计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2017.20151125
54(2):348-360,2017

面向多关键字的模糊密文搜索方法

王恺璇　李宇溪　周福才　王权琦
(东北大学软件学院　沈阳　110819)
(wkx_king@126.com)

Multi-Keyword Fuzzy Search over Encrypted Data

Wang Kaixuan, Li Yuxi, Zhou Fucai, and Wang Quanqi
(*Software College, Northeastern University, Shenyang* 110819)

Abstract　Cloud computing is one of the most important and promising technologies. Data owners can outsource their sensitive data in a cloud and retrieve them whenever and wherever they want. But for protecting data privacy, sensitive data have to be encrypted before storing, which abandons traditional data utilization based on plaintext keyword search. Around the multi-keyword fuzzy matching and data security protection problems, we propose a multi-keyword fuzzy search method on the encrypted data. Based on the Bloom filter, our scheme uses dual coding function and the position sensitive Hash function to build file index. In the meantime, it uses the distance recoverable encryption arithmetic to encrypt the file index, consequently achieving the function which is facing the multi-keyword to fuzzy search over the encrypted data. Meanwhile, the scheme does not need to set index storage space in advance, which greatly reduces the complexity of the search. Compared with the existing solutions, the scheme does not need predefined dictionary library which lowers the storage overhead in consequence. Experimental analysis and security analysis show that the proposed scheme not only achieves the multi-keyword fuzzy search over the encrypted data, and guarantees the confidentiality and privacy.

Key words　cloud storage; Bloom filter (BF); searchable encryption; position sensitive Hash function; multi-keyword fuzzy search

摘　要　围绕多关键字的模糊匹配和数据安全性保障问题,展开对多关键字模糊搜索方法的研究,提出一种面向多关键字的模糊密文搜索方案.该方案以布隆过滤器(Bloom filter)为基础,使用对偶编码函数和位置敏感 Hash 函数来对文件索引进行构建,并使用距离可恢复加密算法对该索引进行加密,实现了对多关键字的密文模糊搜索.同时方案不需要提前设置索引存储空间,从而大大降低了搜索的复杂度.除此之外,该方案与已有方案相比不需要预定义字典库,降低了存储开销.实验分析和安全分析表明,该方案不仅能够实现面向多关键字的密文模糊搜索,而且保证了方案的机密性和隐私性.

收稿日期:2015-12-21;**修回日期**:2016-10-10
基金项目:国家科技重大专项基金项目(2013ZX03002006);辽宁省科技攻关项目(2013217004);中央高校基本科研业务费专项资金(N130317002);沈阳市科技基金项目(F14-231-1-08)
This work was supported by the Major National Science and Technology Program (2013ZX03002006), the Liaoning Province Science and Technology Project (2013217004), the Fundamental Research Funds for the Central Universities (N130317002), and the Shenyang Science and Technology Project (F14-231-1-08).
通信作者:周福才(fczhou@mail.neu.edu.cn)

关键词 云存储;布隆过滤器;可搜索加密机制;位置敏感 Hash 函数;多关键字模糊搜索

中图法分类号 TP309.2

随着云存储的迅速发展以及用户对个人数据隐私性的愈加重视,如何对存储在服务器中的密文进行搜索就显得格外重要.可搜索加密方法(searchable encryption, SE)是解决密文搜索的有效方法.

可搜索加密方法首次由 Song 和 Wagner 等人[1]提出,开创了将搜索机制应用于关键字密文搜索的先河;2005 年,Chang 等人[2]提出了基于预定义字典的可搜索加密机制.这种机制不仅优化了关键字的搜索效率,而且还缩减了计算开销.但是,由于预定义字典的设定,同时也给用户带来了额外的存储开支.随后 Dong 等人[3]在 2008 年针对安全性进行了改进,提出了能够抵抗自适应性攻击的搜索加密模型.2015 年,Revathy 等人[4]提出了排序的搜索加密机制.以上几种搜索关键字均是基于对称密码学的精确的单关键字搜索机制,用户只能在一次搜索过程中发送 1 个单词,这种搜索机制与现实的搜索需求极为不符.

2014 年,Cao 和 Wang 等人[5]提出了多关键字可搜索加密机制.该机制在搜索时为索引和关键字构建向量,并通过向量运算实现了搜索结果的排序.2016 年,文献[6]提出了多关键字的排序搜索方案,该方案利用树构建索引结构,减少了搜索时间.

从以上方案可以看出,不论是面向单关键字还是多关键字,现有研究都是针对精确搜索的.而针对用户输入错误关键字的情况,精确的单关键字搜索加密机制或者基于连接词的搜索加密机制还需要改进.研究学者在精确搜索的基础上相继提出了基于关键字的模糊搜索方案.Bringer 等人[7]在 2009 年提出了面向明文的模糊搜索方案,该方案主要基于布隆过滤器实现数据存储.Van Liesdonk 等人[8]在 2010 年研究了模糊关键字搜索加密方案,但这种方案最大的缺陷是需要用户额外花费时间来执行关键字的循环搜索.2010 年,文献[9]实现了关键字模糊搜索的加密机制,但该方案需要用户付出庞大的数据存储空间,且只实现了基于单关键字的模糊密文搜索.之后在 2013 年,文献[10-11]也相继提出了不同的模糊关键字搜索方法.其中,文献[10]给出了单关键字可排序模糊关键字搜索方法,文献[11]提出的是一种可验证的模糊关键字搜索方案.

本文针对密文的多关键字模糊搜索展开研究,以提高搜索效率为目标,提出了一种面向多关键字的模糊密文搜索方案,该方案利用布隆过滤器和位置敏感 Hash 函数技术,能够有效实现多关键字的密文模糊搜索.主要研究内容包括:将上传文件利用对偶编码函数将其关键字转换成向量,再利用位置敏感 Hash 函数将其映射到布隆过滤器中;服务器在执行密文搜索时,需要先对输入的查询关键字进行上述转换,再通过计算安全索引参数和陷门的内积来搜索到符合条件的密文文件.与已有方案相比,本方案不需要提前设置索引存储空间,从而大大降低了搜索的复杂度;同时,该方案不需要预定义字典库,降低了存储开销.安全性分析表明方案不仅保证了可用性,而且满足了机密性和隐私性.因而,该方案具有重要的理论价值和应用前景.

1 预备知识

1.1 位置敏感 Hash 函数

位置敏感 Hash 函数于 1998 年由 Indyk 等人[12]提出,主要用于解决高维数据的搜索问题.

定义 1. 位置敏感 Hash 函数.给定 n 个 R^d 维的点集 S 以及各点之间的相似性度量 D,Hash 族 $H=\{h:H\rightarrow U\}$ 被称为 (r_1,r_2,p_1,p_2)-敏感 Hash 函数,需要满足以下条件,对于 $q\in S$:

如果 $v\in d(q,r_1)$,那么 $Pr[h(q)=h(v)]\geqslant p_1$;

如果 $v\notin d(q,r_1)$,那么 $Pr[h(q)=h(v)]\leqslant p_2$.

其中,$r_1<r_2$,$p_1\geqslant p_2$,$d(q,r_1)$ 是距离度量.

本文方案将使用文献[13]中的基于 p-stable 的位置敏感 Hash 函数.p-stable 分布可以有效对高维向量进行降维.随机构建 1 个 d 维的向量 v,如果向量中的每个元素均满足 p-stable 分布,随机变量 $a\cdot v$ 和 $\left(\sum_{i=1}^{n}|v_i|^p\right)^{1/p}X$ 同分布,则 $a\cdot v$ 和 $\|v\|p$ 的值大致相同.位置敏感 Hash 函数的计算公式如下:

$$h_{a,b}(v)=\left\lfloor\frac{a\cdot v+b}{w}\right\rfloor, \tag{1}$$

其中,参数 a 和 b 分别是 d 维的向量和 $[0,w]$ 间的一个随机数,向量 a 中的每个元素都满足 p-stable 分布.

式(1)的计算过程实际是向量的映射过程,如图 1 所示.$a\cdot v$ 即把向量 a 映射到以向量 v 为基向量的数轴上,如果将此数轴等分为 w 份,同时每份

做上标记，$a \cdot v$ 的值对应哪个区间就将这个区间的标记号作为向量 a 的 Hash 函数值.

