加密数据的一种高效查询方法

王正飞[1,2],汪 卫[3],施伯乐[3]
WANG Zheng-fei[1,2],WANG Wei[3],SHI Bo-le[3]

1.湖南商学院 计算机与电子工程系,长沙 410205
2.国防科学技术大学 计算机学院,长沙 410073
3.复旦大学 计算机信息与技术系,上海 200433
1.Department of Computer Electronic Engineering,Hunan Business College,Changsha 410205,China
2.School of Computer Science,National University of Defense Technology,Changsha 410073,China
3.Department of Computer Information and Technology,Fudan University,Shanghai 200433,China
E-mail:feizhengwang@163.com

WANG Zheng-fei,WANG Wei,SHI Bo-le.Efficient method of querying encrypted data.Computer Engineering and Applications,2008,44(12):29-33.

Abstract: Encryption technology has become an important mechanism of securing data stored in database.However,it is difficult to search efficiently the encrypted data and many researchers take it into consideration.To solve the problem,a novel way is proposed,we not only encrypt the character data by using the conventional encrypted algorithms,but also adopt the way of flatting and scrambling,and then store the translated characteristic values together with the encrypted data in the encrypted table. The approach can provide better security,and results of experiments validate the performance of our approach compared with the conventional approach.

Key words: database security;encrypted data;query;performance

摘 要:加密技术是保护数据库中数据安全的一种有效方法,但如何对加密数据进行高效查询是一个难点,引起了研究界的重视。针对这个问题,除了采用常规加密方法对字符数据进行加密外,还对字符数据进行扁平化和扰乱化处理,并把处理后的特征值作为附加字段与加密数据一起存储。该方法不仅安全性很好,而且通过实验证明其性能较传统方法有很大提高。

关键词:数据库安全;加密数据;查询;性能
文章编号:1002-8331(2008)12-0029-05 **文献标识码:**A **中图分类号:**TP311.13;TP309

1 引言

数据库中包含大量重要的数据,其安全问题引起越来越多的关注。传统的,数据库中一些敏感信息通过访问控制机制来提供保护,但是它并不能完全有效地防止外部人员的攻击,内部人员,甚至是数据库管理员(DBA)的蓄意破坏。例如,如果一个未授权用户能够绕过数据库的访问控制机制,他就可以非常容易地获得对原始数据的访问权限。如何在这样一种非可信的环境下保证数据库的安全已经成为研究的一个热点,其中,通过加密机制保护数据库中的敏感信息是一种有效的手段[1-5],如身份证、银行账户、个人健康信息等。

采用传统的加密算法(如:DES,ADS)对数据进行加密后,对于字符串的精确匹配查询,可以不用解密数据而对加密数据直接查询。例如,当SQL条件语句包含"="时,先对SQL中的条件值进行同样加密处理,然后与数据库中存储的加密数据比较,如果相等,返回相应的记录。然而,对于加密数据上的大量模糊匹配查询(如:>,<,like),需要先解密才能进行查询。因为数据加密后,一些固有属性发生变化[12],例如数据的有序性(Ordering)、相似性(Similarity)、可比性(Comparability)遭到破坏,没有办法直接对密文数据进行比较判断。如果对所有加密数据进行解密,然后再查询,这种操作开销巨大,将极大地影响查询性能,在实际操作中是不可行的[13]。所以,需要找到一种有效处理字符数据的加密方法,既能保证其安全性,又不会对查询性能产生较大影响。

文章剩余部分组织如下:第2章分析了相关工作,第3章给出字符数据的加密存储和查询方法,第4章进行安全性分析,第5章通过实验数据验证这种方法的可行性,第6章总结全文。

基金项目:国家自然科学基金(the National Natural Science Foundation of China under Grant No.69933010);湖南省教育厅资助科研课题(the Research Project of Department of Education of Hunan Province,China under Grant No.07C400)。

作者简介:王正飞(1970-),男,博士,讲师,CCF会员,研究方向为数据库安全技术,数据库加密等;汪卫(1970-),男,教授,博士生导师,研究方向为数据库理论,数据库安全技术等;施伯乐(1936-),教授,博士生导师,研究方向为数据库理论等。

