要经过授权模块和授权序列的匹配检测，对查询的性能有一定影响，在微机上测试了模板授权对查询性能的影响。测试硬件环境：基于 Intel 结构微机二台，CPU 1500MHz，内存 256M；软件环境：Windows 2000 Server，SQL Server 2000，采用如上所述 CHRMS 系统的编程环境。数据库用例：图书馆采购数据库（包含书名、作者、出版者、出版地、时间、价格、ISBN 号等共 12 个字段），测试前共有 229073 条记录。

表 4 顺序查找时匹配模板时间占用整个查找时间的百分比（最坏情况）

模板数 / 查询语句数	500	1000	2000	3000	5000	10000
1	0.6641%	2.0319%	3.2038%	6.0587%	6.6277%	16.3051%
10	0.8198%	1.3182%	1.7721%	3.1308%	5.8870%	14.8960%
100	0.8626%	1.7767%	1.9870%	5.0608%	6.8089%	15.2010%
500	0.8949%	1.8623%	2.9674%	5.2956%	8.4513%	15.6616%
1000	0.7951%	1.8841%	3.2189%	5.2917%	7.7704%	16.3060%

测试过程中先将所有模板读入内存，这是因为在三层体系结构中，用户通过应用层来代理用户的请求，对数据库来说，单个具体应用模块的所有用户账号往往都映射到单一的数据库账号，对单个应用模块来说需要基于模板授权的数据库帐号并不多，经过精简合并后的模板数量有限，而当今服务器的内存往往很大，这个假设前提是有理由的。模板在内存中的组织采用有序表保存属性集 F 和表集 T，Hash 表保存条件集 C，使用 Hash 表增加了模板读入内存的速度但提高了查找效率。为了测试最坏情况下的匹配速度，采用完全顺序查找的方式匹配模板，分别在模板数为 500、1000、2000、3000、5000、10000 情况下，对查询语句数分别为 1、10、100、500、1000 的情况下进行测试，测试次数为 100 次，取其平均数，得到下表所示模板匹配时间占整个查询时间的百分比。

从表 4 中可以看出匹配时间比并不随查询语句数增加而增加，影响匹配时间比的主要因素在于模板数，当模板数以下 5000，匹配时间只占总时间几个百分比，当模板数达到 10000，匹配百分点超过十位数，对性能影响较大。由于上表是在完全顺序匹配的情况下得到的最坏结果，通过在内存中建立按类型 O、表集 T、属性集 F 的索引，可以极大提高匹配时间，在此不再详述。

以上性能分析表明，基于模板的存取方式是可行的，对查询性能的影响十分有限。

结束语 本文针对数据库中存在的 SQL 注入、反馈信息利用、合法用户权限滥用等数据库入侵或滥用行为，提出了一个事务语句模板的存取控制模型以及应用方法，并对该方法对数据库查询性能的影响做了测试。

下一步将本方法应用到开发的数据库原型 NHSDB 的内核中，作为数据库入侵监测模型的一部分，将需要研究如何通过训练和学习来自动生成用户查询模板，以及在数据库内核中对数据库查询的影响。

参 考 文 献

1 Anley C. Advanced SQL Injection In SQL Server Applications. Next Generation Security Software Ltd. Available at: URL http://www. nextgenss. com/papers/advanced sql injection. pdf (2002)

2 O'Neill M，等著. 冉晓旻，郭文伟译. Web 服务安全技术与原理. 北京：清华大学出版社，2003. 168～169

3 Litchfield D. Web Application Disassembly with ODBC Error Messages. http://www. nextgenss. com/papers/webappdis. doc

4 彭剑锋，饶征. 基于能力的人力资源开发与管理. 中国人民大学出版社，2003. 43～53

5 Zhong Yong，Qin Xiaolin. Adaptive Data Protection Mechanism in Intrusion Tolerant Multilevel Secure Database. In: the Proceedings of the Second Asian Workshop on Foundations of Software. Nanjing，China，Dec. 2003. 21～24