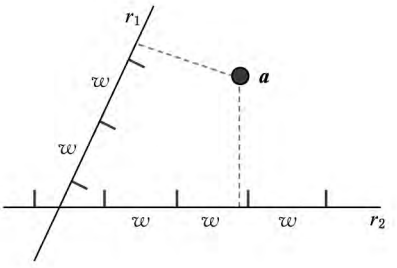

Fig. 1 p-stable distribution definition

图 1 p 稳定分布定义图

判断 2 点是否可以通过位置敏感 Hash 函数映射到 1 个桶中，主要通过计算 Hash 值相等的概率来进行. 假设给定 2 个向量 v_1, v_2，让 $c=\|v_1-v_2\|_p$，那么向量 v_1, v_2 的位置敏感 Hash 函数值相等的概率如下：

$$p(c)=Pr_{a,b}[h_{a,b}(v_1)=h_{a,b}(v_2)]=\int_0^w \frac{1}{c}f_p\left(\frac{t}{c}\right)\left(1-\frac{t}{w}\right)\mathrm{d}t,$$

其中，f_p 是分布 D 的密度函数，$p_1=p(r_1)$, $p_2=p(r_2)$.

1.2 布隆过滤器

Bloom[14]在 1970 年提出了布隆过滤器(Bloom filter, BF)的概念. BF 是一种概率型数据结构，主要用于判断某个元素是否存在于集合中.

布隆过滤器的主要工作原理是：运用一个 m(单位为 b)长的数组表示这个布隆过滤器，数组的每一位元素初始化为 0. 随后随机选择 k 个 Hash 函数，将数据域 $S=\{y_1, y_2, \cdots, y_n\}$ 中的 n 个元素分别一一映射到上面的数组中，当需要映射 1 个元素时，计算每个元素的 k 个 Hash 函数对应的 k 个 Hash 值，然后把数组中对应的位置设置为 1. 如果 Hash 值对应的位置已经设置成为 1，就保留第 1 次作用的效果. 如图 2 所示：

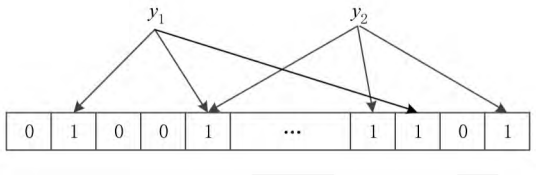

Fig. 2 The sample figure of Bloom filter

图 2 布隆过滤器示例图

判断元素 x 是否存在于数据域中，计算元素 x 对应的上面的 k 个 Hash 函数的 Hash 值，如果该元素的所有 Hash 值对应的位置都为 1，就说明元素 x 存在于数据域中.

向布隆过滤器中加入 1 个新的元素时，需要使用 k 个不同的 Hash 函数对其进行 Hash 计算，将这个元素映射到布隆过滤器的 k 个比特位中，同时将这些比特位设置为 1.

查询某个元素是否存在于布隆过滤器中时，需要利用提前定义的 k 个 Hash 函数将其映射到 BF 的 k 个比特位上. 如果这 k 个比特位对应的值都是 1，则说明该元素存在于集合中，只要有一个位置不为 1，则此元素不存在于该集合中.

1.3 对偶编码函数

在面向密文的多关键字模糊搜索方案中，构建索引、构建陷门和关键字查询的过程都是基于向量的操作过程. 数据拥有者输入的关键字都由字符组成，由于字符的不可计算性，需要将其转换成向量的形式.

定义 2. 对偶编码函数. 给定字符串 $\boldsymbol{S}_1=C_1C_2\cdots C_n$ 和二进制向量 $\boldsymbol{S}_2=b_0b_1\cdots b_{m-1}$，$\boldsymbol{S}_2$ 中每位元素的初始值为 0，其中 $n<m$. 通过 Hash 函数 H，把 $\boldsymbol{S}_1$ 中相邻 2 个字符散列映射为 0～$(m-1)$ 之间的数，当且仅当 $H(C_j, C_{j+1})=i$ 时，$b_i=1$. 把 $\boldsymbol{S}_1 \rightarrow \boldsymbol{S}_2$ 的函数称为对偶编码函数.

1.4 距离可恢复加密方案

距离可恢复加密算法可以将数据库中的各个元素表示成 1 个多维的点，其中属性作为该点的维度，属性对应的值作为该维度上的值.

定义 3. 距离可恢复加密(distance recovery encryption, DRE)[15-16]. 给定加密函数 E 以及加密密钥 K，则数据库 DB 中的点 p 对应的密文为 $E(p, K)$. 当且仅当 E 满足以下条件，E 可称为距离可恢复加密. 对于任意 2 点 p_1, p_2，以及密钥 K，存在函数 f，使得

$$f(E(p_1,K),E(p_2,K))=d(p_1,p_2)$$

成立. 其中 $d(p_1, p_2)$ 为 p_1, p_2 的距离. 如果函数 f 是欧氏距离计算函数，则 E 被称为距离保留转换函数.

2 多关键字模糊密文搜索方案

本节将描述该方案的模型、形式化和安全性定义以及方案的详细设计.

2.1 模 型

如图 3 所示，方案中涉及 3 个实体：1)数据拥有者. 该实体把数据外包给服务器进行加密存储. 2)可信赖用户. 该实体并不持有数据但是可以对数据进

行搜索,最终获取需要的数据. 3)服务器. 此实体提供存储服务,存储用户的密文数据,并在收到用户的搜索请求时,执行密文的搜索操作.

数据拥有者在本地选择待存储的文件集,对其进行处理后上传到服务器端. 文件集的处理包括2个部分:1)对文件进行加密,之后生成相应的密文;2)为待存储的文件设置索引,利用预定义的加密算法对其进行加密,最终生成加密索引. 数据拥有者需要将处理之后的文件上传至服务器,由于存储文件和索引的工作由服务器负责,所以数据拥有者只需要执行上传操作即可,不需要再关心具体的存储细节.

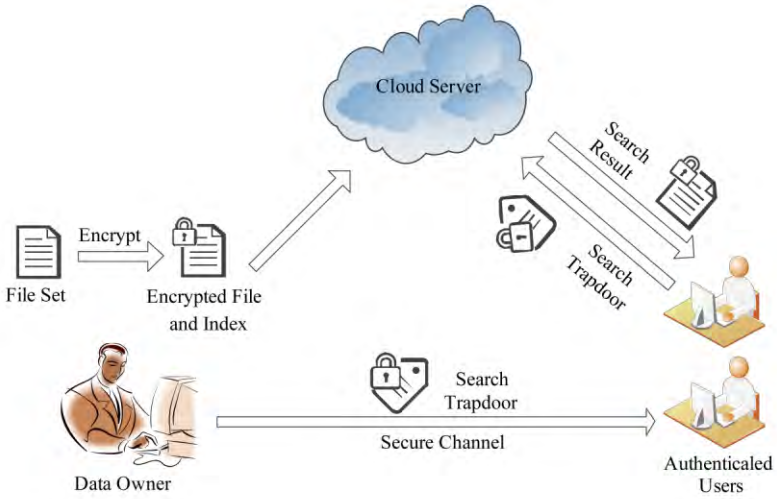

Fig. 3 Basic architecture

图3 方案基本架构

各个实体的功能介绍如下:

1) 数据拥有者. 它是一个特殊的可信赖用户,因为具体的明文是由数据拥有者确定,所以数据拥有者是3个实体中唯一能够与明文直接接触的实体. 数据拥有者首先选择明文数据,然后加密这些明文,同时构建这些明文相对应的索引,最后把密文以及加密后的索引发送给云服务器进行存储. 数据拥有者还需要选择可信赖用户(包括自己),并向其发送搜索陷门.

2) 可信赖用户. 即数据拥有者授权的可信实体. 它主要接收数据拥有者发送的搜索陷门,然后生成安全陷门,向服务器端发送搜索请求. 接收到服务器发送回来的搜索结果后,对搜索结果进行解密,从而获取需要的明文数据.

3) 服务器. 它主要进行数据密文以及索引密文的存储,同时对于数据拥有者以及可信赖用户发来的搜索请求进行相应的搜索以及应答.

对于基于布隆过滤器以及位置敏感Hash函数实现的面向多关键字的模糊密文搜索方案,在该方案中不需要设置预定义字典存储错误的关键字. 根据上述描述提出面向多关键字的模糊密文搜索方案的基本架构.