收稿日期:2007-12-05 **修回日期:**2008-02-01

2 相关工作

当数据库中存储密文数据时,查询密文数据的性能急剧下降。为了平衡数据库中敏感信息的安全和查询性能,一些工作人员对数据库加密进行了研究[6-11]。文献[6]提出了一种比较简单的加密方法,对于某整数 p,其对应的加密值 $c=\sum_{j=0}^{p} R_j$,其中 R_j 是由伪随机产生器 R 产生的第 j 个伪随机数。由于两个密文值的差与其对应两个明文值的差成一定比例关系,攻击者能够根据密文值的概率分布推导出其明文值的概率分布,从而推导其对应的明文值。所以,这种加密方法安全性不好,容易遭受已知明文攻击和统计攻击。

文献[7]中,在数据库作为一种服务(Database As a Service)的背景下,提出了对加密数据查询的方法。存储时,除了对关系表中的元组采用常规加密外,还给每个属性值增加一个桶号,桶号表示明文数据值落在某段区间内。查询时,客户端提交的 SQL 查询语句可以直接执行在加密数据上,而无须解密。文献[8]进一步对它如何分桶进行改进,给出最优分桶算法,使得查询代价最少。但是,这种方法中,返回给客户端的记录集可能包含一些不满足查询条件的记录,需要再进行解密和查询处理。而且,这种方法对于多表连接查询的代价非常大。

文献[9]提出了一种保持有序的加密方法。给定一个目标分布函数,对明文值进行转换得到密文,使得密文不仅保持有序,而且服从某一目标函数的分布。由于其密文保持有序,无须解密就可以直接对密文进行等值和范围查询,也可以进行 MAX、MIN、COUNT 和 ORDER BY 查询。很明显,这种方法的安全性较差,因为密文保持有序性,容易遭受选择密文攻击。也就是说,如果攻击者能够选择一定数量的明文(或密文),并且把它们加密(或解密)成对应的密文(或明文),那么他就能够以较大的概率估计出密文对应的明文值。类似地,如果攻击者知道这个域的一些信息,比如说该域的数据分布,他也能够以较大的概率估计密文对应的明文值。

文献[10]中,智能卡具有加密和查询处理能力,它把数据加密后存储在服务器中,密钥也存储在智能卡中。查询时,智能卡能够对等值查询语句进行转换,使之能够对密文数据进行查询。但是,它不能对实数类数据进行范围查询。

上述方法主要集中在对数值型数据的加密研究,对于另一主要数据类型——字符型数据的研究成果相对较少。与之相近的研究出现在文献[11]中,它使用序列加密(stream cipher)方法对文本数据进行加密处理,这样可以无须解密而直接对加密文本搜索关键词,但是这种方法没有涉及到如何应用到数据库中。

3 字符数据的加密存储与查询

3.1 存储与查询的体系结构

考虑到 DBMS 内部结构非常复杂,很难对现有 DBMS(如 Oracle,SQL SERVER 等)进行修改,所以采用 DBMS 外部方法对数据进行加密处理,如图 1 所示,在 DBMS 和应用程序之间增加了加/解密层,专门由这一层来完成数据的加密存储和查询。

存储时,在应用程序中的数据存入数据库之前,需要进行两方面处理。一方面,由加/解密层调用加密算法对字符数据进行加密处理,然后存储到数据库;另一方面,为提高加密数据查询性能作好准备,在原有数据表中增加一个新字段,并存储该字符数据的特征值。

查询时,采用两阶段查询方法。首先对加密数据进行一次粗糙查询(Coarse Query),主要利用加密数据表中的新增字段,过滤大部分与查询条件无关的记录;然后对剩余记录中的加密数据进行解密,在解密的数据上再进行一次精确查询(Refined Query),最终实现查询目标。

在图 1 的加/解密层中,元数据是一些函数映射规则,用来修改存储和查询语句,存储时,除了加密数据外,还通过这些规则修改 SQL 语句,存储加密数据的特征值;查询时,通过这些规则修改查询语句,转换为对加密数据查询的 SQL 语句。加密和解密分别是由加/解密函数来实现对数据库中敏感信息列进行加密和解密的功能模块。