2.2 形式化定义

本方案主要由4个主要算法组成:初始化算法、生成索引算法、生成陷门算法、搜索算法. 形式化地表示为 $MKFS=(KeyGen, BuildIndex, Trapdoor, Search)$. 下面分别对其进行具体的描述.

1) $sk \leftarrow KeyGen(1^k)$:初始化算法. 运行于数据拥有者端,主要用于生成密钥. 输入安全参数 1^k,输出密钥 sk. 加密索引和查询关键字都需要该密钥. 如果该密钥丢失,数据拥有者将无法从服务器端再获取目标数据.

2) $I \leftarrow BuildIndex(sk, F, W)$:生成索引算法. 运行于数据拥有者端,主要用于生成文件对应的安全索引. 输入密钥 sk、文件标记序号 F 以及文件 F 对应的关键词集合 δ,输出安全索引 I.

3) $t \leftarrow Trapdoor(sk, Q)$:生成陷门算法. 运行于用户端,用于加密查询关键字集合生成安全陷门. 输入密钥 sk 以及查询关键词集合 Q,输出安全陷门 t. 用于加密查询关键词集合 Q,输出安全陷门 t.

4) $F_R \leftarrow Search(I, t)$:搜索算法. 运行于第三方服务器端. 输入安全索引和安全陷门,输出文件序列集合. 主要进行安全索引与安全陷门的匹配,如果匹配成功就返回该索引对应的文件标识符;如果匹配不成功,则无返回结果.

当服务器端搜索到满足搜索条件的结果时,就将结果集合返回给用户,用户需要利用数据拥有者传送的搜索陷门进行解密,最后得到需要的目标文件.

2.3 安全性定义

根据本文提出的方案,在给出安全模型之前,首先定义以下几个与安全性模型相关的概念,方便后续的安全性定义.

定义 4. 数据拥有者所持有的所有明文信息以及查询关键字归纳定义为明文信息集 $H=(F,W,Q)$. 其中 F 为文件集合,W 为各文件对应的关键字集合,Q 为查询关键字集合.

定义 5. 明文信息集 H 加密生成的密文信息集 $V(H)$. 该集合包括文件集合 F 对应的密文集合 $Enc_{sk}(F)$,关键字集合 W 加密后对应的安全索引集合 I 和查询关键字集合 Q 加密后对应的安全陷门 t.

定义 6. 将服务器能够获取的信息统称为足迹集合 $Tr(H)$. $Tr(H)$主要由服务器搜索到的结果组成,即 $Tr(H)=\{Tr(w_1),Tr(w_2),\cdots,Tr(w_t)\}$.

为了更好更直观地评估搜索方案的安全性,本文根据敌手能够获取的资源 S_A 将攻击模型分为 3 个级别.

级别 1. 已知密文模型. 在这个安全级别中,敌手只能获取密文信息 $V(H)$,即 $S_A=\langle V(H)\rangle$. 实际应用中,服务器除了能够获取密文资源外,无法获取其他任何的资源.

级别 2. 已知密文与某些明文. 在这个安全级别中,敌手除了能够获取密文资源,还能够获取一组明文信息 $H'\in H$,但是敌手无法确认这些信息所对应的密文,即 $S_A=\langle V(H),H'\rangle$.

级别 3. 已知密文与某些明文以及这些明文对应的密文. 在这个攻击模型中,敌手除了能够获取密文资源、一组明文信息 $H'\in H$,还能够获取这些明文信息对应的密文 $V(H')$,即 $S_A=\langle V(H),H',V(H')\rangle$.

上述 3 个级别中,级别越高说明该方案安全性越高,所以如果一个加密方案能够抵抗级别高的攻击模型,那么该模型也能够抵抗级别低的模型. 在上诉定义的 3 个级别的安全模型中,级别 2 与现实环境更接近. 由于实际应用中,敌手无法在不知道密钥的情况下,由密文关联明文,所以级别 3 在实际应用中很少会发生.

在上述安全模型中,一个能够保证数据安全性的密文搜索方案应该满足以下安全特性:

1) 文件隐私性. 文件隐私性表示存储在第三方服务器中的密文文件不应向服务器泄露任何与明文文件相关的信息.

2) 关键字隐私性. 服务器能够获取的不仅是搜索结果,还能够获取安全索引以及安全陷门. 关键字隐私性要求服务器无法通过安全索引以及安全陷门获取任何与关键字相关的信息.

3) 陷门不可链接性. 服务器无法将获取到的一个安全陷门扩展到另一个安全陷门,即进行 2 次搜索操作时,输入同一组关键词,服务器无法对这 2 次生成的陷门进行区分.

除此之外,提出面向多关键字的模糊密文搜索方案能够抵抗选择明文攻击. 方案的安全性证明主要是敌手和用户/模拟器之间的交互性游戏.

定义 $Real_{S_1}(sk)$为模拟器 S_1 执行真实方案的 1 次实验,在整个实验过程中均使用真实的方案算法. 用户首先执行初始化算法,然后分别根据敌手需要挑战的内容生成不同的内容,并保存已获取的信息构成 V_{Real}. 该实验最终输出 1 个比特 b 作为输出结果.

定义 $Ideal_{S_2}(sk)$为模拟器 S_2 的 1 次模拟仿真实验. 与 $Real_{S_1}(sk)$不同的是 $Ideal_{S_2}(sk)$实验中并不运行真实的算法,而是把真实算法中服务器可以获取的资源作为 $Ideal_{S_2}(sk)$实验的输入,然后模拟器通过生成随机数据作为实验输出. 同 $Real_{S_1}(sk)$一样,S_2 收集所有已获取的信息构成 V_{Ideal}. 该实验最终输出 1 个比特 b'作为输出结果.

选择关键字安全性指的是,存在任意模拟器,使得任意敌手都无法将 V_{Ideal}和 V_{Real}区分开来. 即敌手无法真正区分整个实验过程中到底是真实的用户还是模拟器在与其进行交互. 下面将给出面向多关键字的模糊密文搜索方案的安全模型的形式化定义.

定义 7. 自适应选择关键字安全性. 给定一个面向多关键字的模糊密文搜索方案,令 A 为一个敌手,S_1,S_2 为 2 个模拟器. 函数 G_1,G_2,G_3 分别是在生成密文阶段、生成安全索引阶段、生成陷门阶段中实验 $Real_{S_1}(sk)$和实验 $Ideal_{S_2}(sk)$之间互相共享的信息,即实验 $Real_{S_1}(sk)$的泄露函数. 下面是实验的交互过程:

实验 $Real_{S_1}(sk)$和实验 $Ideal_{S_2}(sk)$在文件加密阶段的形式化定义如下:

$Real_{S_1}(sk)$:

$sk\leftarrow KenGen(1^k)$,

$F\leftarrow S_1(m)$,

$X\leftarrow Enc(F,sk:S_1)$,

output $b_1 \leftarrow S_1(X)$.

$Ideal_{S_2}(sk)$:

$sk' \leftarrow KenGen(1^k)$,

$F' \leftarrow S_2^{G_1}$,

$X' \leftarrow Enc(F', sk': S_2)$,

output $b_1' \leftarrow S_2(X')$.

实验 $Real_{S_1}(sk)$ 和实验 $Ideal_{S_2}(sk)$ 在生成陷门阶段的形式化定义如下:

$Real_{S_1}(sk)$:

$sk \leftarrow KenGen(1^k)$,

$(l) \leftarrow S_1$,

$Q \leftarrow S_1(l)$,

$t \leftarrow BuildTrapdoor(Q, sk: S_1)$,

output $b_2 \leftarrow S_1(t)$.

$Ideal_{S_2}(sk)$:

$sk' \leftarrow KenGen(1^{k'})$,

$Q' \leftarrow S_2^{(G_1, G_2)}$,

$t' \leftarrow BuildTrapdoor(Q', sk': S_2)$,

output $b_2' \leftarrow S_2(t')$.

实验 $Real_{S_1}(sk)$ 和实验 $Ideal_{S_2}(sk)$ 在生成安全索引阶段的形式化定义如下:

$Real_{S_1}(sk)$:

$sk \leftarrow KenGen(1^k)$,

$(f_i, t) \leftarrow S_1$,

$W_i \leftarrow S_1(t, f_i)$,

$I \leftarrow BuildIndex(f_i, W_i, sk: S_1)$,

output $b_3 \leftarrow S_1(I)$.

$Ideal_{S_2}(sk)$:

$sk' \leftarrow KenGen(1^{k'})$,

$W_i' \leftarrow S_2^{(G_1, G_2, G_3)}$,

$I' \leftarrow BuildIndex(f_i, W_i', sk': S_2)$,

output $b_3' \leftarrow S_2(I')$.

在前文已对敌手具有的能力进行了分析,并且定义了方案在安全目标为不可区分性的条件下的安全模型. 下面利用敌手和挑战者之间的攻击游戏来定义面向多关键字的模糊密文搜索方案的安全性.

1) 挑战密文

① 初始阶段. 挑战者执行初始化算法 $KenGen(1^k)$,密钥 sk 作为该阶段的输出结果.

② 挑战阶段. 敌手 A 选择想要挑战的明文信息 F,并将其发送给挑战者,挑战者在接收到明文信息后利用密钥 sk 完成文件加密阶段的真实性实验 $Real_{S_1}(sk)$,然后利用实验 $Real_{S_1}(sk)$ 中的泄露函数 G_1 完成模拟实验 $Ideal_{S_2}(sk)$,最后挑战者随机选择其中一个实验的结果 $\hat{b}$ 返回给敌手.

③ 猜测. 最后,敌手输出对结果 $\hat{b}$ 的 1 个猜测 b''.

如果完成上述游戏后,对所有的敌手 A,均存在一个模拟器 S,使:

$$|Pr[b''=1]-Pr[\hat{b}=1]| \leqslant \frac{1}{p(sk)},$$

其中 $p(sk)$ 是以 sk 为输入的多项式,那么面向多关键字的模糊密文搜索方案针对密文是选择明文攻击安全的.

2) 挑战安全陷门

① 初始阶段. 挑战者执行初始化算法 $KenGen(1^k)$,密钥 sk 作为该阶段的输出结果.

② 挑战阶段. 敌手 A 选择想要挑战的查询关键字 Q,并将其发送给挑战者,挑战者在接收到查询关键字后利用密钥 sk 进行试验 $Real_{S_1}(sk)$,生成安全陷门,然后利用实验 $Real_{S_1}(sk)$ 中的泄露函数 G_1,G_2 完成模拟实验 $Ideal_{S_2}(sk)$,最后挑战者随机选择其中一个实验的结果 $\hat{b}$ 返回给敌手.

③ 猜测. 最后,敌手输出对结果 $\hat{b}$ 的一个猜测 b''.

3) 挑战安全索引

① 初始阶段. 挑战者执行初始化算法 $KenGen(1^k)$,密钥 sk 作为该阶段的输出结果.

② 挑战阶段. 敌手 A 选择想要挑战的安全索引,将该索引对应的泄露函数 G_1, G_2, G_3 发送给挑战者,挑战者在接收到泄露函数后利用密钥 sk 首先进行试验 $Real_{S_1}(sk)$,生成安全索引,然后利用实验 $Real_{S_1}(sk)$ 中的泄露函数 G_1 完成模拟实验 $Ideal_{S_2}(sk)$,最后挑战者随机选择某个实验的输出结果 $\hat{b}$ 返回给敌手.

③ 猜测. 最后,敌手输出对结果 $\hat{b}$ 的 1 个猜测 b''.

如果敌手能够以可忽略的概率猜出 $\hat{b}$ 的值,则说明该方案中构建索引方案和生成陷门的方案是选择关键字攻击安全的.

2.4 算法详细设计

2.4.1 密钥生成算法

$sk \leftarrow KeyGen(1^k)$:该算法用于方案的初始化、生成密钥,其中 k 需要取足够长,例如 128 b,这样才能够有效阻止蛮力攻击. 该算法的主要步骤如下:

1) 随机构建 2 个 $k \times k$ 维的矩阵 $\boldsymbol{M}_1, \boldsymbol{M}_2 \in \mathrm{R}^{k \times k}$;

2) 随机构建 1 个 k 维的向量 $\boldsymbol{S} \in \{0,1\}^k$,为了使引入参数向量 $\boldsymbol{S}$ 的随机性最大化,$\boldsymbol{S}$ 中 0 和 1 的数量需要大致相同.

3) 输出 $sk=(\boldsymbol{M}_1,\boldsymbol{M}_2,\boldsymbol{S})$作为生成加密索引和生成陷门的密钥.

2.4.2 生成安全索引算法

$I\leftarrow BuildIndex(sk,F,W)$:该算法用于加密用户输入的索引.主要步骤如下:

1) 对于文件集合 $F=\{f_1,f_2,\cdots,f_n\}$中的每一个文件 f_i:

① 为文件 f_i 构建一个 k 位的 Bloom 过滤器 $\boldsymbol{B}_i$;

② 对于文件 f_i 所对应的关键字集合 $W_i=\{w_1,w_2,\cdots,w_t\}$,使用对偶编码函数将其转换成向量集合 $V=\{\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_t\}$,其中每个向量 $\boldsymbol{v}_i\in\{0,1\}^{676}$;

③ 对于向量集合 $V=\{\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_t\}$中的每个向量 $\boldsymbol{v}_i$,使用位置敏感 Hash 函数$\{H_1,H_2,\cdots,H_l\}$进行计算$(H_1(\boldsymbol{v}_i),H_2(\boldsymbol{v}_i),\cdots,H_l(\boldsymbol{v}_i))$,并将结果插入到 Bloom 过滤器 $\boldsymbol{B}_i$ 中.

2) 将私钥 sk 中的 $\boldsymbol{S}$ 表示为 $\boldsymbol{S}=(s_1,s_2,\cdots,s_k)$.对于每一个文件 f_i 所得到的 Bloom 过滤器 $\boldsymbol{B}_i$,

① 随机选择参数 $r\in\mathrm{R}$.

② 将 $\boldsymbol{B}_i$ 表示为 $\boldsymbol{B}_i=(b_1,b_2,\cdots,b_k)$.$\boldsymbol{B}_i$ 与 $\boldsymbol{S}$ 具有同样的结构,均为由$\{0,1\}$组成的 k 位向量.对于 $\boldsymbol{B}_i$ 中的每一个 b_j,若在 $\boldsymbol{S}$ 中对应的 $s_j=1$,则令 $b'_j=b''_j=b_j$;若对应的 $s_j=0$,则令

$$b'_j=\frac{1}{2}b_j+r,\ b''_j=\frac{1}{2}b_j-r.$$

③ 令 $\boldsymbol{B}'_i=(b'_1,b'_2,\cdots,b'_k)$,$\boldsymbol{B}''_i=(b''_1,b''_2,\cdots,b''_k)$.

④ 计算 $I'_i=\boldsymbol{M}_1^{\mathrm{T}}\cdot\boldsymbol{B}'_i$, $I''_i=\boldsymbol{M}_2^{\mathrm{T}}\cdot\boldsymbol{B}''$.

⑤ 令 $I_i=(I'_i,I''_i)$.

3) 输出 $I=(F,I_1,I_2,\cdots,I_n)$.

根据上述生成安全索引的步骤,给出一组具体的示例.给定 3 个关键词:Network,Search,Bloom.先将这 3 个关键词转换成 676 B 长的数组表示,其中每个元素都表示每 2 个字符的组合,然后利用选择的 l 个位置敏感 Hash 函数(这个示例中 $l=2$),计算单个关键词在布隆过滤器中对应的位置,最后该对应的位置置位为 1.该示例的具体操作如图 4 所示:

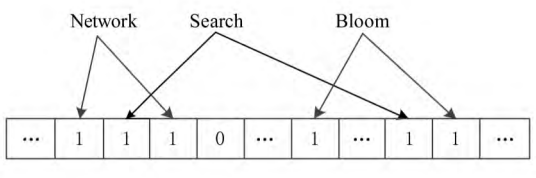

Fig. 4 Sample of generation security index

图 4 生成安全索引示例

2.4.3 陷门算法

$t\leftarrow Trapdoor(sk,Q)$:根据查询关键词集合生成安全陷门.主要步骤如下:

1) 构建 1 个 k 位的 Bloom 过滤器 $\boldsymbol{B}$.

2) 对于查询关键字集合 $Q=\{q_1,q_2,\cdots,q_t\}$,使用对偶编码函数将其转换成向量集合 $V=\{\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_t\}$,其中每个向量 $\boldsymbol{v}_i\in\{0,1\}^{676}$.