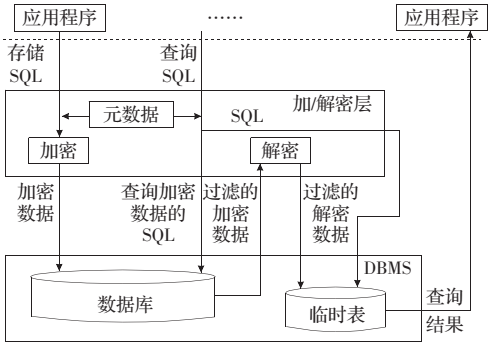


图 1 加密字符串数据存储与查询的体系结构

3.2 存储模式

对于传统关系模式 $R(X_1,\cdots,X_r,\cdots,X_n)$,其中,$X_r$ 为需要加密的属性,存储到数据库中,将以加密关系模式 $R^E(X_1,\cdots,X_r^E,\cdots,X_n,X_r^S)$ 存储,其中,R^E 中 X_r^E 是 R 中 X_r 的一个映射,$X_r^E=E(X_r)$,E 为加密算法,X_r^S 是新增的一个属性,用来存储 X_r 的特征值,在本文中称为特征属性。

显然,对 R^E 中的 X_r^E,只需要调用加密算法对 R 中的 X_r 进行处理即可得到。而 R^E 中的 X_r^S 用来存储 X_r 中数据的特征值,如何提取 X_r 中的特征值是一个关键问题。一般来说,提取特征值时,一定要考虑如下两方面:(1)保证加密数据的安全,提取数据的特征值时,特征值本身不能泄漏 X_r 中的信息;(2)保证提高加密数据的查询性能,在第一阶段的粗糙查询阶段中,利用特征值过滤与查询条件无关的大部分记录,这是提高查询性能的重要环节。下面将详细讨论特征值的提取过程。

3.3 特征值提取

为了保证特征值安全和提高加密数据查询性能,数据特征值的提取分为排序、扁平化和散列三个过程,经过处理后,将得到每一字符对所对应的特征值。排序和扁平化过程主要功能是根据字符对的出现频率,将它们均匀地分布到各个桶中,保证提取的特征值不会出现大的偏向性,而散列过程主要功能是打乱数据的分布顺序。通过上述三个过程的处理,攻击者很难从特征值获取敏感信息。

3.3.1 排序:频率排序(Frequency Sorting)函数

定义 1 设有排序函数 $FS:s_1\rightarrow s_2$,其中,s_1 为二元组$\{(c_1c_1,f_{11}),(c_1c_2,f_{12}),\cdots,(c_ic_j,f_{ij}),\cdots,(c_nc_n,f_{nn})\}$,$c_ic_j$ 表示字符表中的字符对,f_{ij} 表示 c_ic_j 出现的频率,且 $f_{ij}>0$,s_2 也包含同样的二元组,但是按照 f_{ij} 的降序(或升序)进行排列,称函数 FS 为频率

排序函数。

例 1 假设一些字符对和其对应的频率为{(ly,186),(re,64),(sl,13),(tl,80),(fu,103),(gr,37),(in,78),(ei,45)},经过排序后,其结果为{(ly,186),(fu,103),(tl,80),(in,78),(re,64),(ei,45),(gr,37),(sl,13)}。

3.3.2 扁平化(Flatten)

扁平化的主要思想是把各字符对分配到一定数量的桶中,使各个桶中所包含的字符对的频率之和大致相等。在扁平化的过程中,桶与字符对的关系为多对多的关系,也就是说,一个桶中可能包含多个字符对,而一个字符对也可能出现在多个桶中。设桶的个数为 m,$sum(f)$为所有字符对的频率之和,$f(B)$表示一个桶可分配字符对的频率总和,且应满足下面的约束:

$$f(B)=\lceil \frac{sum(f)}{m} \rceil+\gamma$$

γ 为浮动值,它的大小一般为$\lceil \frac{sum(f)}{m} \rceil \times 5\%$。

具体来说,扁平化的过程为:

(1)求总频率:$sum(f)=\sum_{i}^{n}\sum_{j}^{n} f_i f_j = f_{11}+f_{12}+\cdots+f_{nn}$

(2)按照已经排序的字符串,顺序地分配字符到各个桶中。分配的原则是:

①按照顺序给各个桶分配字符对,只有当前一桶分配好,才给下一个桶分配,直到分配结束。

②当一个桶还有剩余空间时,从未分配的字符对中找到一个频率最大的字符对,分配给该桶,直到桶没有剩余空间。

③当一个字符对的频率大于桶的空间 $f(B)$或者桶的剩余空间时,把该字符对分配给该桶,同时,继续在下一个桶分配该字符对。

3.3.3 扰乱函数(Scrambling Function)