3) 对于向量集合 $V=\{\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_t\}$中的每个向量 $\boldsymbol{v}$,使用位置敏感 Hash 函数$\{H_1,H_2,\cdots,H_l\}$进行计算,并将结果插入到 Bloom 过滤器 $\boldsymbol{B}$ 中.

4) 将私钥 sk 中的 $\boldsymbol{S}$ 表示为 $\boldsymbol{S}=(s_1,s_2,\cdots,s_k)$.对于 Bloom 过滤器 $\boldsymbol{B}$:

① 随机选择数 $r'\in\mathrm{R}$.

② 将 $\boldsymbol{B}$ 表示为 $\boldsymbol{B}=(b_1,b_2,\cdots,b_k)$.对于 $\boldsymbol{B}$ 中的每一个 b_j,若在 $\boldsymbol{S}$ 中对应的 $s_j=0$,则令 $b'_j=b''_j=b_j$;若对应的 $s_j=1$,则令

$$b'_j=\frac{1}{2}b_j+r',\ b''_j=\frac{1}{2}b_j-r'.$$

③ 令 $\boldsymbol{B}'=(b'_1,b'_2,\cdots,b'_k)$,$\boldsymbol{B}''=(b''_1,b''_2,\cdots,b''_k)$.

④ 计算 $t'=\boldsymbol{M}_1^{-1}\cdot\boldsymbol{B}'$, $t''=\boldsymbol{M}_2^{-1}\cdot\boldsymbol{B}''$.

⑤ 令 $t=(t',t'')$.

5) 输出陷门 t.

针对上述给出的生成陷门的具体步骤,给定 2 个查询关键词对上述的步骤进行形象化的表示.假设给定的 2 个查询关键词为 Netword,Search.对比索引示例图中的关键词,Netword 与 Network 之间仅仅相差了 1 个字符,但是通过同一组位置敏感 Hash 函数计算后 Netword 在布隆过滤器中对应的位置与 Network 的位置相同.下面给出示例中 2 个查询关键词的布隆过滤器表示,如图 5 所示:

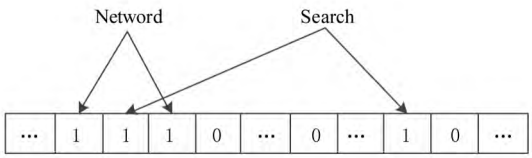

Fig. 5 Example of generating the trapdoor

图 5 生成陷门示例

2.4.4 搜索算法

$F_R\leftarrow Search(I,t)$:服务器根据陷门和加密索引执行搜索.主要步骤如下:

1) 令 F_R 为一个初始为空的文件集合.

2) 将 I 表示为 $I=(F,I_1,I_2,\cdots,I_n)$.对于其中的每一个 I_i:

这里的 Hi(v) 指的是 P-Stable 算法返回的一组值

S_j=0/1时，索引列表I和陷门T的b'和b''相反，
S_j =1，b_ij'=1/2*b_j+r,b_ij''=1/2*b_j-r。
S_j =0，陷门T的 b_j'' = 1/2*b_j+r 和b_j''=1/2*b_j-r。

① 将 I_i 表示为 $I_i=(I_i',I_i'')$，将 t 表示为 $t=(t',t'')$.

② 计算向量的数量积

$$R_I=I_i'\cdot t'+I_i''\cdot t''. \tag{2}$$

③ 按排序算法将计算的向量内积作排序，将排序序列中的前 λ 条记录加入到结果集合 F_R 中.

3) 输出 F_R 作为最终搜索结果.

图 6 是通过 2.4.2～2.4.3 节中给出的生成安全索引的示例以及生成陷门的示例给出的最终搜索示例结果，按照上述搜索步骤，实际服务器执行的就是 2 个向量之间的内积，然后获取内积结果，找出对应的文件标识.

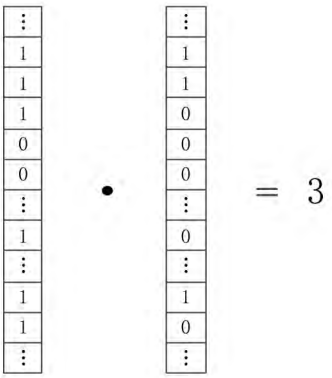

Fig. 6 Retrieval figure

图 6 搜索示例图

$\boldsymbol{B}_i$ 中的每一个 b_j，若在 $\boldsymbol{S}$ 中对应的 $s_j=1$，则令 $b_j'=b_j''=b_j$；若对应的 $s_j=0$，则令 $b_j'=\frac{1}{2}b_j+r$，$b_j''=\frac{1}{2}b_j-r$.

B里面存放的是布隆过滤器k个位所对应的0/1，B'和B''存放的是根据随机k维向量S中位的取值来判断，若在S 中对应的s_j=1则令

为了确保索引以及查询关键词经过上述加密后能否返回正确的结果，只要保证 $R_I=\boldsymbol{B}_i\cdot\boldsymbol{B}$ 就说明该搜索方案能够实现面向多关键字的模糊密文搜索. 下面就给出上述过程的正确性推导：

1) $I_i=(I_i',I_i'')$，其中 $I_i'=\boldsymbol{M}_1^{\mathrm{T}}\cdot\boldsymbol{B}_i'$，$I''=\boldsymbol{M}_2^{\mathrm{T}}\cdot\boldsymbol{B}_i''$.

2) $t=(t',t'')$，其中 $t'=\boldsymbol{M}_1^{-1}\cdot\boldsymbol{B}'$，$t''=\boldsymbol{M}_2^{-1}\cdot\boldsymbol{B}''$.

3) $R_I=I_i\cdot t$，安全索引与陷门作内积运算.

4) 将 $I_i=(\boldsymbol{M}_1^{\mathrm{T}}\cdot\boldsymbol{B}_i',\boldsymbol{M}_2^{\mathrm{T}}\cdot\boldsymbol{B}_i'')$ 和 $t=(\boldsymbol{M}_1^{-1}\cdot\boldsymbol{B}',\boldsymbol{M}_2^{-1}\cdot\boldsymbol{B}')$ 代入 $R_I=I_i\cdot t$，则

$$R_I=\boldsymbol{M}_1^{\mathrm{T}}\cdot\boldsymbol{B}_i'\cdot\boldsymbol{M}_1^{-1}\cdot\boldsymbol{B}'+\boldsymbol{M}_2^{\mathrm{T}}\cdot\boldsymbol{B}_i''\cdot\boldsymbol{M}_2^{-1}\cdot\boldsymbol{B}''=\boldsymbol{B}_i'\cdot\boldsymbol{B}'+\boldsymbol{B}_i''\cdot\boldsymbol{B}''.$$

5) 结果集 $R_I=\{r_{i,j}\mid 1\leqslant i\leqslant t,1\leqslant j\leqslant t\}$，当 $s_j=1\in\boldsymbol{S}$ 时，$b_{i,j}'=b_{i,j}''=b_{i,j}$，$b_j'=\frac{1}{2}b_j+r'$，$b_j''=\frac{1}{2}b_j-r'$. 将上述参数取值代入式(2)，那么，

$$r_{i,j}=b_{i,j}'\cdot b_j'+b_{i,j}''\cdot b_j''=b_{i,j}\cdot\left(\frac{1}{2}b_j+r'\right)+b_{i,j}\cdot\left(\frac{1}{2}b_j-r'\right)=b_{i,j}\cdot b_j.$$

由上述推算结果可以看出，参数 r' 与最终的结果没有关系，所以可以随机选择.

6) $s_j=0\in\boldsymbol{S}$，$b_{i,j}'=\frac{1}{2}b_{i,j}+r$，$b_{i,j}''=\frac{1}{2}b_{i,j}-r$，$b_j'=b_j''=b_j$. 将上述参数值带入式(2)，那么有

$$r_{i,j}=b_{i,j}'\cdot b_j'+b_{i,j}''\cdot b_j''=\left(\frac{1}{2}b_{i,j}+r\right)\cdot b_j+\left(\frac{1}{2}b_{i,j}-r\right)\cdot b_j=b_{i,j}\cdot b_j.$$

由上述推算结果可以看出，参数 r' 与最终的结果没有关系，所以可以随机选择.

7) 通过 5) 6)的推导，得出

$$R_I=\{b_{i,j}\cdot b_j,b_{i,j}\in\boldsymbol{B}_i,b_j\in\boldsymbol{B}\}=\boldsymbol{B}_i\cdot\boldsymbol{B}.$$

由 7)中最终的推算结果，可知本文提出的面向多关键字的模糊密文搜索方案是正确的.

2.5 方案详细设计

本节将通过对数据拥有者、可信赖用户和服务器这 3 个实体间的交互对方案的详细设计进行描述.

1) 在初始化阶段，数据拥有者首先选定待存储的文件集；然后运行算法 $KeyGen(1^k)$ 为系统生成密钥.

2) 对于已选定的文件集，数据拥有者将采用传统的对称加密算法对其进行加密.

3) 数据拥有者为待存储的文件设置索引，即运行算法 $BuildIndex(sk,F,W)$ 得到文件的安全索引；然后将加密后的文件集与安全索引一同存储到服务器中.

4) 数据拥有者根据判断选择自己的可信赖用户.

5) 当可信赖用户想要在服务器中搜索特定文件时，他将给定关键字集合，并将其发送给数据拥有者.

6) 数据拥有者在接收到用户发送的关键字集合后，运行 $Trapdoor(sk,Q)$ 算法为关键字集合生成搜索陷门，并返回给可信赖用户.

7) 可信赖用户将搜索陷门发送给服务器进行搜索请求.

8) 服务器在接收到搜索陷门后将对搜索陷门和加密的安全索引运行 $Search(I,t)$ 算法，得到的结果为加密的目标文件集；然后将目标文件集返回给可信赖用户.

9) 可信赖用户在接收到搜索结果后，通过数据拥有者的协助将搜索结果进行解密，得到最终的明文文件.

3 方案分析

3.1 正确性分析

本文提出的面向多关键字的密文搜索方法既可对准确的关键字进行搜索，又可对模糊的关键字

进行搜索. 因此我们将从下面 2 个方面对方案的搜索结果的正确性进行分析.

1) 本文方案能够返回对于准确的关键字搜索的正确结果. 如果查询关键字 $Q\subset W_D$,云服务器应该包含结果集中的文件 D. 回想一下在构建索引和查询的过程中,我们使用了相同的 l 个 Hash 函数 $h_j\in H,1\leqslant j\leqslant l$. 那么在查询 Q 中被设为 1 的位置,同样在索引 I_D 中也设为 1. 这就表明查询能够产生内积操作的最大值. 因此,文件 D 一定包含在结果集中.

2) 本文方案能够以高的概率返回对于模糊的关键字搜索的正确结果. 假设关键字 $w\in Q$ 与关键字 $w'\in W_D$ 稍有不同,也就是说 $d(w,w')\leqslant r_1$,其中 r_1 是位置敏感 Hash 函数中的距离阈值. 如果对于所有 l 个位置敏感 Hash 函数有 $h_j(w)=h_j(w')$, $h_i\in H,1\leqslant i\leqslant l$,那么搜索结果将返回如同精确关键字搜索一样的内积操作最大值.

而如果 $d(w,w')\leqslant r_1$,但是 $h_j(w)\neq h_j(w')$,那么我们把它叫做位置敏感 Hash 失误,它降低了内积计算结果. 当 $d(w,w')\leqslant r_1$ 时,w 和 w' 之间发生了 k 次位置敏感 Hash 失误,则该概率为 $\binom{l}{k}(1-p_1)^k p_1^{l-k}$,其中 p_1 是被定义在位置敏感 Hash 函数中的概率. 而在实际中 p_1 接近于 1, $\binom{l}{k}(1-p_1)^k p_1^{l-k}$ 会随着 k 的减少而减少.

而如果 $d(w,w')>r_1$,那么位置敏感 Hash 函数将它们散列在一起的概率就会非常低. 因此,本文方案会以高的概率返回相当高的内积计算结果.

3.2 安全性证明

目前大多数的密文搜索方案或多或少会泄露用户的某些相关信息. 本文在这些泄露信息存在的条件下,给出方案的选择关键字安全性证明. 安全性证明过程中,将方案的执行过程描述成敌手与用户或者模拟器之间的交互游戏,证明对于所有敌手,都无法从搜索过程中获取除了允许泄露的信息之外的任何信息.

接下来首先分析在方案的密文生成过程、安全索引生成过程以及陷门生成过程允许向服务器泄露的信息,然后给出这些泄露函数的形式化定义.

文件集合 F 中的每个文件对应的唯一标识可记为 f_i,文件数量记为 $|F|$. 对于生成密文、生成安全索引、生成陷门 3 个操作,分别定义 3 个泄露函数来描述实验 $Real_{S_1}(sk)$ 和实验 $Ideal_{S_2}(sk)$ 在执行这些操作过程中互享的信息. 具体内容如下:

1) 生成密文

定义文件密文 F_E,密文大小可定义为 $|F_E|$,实验 $Ideal_{S_2}(sk)$ 可以获取实验 $Real_{S_1}(sk)$ 的密文 F_E、文件大小 $|F_E|$、文件数量 $|F|$ 以及文件的标识 f_i,将这些信息记为 G_1,即:

$$G_1=(|F|,f_i,F_E,|F_E|).$$

2) 生成陷门

在生成搜索陷门过程中,实验 $Ideal_{S_2}(sk)$ 可以获取实验 $Real_{S_1}(sk)$ 的查询关键字集合 Q 中每一个元素 q_i,Q 的大小记为 $|Q|$,将这些信息记为 G_2,即:

$$G_2=(Q,|Q|).$$

3) 生成安全索引

在生成陷门过程中,实验 $Ideal_{S_2}(sk)$ 可以了解在实验 $Real_{S_1}(sk)$ 中关键字集合 Q 中每一个元素 q_i 是否存在于文件 f_j 中,可以将这些信息记为 G_3.

定理 1. 如果文件加密方案、生成安全索引方案、生成陷门方案都是 CPA 安全的,则本文提出的面向多关键字的模糊密文搜索方案在随机预言机模型下是针对自适应选择关键字攻击的 (G_1,G_2,G_3) 安全的.

定理 1 表明,在给定上述泄露函数的情况下,本文提出的面向多关键字的模糊密文搜索方案在预言机模型下是安全的.

证明. 构造一个多项式模拟器,与敌手进行交互,在这个过程中,模拟器需要分别执行实验 $Real_{S_1}(sk)$ 和实验 $Ideal_{S_2}(sk)$,最后敌手无法区分收到的结果是实验 $Real_{S_1}(sk)$ 的结果还是实验 $Ideal_{S_2}(sk)$ 的结果.

1) 生成密文阶段

初始阶段:挑战者执行初始化算法 $KenGen(1^k)$,密钥 sk 作为该阶段的输出结果.

挑战阶段:敌手 A 选择想要挑战的明文信息 F,并将其发送给挑战者,挑战者分别执行实验 $Real_{S_1}(sk)$ 和实验 $Ideal_{S_2}(sk)$.

实验 $Real_{S_1}(sk)$:明文信息集 $F=\{f_1,f_2,\cdots\}$,执行加密算法 $Enc(f_i)$,其中 $1\leqslant i\leqslant|F|$,输出 $X=(Enc(f_i),1\leqslant i\leqslant|F|)$.

实验 $Ideal_{S_2}(sk)$:根据泄露函数 G_1,随机生成 $f_i\in\{0,1\}^{|f_i|}$,其中 $1\leqslant i\leqslant|F|$,输出密文的模拟值 $X'=(f_i',1\leqslant i\leqslant|F|)$.

生成随机参数 $b\in\{0,1\}$. $b=0$ 时,将 X 发送给敌手;当 $b=1$ 时,将 X' 发送给敌手.

猜测:最后,敌手输出对 b 的 1 个猜测 b''.

密文加密采用 AES 加密方案,由于该方案是 CPA 安全的,因此,保证了对于任何多项式敌手来说,都无法区分模拟值 X' 与真实值 X,即敌手获胜的概率是可忽略的.

2) 生成陷门阶段

初始阶段:挑战者执行初始化算法 $KenGen(1^k)$,密钥 $sk=(\boldsymbol{M}_1,\boldsymbol{M}_2,\boldsymbol{S})$作为该阶段的输出结果,其中 $\boldsymbol{M}_1,\boldsymbol{M}_2\in(\mathrm{R}\times\mathrm{R})^k$,$\boldsymbol{S}\in\mathrm{R}^k$.