字符对经过扁平化后,已经基本上均匀地分布在各个桶中。但是,由于给各桶分配字符对时,是按照频率的大小顺序进行的,那么攻击者可以根据先验知识(如:字符对的频率,桶的个数),推导出字符对与桶的对应关系,即知道哪些字符对分配给哪些桶。为了防止这种攻击,使用一个扰乱函数 S,打乱字符对与桶的对应关系。具体来说,扰乱函数 S 应满足下列性质:

(1)扰乱函数 S 把键值散列为一个随机值;

(2)如果 $k_1 \neq k_2$,则 $S(k_1)\neq S(k_2)$。

性质(1)能够保证,通过散列后,打乱原来桶的顺序,使得攻击者很难分析得到字符对与桶号的对应关系。性质(2)保证了散列后的桶号不会出现冲突,因此各桶所包含字符对的频率总和仍然不变,从而桶中字符对的频率和仍然保持均匀,不会出现大的偏向性。通常来说,既具有散列作用又不会发生冲突的函数是异常稀少的。文献[14]给出了一种乘法散列函数,正好能够满足扰乱函数的上述性质。

定义 2 乘法散列函数。设 ω 是计算机字的大小,它通常是 2^{30} 或 10^{10},A 是与 ω 互素的某个整数常数,且命

$$s(k)=\left\lfloor \left(\frac{A}{\omega}K \bmod 1\right) \right\rfloor \quad (1)$$

则 $S(k)$称为乘法散列函数。

经过乘法散列函数的映射后,$S(K)$在[0,1]范围内取值,按照 $S(K)$的大小顺序,相应地把 m 个桶分别赋予它们。

图 2 给出了字符串特征值的排序、扁平、扰乱的整个提取过程。

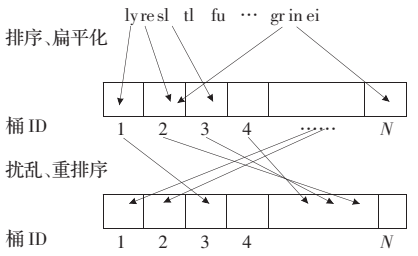


图 2　字符串的映射过程

例 2 对例 1 的字符对进行处理,其中的参数设置为:桶数 $m=16$,$\omega=2^{30}$,可以得到表 1 所示结果,包含各字符对与桶的对应关系。

表 1　字符对与桶的对应关系

字符对	ly	re	Sl	ur	En	Gr	…	el
桶	3	7	15	4	13	7	…	5

3.4 存储

从上面的特征值的提取过程中,可以得到桶与字符对的对应关系,然后根据这种对应关系,在新增的特征属性存储其相应的特征值。

定义 3 设有 s_1 为要加密的字符串 $c_1c_2\cdots c_n$,s_2 为新增特征属性的二进制字符串 $b_0b_1\cdots b_{m-1}$,且 $b_i=0(0\leqslant i\leqslant m-1)$,对于 s_1 中的任何两个相邻字符,都可以从桶与字符对的对应关系发现其对应的桶号,如果桶号为 $i(i\leqslant m)$,那么,$b_i=1$。

例 3 一字符串为"relational",从字符对与桶的对应关系中可以得到该字符串中各字符对所对应的桶,假设"re","el","la","at","ti","io","on","na","al"分别对应桶 7,5,8,13,5,3,11,14,6,则在新增的特征属性值中,把相应的位置为"1",即:"0010 1011 0010 1100"。

3.5 特征值提取和加密数据查询的算法描述

算法 1 特征值提取算法。

输入:各字符对 c_ic_j 和其所对应的频率 f_{ij},桶的个数 m,浮动值 γ。

输出:各桶所分配的字符对。

方法:

(1)把字符对 c_ic_j 按照其频率 f_{ij} 进行降序(或升序)排列,作为一个可分配队列。

(2)计算$\lceil \frac{sum(f)}{m} \rceil$。

(3)把字符对分配到各个桶中,分配时遵循下列步骤:

①初始时,桶可以接收的频率 $avail(B)=\lceil \frac{sum(f)}{m} \rceil+\gamma$。

②从可分配队列中找出最大频率的字符对 c_ic_j,如果 $f_{ij}\leqslant avail(B)$,分配 c_ic_j 给该桶,并把该字符对从分配队列中删除,同时,令 $avail(B)=avail(B)-f_{ij}$;如果 $f_{ij}\geqslant avail(B)$,分配 c_ic_j 给该桶,同时,令 $f_{ij}=f_{ij}-avail(B)$,并选取下一个桶继续分配。

(4)输出各桶所分配的字符对。

算法 2 加密数据的查询算法。

输入:SQL 查询语句,元数据(各桶所分配的字符对)。

输出:返回的数据集。

方法:

第一阶段:粗糙查询(Coarse Query)。

(1)利用元数据(各桶所分配的字符对),转换查询 SQL 中的条件语句。

(2)执行已经转换的 SQL 语句,返回记录,并抛弃过滤字段。

第二阶段:精确查询(Refined Query)。

(1)解密第一阶段返回的记录,存放到一个临时表。

(2)调整 SQL 语句,把原始 SQL 语句中的关系表用临时表替换。

(3)执行调整后的 SQL 语句,得到查询结果。

定理 1 算法 2 的正确性:利用算法 2 对加密数据查询时,返回的数据集是正确的。

证明 从两个方面考虑:一方面,符合查询条件的记录不会被过滤掉。任意一记录,如果明文字段值包含查询条件的字符串,经过扁平-扰乱化的映射后,其对应的特征属性值的相应位数一定被置为 1,根据算法 2 的第一阶段查询,此记录不会被过滤掉,在第二阶段查询中,加密字段被解密后,也必然包括条件字符串,所以也不会过滤掉。另一方面,不符合查询条件的记录一定被过滤掉。数据解密后,在第二阶段的查询中,不符合查询条件的记录一定会被过滤掉。所以说查询处理的结果是正确的。

4 安全性分析

在加密关系模式中,与敏感信息相关的属性有两个,一个是新增的特征属性 X_r^S,存储敏感信息的特征值,另一个是加密属性 X_r^E,存储加密的敏感信息。首先,讨论特征属性的安全性。由于在提取特征值的过程中使用了乘法散列函数,一般来说,攻击者很难通过特征属性直接分析出敏感信息。然而,由于数据库中的属性值较短,且存在许多重复的特点,新增的特征属性将遭受以下两种主要类型的攻击。

(1)统计攻击。假设乘法散列函数分布均匀,当特征属性的位数 m 越大时,即算法 1 中的桶个数 m 越大,不同的字符对散列后,其对应特征属性值上二进制值重复的可能性越少,也就是说,不同的字符对很可能对应着特征属性值中不同的位,攻击者则可以通过统计攻击获取敏感属性的值。

例如,一加密表有 10 万条记录,攻击者可以统计得到特征属性值上二进制值各位出现 1 的累计次数:n_1,n_2,…,n_{32},然后求得其相应的概率:n_1/10 万,n_2/10 万,…,n_{32}/10 万。同时,攻击者也容易得到各字符对在英语中出现的概率。由于各字符在英语中出现具有高度偏向性,其概率具有明显的差异,于是攻击者通过比较两种概率,找出比较接近的概率对,很可能根据特征属性值推断出敏感属性的值。

统计攻击是建立在特征属性值"1"出现次数具有明显偏差性的基础上。采用了扁平化的映射方法后,特征值中"1"的出现次数基本上是均匀的,不会出现大的差异,后面的实验验证了这种情形。所以说采用扁平化的映射方法后,基本上消除了对特征属性的统计攻击,从而保证其安全性。

(2)对比明密文攻击。当特征属性值的位数 m 越大时,不同字符对经过处理后,其特征值不同的概率越大。由于数据库存在大量重复或相近的字段值,攻击者可以通过对比明密文攻击来获取敏感信息。例如,假设攻击者知道某一记录的特征值 e_i 和其对应的明文 p_i,那么,当其它记录的特征值与 e_i 相同时,他可以推断出相应的敏感信息为 p_i。

如果 m 较少时,由于不同的字符对可能对应相同的特征值,那么相应地增加上述攻击的难度。但是 m 太少,将会降低查询性能。从后面的实验可以发现 m 值为 32 位比较合适,既具有较好的安全性,又具有较好的查询性能。