挑战阶段:敌手 A 选择想要挑战的查询关键字 Q,并将其发送给挑战者,挑战者分别执行实验 $Real_{S_1}(sk)$和实验 $Ideal_{S_2}(sk)$.

实验 $Real_{S_1}(sk)$:挑战者将 Q 作为输入,执行算法 $BuildTrapdoor(sk,Q)$,输出真实陷门值 t.

$BuildTrapdoor(sk,Q)$算法中,记查询关键字 Q 对应的布隆过滤器为 $\boldsymbol{B}$,若 $\boldsymbol{S}$ 中对应的 $s_j=0$,则令 $b'_j=b''_j=b_j$;若对应的 $s_j=1$,则令:$b'_j=\frac{1}{2}b_j+r'$,$b''_j=\frac{1}{2}b_j-r'$. 令 $\boldsymbol{B}'=(b'_1,b'_2,\cdots,b'_k)$,$\boldsymbol{B}''=(b''_1,b''_2,\cdots,b''_k)$. 最终的真实陷门为 $t=(\boldsymbol{M}_1^{-1}\cdot\boldsymbol{B}',\boldsymbol{M}_2^{-1}\cdot\boldsymbol{B}'')$.

实验 $Ideal_{S_2}(sk)$:根据泄露函数 G_1,G_2,按下面步骤生成模拟的陷门值.

① 建立查询关键字 $Q'=(q'_i,1\leqslant q\leqslant|Q|)$,其中 $q'_i\in\{0,1\}^k$,而且 q'_i 中 1 的数量与 q_i 的数量相同,只是 1 所在的位置不同.

② 执行 $BuildTrapdoor(sk',Q')$,输出安全索引模拟值 t'.

生成随机参数 $b\in\{0,1\}$. $b=0$ 时,将 X 发送给敌手;当 $b=1$ 时,将 X'发送给敌手.

猜测:最后,敌手输出对 b 的 1 个猜测 b''.

由于加密查询关键字中参数 r 和密钥 sk 的伪随机性,保证了对于任何多项式敌手来说,都无法区分模拟值 X' 与真实值 X,即敌手获胜的概率是可忽略的.

3) 生成安全索引阶段

初始阶段:挑战者执行初始化算法 $KenGen(1^k)$,密钥 $sk=(\boldsymbol{M}_1,\boldsymbol{M}_2,\boldsymbol{S})$作为该阶段的输出结果,其中 $\boldsymbol{M}_1,\boldsymbol{M}_2\in(\mathrm{R}\times\mathrm{R})^k$,$\boldsymbol{S}\in\mathrm{R}^k$.

挑战阶段:敌手 A 选择想要挑战的对于关键字 W 的安全索引,并将其发送给挑战者,挑战者分别执行实验 $Real_{S_1}(sk)$和实验 $Ideal_{S_2}(sk)$.

实验 $Real_{S_1}(sk)$:挑战者将 W 作为输入,执行算法 $BuildIndex(sk,f_i,W)$,输出真实的安全索引 I_i.

实验 $Ideal_{S_2}(sk)$:根据泄露函数 G_1,G_2,G_3,按下面步骤生成模拟的安全索引的值.

① 初始化 $|X'|$ 个空向量 $\boldsymbol{W}_{f'_i}$.

② 任意 $q_i\in Q$,如果关键字 q_i 包含在 f_j 中,对 q'_i 使用对偶编码函数转换成向量 $\boldsymbol{v}'_i$。对于 $1\leqslant j\leqslant|X|$,计算 $\boldsymbol{W}_{f'_j}=\boldsymbol{W}_{f'_j}+\boldsymbol{v}'_i$.

③ 将 $\boldsymbol{W}_{f'_i}$ 中大于 1 的元素重新设置为 1.

④ 执行 $BuildIndex(sk,f_i,W)$,输出安全索引的模拟值 I'_i.

生成随机参数 $b\in\{0,1\}$. $b=0$ 时,将 X 发送给敌手;当 $b=1$ 时,将 X'发送给敌手.

猜测:最后,敌手输出对 b 的 1 个猜测 b''.

由于生成安全索引中参数 r'和密钥 sk 的伪随机性,保证了对于任何多项式敌手来说,都无法区分模拟值 X'与真实值 X,即敌手获胜的概率是可忽略的.

综上所述,对任意多项式敌手,实验 $Real_{S_1}(sk)$和实验 $Ideal_{S_2}(sk)$的输出是一致的,即存在可忽略概率 $negl(k)$,使得:

$$|Pr[Real_{S_1}(sk)=1]-Pr[Ideal_{S_2}(sk)=1]|\leqslant negl(k).$$

因此,本文提出的面向多关键字的模糊密文搜索方案在随机预言机模型下针对选择关键字攻击是 (G_1,G_2,G_3)安全的. 证毕.

4 测试分析

4.1 性能比较

本节我们对本文方案与文献[11]进行对比. 为了使得对比更加清晰,假设本方案中存储文件总个数为 N,使用到的 Bloom 过滤器大小为 k;文献[11]中关键字的总个数为 W,模糊关键字集合大小的最大值为 M.

表 1 给出了 2 个方案在效率方面的对比结果. 与文献[11]相比,本文方案的存储代价与文件总个数呈线性关系,而文献[11]的存储代价与关键字总个数呈线性关系. 从初始化时间复杂度和搜索时间复杂度上看,2 个方案都有较好的效率. 而对于索引生成代价,本文显然具有更少的代价.

表 2 给出了 2 个方案在性能方面的对比分析. 首先,2 个方案都能进行模糊关键字搜索,而文献[11]在此基础上还能够进行可验证搜索. 但本文方案支持多关键字搜索,而文献[11]只能进行单关键字搜索.

Table 1 Efficiency Analysis

表 1 效率分析

Efficiency	Ref[11]	This Paper
Storage Cost	$O(MW)$	$O(kN)$
Initialization Time Complexity	$O(1)$	$O(1)$
Search Time Complexity	$O(1)$	$O(1)$
Index Generation Cost	$O(M^2)$	$O(N)$

Table 2 Performance Analysis

表 2 性能分析

Performance	Ref[11]	This Paper
Fuzzy Searchability	Yes	Yes
Verifiable Searchability	Yes	No
Number of Keyword	1	N

4.2 性能测试

本节主要分析各个影响因素对系统耗时的影响;然后针对不同的影响因素,设计测试数据进行性能测试. 该系统的测试指标可分为:文件的大小、文件对应的关键字个数和文件的数量. 根据这 3 类测试指标的划分,设定 3 种测试数据. 第 1 组数据主要用于测试文件大小对该系统性能的影响,所以需要设定固定数量的文件和每个文件所拥有的关键字个数;第 2 组数据主要用于测试文件拥有的关键字个数对系统性能的影响,因而需要提前设定文件的数量以及文件的大小;第 3 组数据主要用于测试系统中文件的数量对其性能的影响,也要将文件的大小固定在一定区间内,而且每个文件所拥有的关键字个数相同.

定义完测试数据后,要在模型系统中测试每一组数据在系统中的表现. 影响系统耗时的主要操作分为:构建索引以及陷门、加密索引以及陷门、搜索. 所以进行数据测试时,需要分别记录各指标对这些操作耗时的影响. 现实应用中,模型系统中的客户端和服务器端分别在 2 台独立的计算机上运行,主机需要装载 Windows 7 旗舰版,内存一般为 4 GB. 性能测试实验中,仅用 1 台计算机模拟该系统. 记录的结果都是每组数据 10 次运行后的结果的平均值.

实验 1. 测试文件的大小对系统耗时的影响.

按文件大小所在区间分组,将测试数据分为 4 类,分别记为 A 类文件、B 类文件、C 类文件、D 类文件,A 类文件中文件大小为 1～50 KB,B 类文件中文件大小为 100～300 KB,C 类文件中文件大小为 400～500 KB,D 类文件中文件的大小为 1 000～2 000 KB 的值. 这 4 类文件中都包含 500 个文件,且每个文件都包含 4 个关键字. 该模块测试结果如图 7 所示:

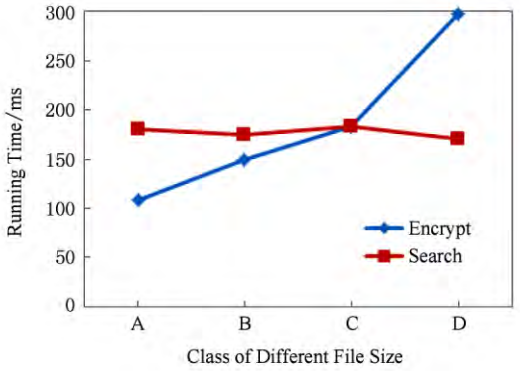

Fig. 7 Influence of file size

图 7 文件大小的影响

从图 7 可以看出,文件体积越大系统耗时越多. 文件的大小对系统搜索过程几乎没有什么影响.