然后,讨论加密属性的安全性。在加密关系模式中,敏感信息通过分组加密算法(如 DES,AES)加密后,以密文形式存在于加密数据库中,一般来说安全性高。但是在数据库这种环境中,不同记录的敏感信息可能存在重复,加密后的数据将遭受字典式攻击和对比明密文攻击。如果直接对敏感信息进行加密,相同的敏感信息经过加密后,得到的密文值仍然相同,那么,攻击者容易通过字典式攻击和对比明密文攻击来获取敏感信息。在本文的加密方法中,采用了把字段值与记录号(rid)连在一起加密,即使敏感信息相同,其对应的密文值也不同,所以,攻击者没有办法对加密数据发动字典式攻击和对比明密文攻击。

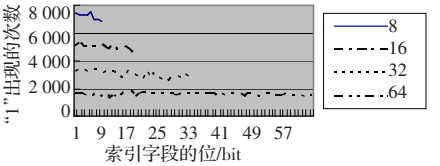
5 实验

实验的目的有两个:一是验证经过扁平化处理后,特征值中各位值不会出现大的偏向性;另一个是验证在查询性能方面,本方法是否较传统的先解密后查询方法优越。

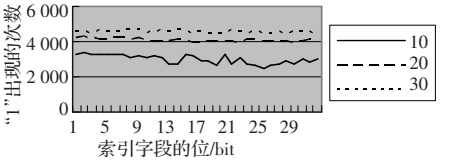
根据 TPC-H 标准[15],利用 dbgen 工具自动生成数据库,数据库的大小为 10 M,也就是比例因子取 0.01。TPC-H 的数据库包括 8 个表,其中,customer 被用来作为本次实验的数据源,它们的字段 comment 需要进行加密处理。加密算法是 safer++,由 C 语言编写,分组长度为 128 位,密钥长度也为 128 位,实验环境为 Windows NT 操作系统,SQL Server 2000 数据库服务器,P4 2.5 GHz 的 CPU,512 MB 的内存。每一个实验数据都做了 10 次,取其平均值而得到。

5.1 特征值的偏向性测试

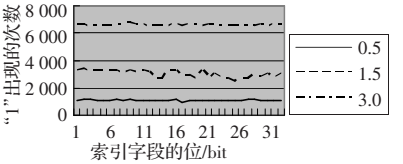
在第一个实验中,测试特征属性中各位值出现"1"的累计次数 k 是否具有大的偏向性。次数 k 与桶的个数 m、明文字符串的长度 L 以及记录数 n 有关,图 3(a)、3(b)和 3(c)分别反映



(a)不同桶数与特征值偏向性关系



(b)不同明文字符串长度与特征值偏向性关系



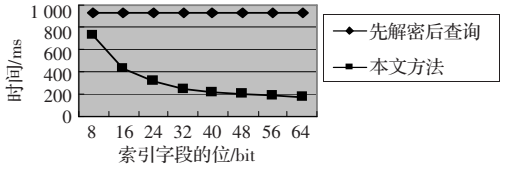
(c)不同记录数与特征值偏向性关系

图 3 特征值的偏向性测试

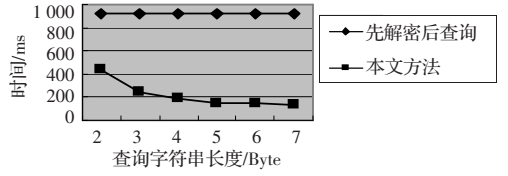
它们与次数 k 的关系,x 轴表示特征属性中的各个位,y 轴表示各位中"1"的累计次数。图 3(a)给出了当桶的个数 m 变化时,对 k 所产生的影响,不同的折线表示不同的桶数 m 对 k 所产生的影响。其中,明文字符串的长度 L=8 Byte,记录数 n=15 000 条。图 3(b)给出了当明文字符串的长度 L 变化时,对 k 所产生的影响。其中桶的个数 m=32,记录数 n=15 000 条,不同的折线表示不同的明文字符串长度 L 对 k 所产生的影响。图 3(c)给出了当记录数 n 变化时,对 k 所产生的影响。其中桶的个数 m=16,明文字符串的长度 L=16 Byte,不同的折线表示不同的记录数 n 对 k 所产生的影响。从图中可以发现,经过扁平化处理后,"1"在各位出现的次数 k 基本上相等,不会有较大的波动,也就是说,索引字段各位值出现"1"的次数 k 没有大的偏向性。

5.2　查询性能测试

在第二个实验中,测试查询加密数据所花费的时间代价,并与传统的先解密后查询方法进行比较,如图 4 所示。其中,SQL 语句为:select*from lineitem where comment like "查询字符串"。



(a)特征值位数与性能关系



(b)查询字符串长度与性能关系

图 4　查询性能测试

图 4(a)和图 4(b)分别给出特征值位数和查询字符串长度不同时,两种查询方法所花费的时间代价,X 轴分别表示需加密字符串的长度和查询字符串的长度,Y 轴表示查询的时间,不同折线表示不同的查询方法。

图 4(a)给出了特征值位数变化时对查询性能的影响,其中,"查询字符串"的长度为 3 Byte。可以发现,随着特征属性的位数增加,查询所花费的时间逐渐减少。这主要是因为在查询过程中,特征值位数越大,第一阶段查询所起到的过滤效率越好,相应地,第二阶段所需要解密和查询的记录越少。正是由于解密的记录数较少,加密数据的查询性能提高了。

图 4(b)给出了"查询字符串"的长度变化时对查询时间的影响,其中,特征属性的位数为 32 位。可以发现,随着"查询字符串"长度增加,查询所花费的时间逐渐减少。很显然,"查询字符串"长度增加了,匹配的精度就越高,那么过滤的效率就越好,从而查询的时间降低。

6　总结

通过在加密关系模式中增加特征属性,利用两阶段的查询方法,实现了加密字符串数据的模糊匹配查询。在安全性方面,由于对字符串数据进行了扁平化和扰乱化处理,能够很好抗击攻击者对加密数据进行攻击;在性能方面,通过减少需要解密的记录数,提高了查询性能。下一步工作对加密数据的多表连接查询、查询优化等方面进行研究。

参考文献:

[1] Lyer B,Mehrotra S,Mykletun E,et al.A framework for efficient storage security in RDBMS[C]//LNCS 2992:the Proc of the 9th International Conference on Extending Database Technology(EDBT),2004:147-164.

[2] Hacigumus H,Lyer B,Mehrotra S.Providing database as a service[C]//Proc of ICDE 2002,2002:29-38.

[3] Chen Y,Chu W W.Database security protection via inference detection[C]//IEEE International Conference on Intelligence and Security Informatics,May 2006.

[4] Sesay S,Yang Zong-kai,Chen Jing-wen,et al.A secure database encryption scheme[C]//Consumer Communications and Networking Conference,Jan 2005:49-53.

[5] Beaver K.Encryption enhancements in SQL Server 2005.Microsoft SQL Server,June 2006.

[6] Bebek G.Anti-tamper database research:inference control techniques,EECS433 final report[R].Case Western Reserve University,2002.

[7] Hacigumus H,Lyer B,Li C,et al.Executing SQL over encrypted data in the database-server-provider model[C]//Proc of ACM SIGMOD,2002:216-227.

[8] Hore B,Mehrotra S,Tsudik G.A privacy-preserving index for range queries[C]//Proc of 30th International Conference on Very Large Databases,Toronto,Canada,2004:720-731.

[9] Agrawal R,Kirenan J,Srikant R,et al.Order-preserving encryption for numeric data[C]//Proc of the ACM SIGMOD,Paris,France,2004:563-574.

[10] Bouganim L,Pucheral P.Chip-secured data access:confidential data on untrusted servers[C]//Proc of 28th International Conference on Very Large Databases,Hong Kong,China,2002:131-142.

[11] Song Dawn Xiaodong,Wagner D,Perring A.Practical techniques for searches on encrypted data[C]//IEEE Symposium on Security and Privacy,Oakland,California,USA 2000:44-55.

[12] Fanghanel T.Using encryption for secure data storage in mobile database systems[D].Friedrich Schiller University,Jena,Germany,September 2002.

[13] Brown H.Considerations in implementing a database management system encryption security solution[R].The Department of Computer Science at the University of Cape Town,2003.

[14] Knuth D E.The art of computer programming volume 3:sorting and searching[M].2nd ed.[S.l.]:Addison-Wesley Publish.

[15] TPC-H.Benchmark specification.http://www.tpc.org.