实验 2. 测试文件的数量对系统耗时的影响.

按照文件的数量将实验的数据分为 5 组,第 1 组含有 500 个文件,第 2 组有 1 000 个文件,第 3 个有 1 500 个文件,第 4 组有 2 000 个文件,第 5 组有 2 500 个文件. 每组文件集合中各文件的大小均在 50～100 KB,而且每个文件所对应的关键字个数都为 4. 执行文件的上传操作,整理并分析每组操作的耗时,如图 8 所示:

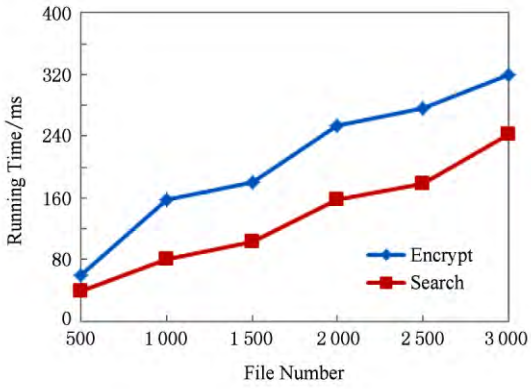

Fig. 8 Influence of file number

图 8 文件数量的影响

文件数量对加密过程和搜索过程耗时的影响如图 8 所示,随着文件数量的增多,系统在生成文件密文阶段所耗费的时间就越多,同样系统在搜索过程耗时也就越多.

实验 3. 测试单个文件所对应的关键字个数对系统的影响.

按每个文件包含的关键字个数将实验数据分为4组,第1组数据中单个文件对应的关键字数量为5,第2组数据中单个文件对应的关键字数量为10,第3组数据中单个文件对应的关键字数量为15,第4组中单个文件对应的关键字数量为20.每组数据中一共包括100个文件,其中每个文件的大小都在50~100 KB之间.该实验结果分析如图9所示:

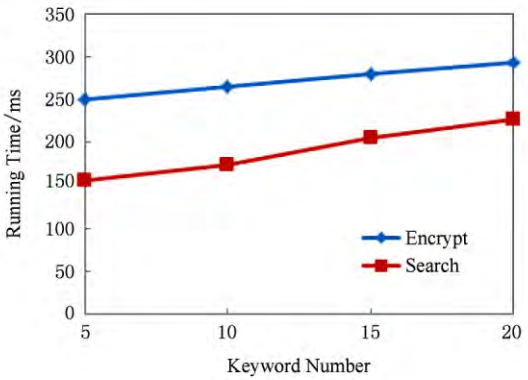

Fig. 9 Influence of keyword number

图9 关键字个数的影响

从图9可以看出,随着每个文件对应的关键字数量的增多,系统在生成安全索引阶段和搜索阶段的耗时就会增多.但通过对图中实验结果的详细分析可知,当关键字个数由10个变为15个时,系统对于搜索阶段的耗时仅由175 ms增长到205 ms.所以系统搜索耗时的增长幅度相对于关键字的增长幅度来说是比较小的.因而我们说文件对应的关键字个数对系统搜索阶段耗费的时间影响是可以忽略的.

5 总 结

本文深入系统地研究了国内外的可搜索加密机制,总结各方案的优缺点,并进一步研究了布隆过滤器以及位置敏感Hash函数等相关理论,提出面向多关键字的模糊密文搜索方案.

面向多关键字的模糊密文搜索方案使用布隆过滤器和位置敏感Hash函数技术,能够有效实现密文的多关键字的模糊搜索.利用对偶编码函数将关键字转换成向量,实现关键字的内积运算.在安全性方面,通过攻击性游戏,模拟用户和服务器之间的交互,并利用参数的随机性,证明该方法是选择关键字攻击安全的.因此,本方案具有很强的理论与实践意义.

参 考 文 献

[1] Song D, Wagner D, Perrig A. Practical techniques for searches on encrypted data [C] //Proc 2000 IEEE Symp on Security and Privacy. Piscataway, NJ: IEEE, 2000: 44-55

[2] Chang Y C, Mitzenmacher M. Privacy preserving keyword searches on remote encrypted data [C] //Proc of Applied Cryptography and Network Security. Berlin: Springer, 2005: 442-455

[3] Dong C, Russello G, Dulay N. Shared and Searchable Encrypted Data for Untrusted Servers [M]. Berlin: Springer, 2008: 127-143

[4] Revathy B, Anbumani A, Ravishankar M. Enabling secure and efficient keyword ranked search over encrypted data in the cloud [J]. International Journal of Recent Advances in Science & Engineering, 2015, 1(1): 28-32

[5] Cao N, Wang C, Li M, et al. Privacy-preserving multi-keyword ranked search over encrypted cloud data [J]. IEEE Trans on Parallel and Distributed Systems, 2014, 25(1): 222-233

[6] Xia Zhihua, Wang Xinhui, Sun Xingming, et al. A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data [J]. IEEE Trans on Parallel and Distributed Systems, 2016, 27(2): 340-352

[7] Bringer J, Chabanne H, Kindarji B. Error-tolerant searchable encryption [C] //Proc of the IEEE Int Conf on Communications. Piscataway, NJ: IEEE, 2009: 1-6

[8] Van Liesdonk P, Sedghi S, Doumen J, et al. Computationally efficient searchable symmetric encryption [C] //Proc of Secure Data Management. Berlin: Springer, 2010: 87-100

[9] Li J, Wang Q, Wang C, et al. Fuzzy keyword search over encrypted data in cloud computing [C] //Proc of the IEEE on INFOCOM. Piscataway, NJ: IEEE, 2010: 1-5

[10] Zhou Wei, Liu Lixi, Jing He, et al. *K*-gram based fuzzy keyword search over encrypted cloud computing [J]. Journal of Software Engineering and Applications, 2013, 6: 29-32

[11] Wang Jianfeng, Ma Hua, Tang Qiang, et al. Efficient verifiable fuzzy keyword search over encrypted data in cloud computing [J]. Computer Science and Information System, 2013, 10(2): 667-684

[12] Indyk P, Motwani R. Approximate nearest neighbors: Towards removing the curse of dimensionality [C] //Proc of the 13th Annual ACM Symp on Theory of Computing. New York: ACM, 1998: 604-613

[13] Datar M, Immorlica N, Indyk P, et al. Locality-sensitive Hashing scheme based on *p*-stable distributions [C] //Proc of the 12th Annual Symp on Computational Geometry. New York: ACM, 2004: 253-262

[14] Bloom B H. Space/time trade-offs in Hash coding with allowable errors [J]. Communications of the ACM, 1970, 13(7): 422-426

[15] Zhang Zhen, Dai Guanzhong, Liu Hang, et al. Safe and resumable secret key management mechanism based on database encryption [J]. Computer Measurement and Control, 2008, 16(10): 1469-1471 (in Chinese)
(张贞, 戴冠中, 刘航, 等. 一种基于数据库加密的安全可恢复密钥管理机制[J]. 计算机测量与控制, 2008, 16(10): 1469-1471)

[16] Zhang Chuanrong, Yin Zhonghai, Xiao Guozhen. Authenticated encryption schemes without using Hash and redundancy functions [J]. Acta Electronica Sinica, 2006, 34(5): 874-877(in Chinese)
(张串绒, 尹忠海, 肖国镇. 不使用 Hash 和 Redundancy 函数的认证加密方案[J]. 电子学报, 2006, 34(5): 874-877)

Wang Kaixuan, born in 1992. Master in Software College, Northeastern University. Her main research interest is searchable encryption.

Li Yuxi, born in 1990. PhD candidate. Her main research interest is secure cloud storage (eliyuxi@gmail.com).

Zhou Fucai, born in 1964. PhD, professor and PhD supervisor. Senior member of CCF. His main research interests include cryptography and network security, trusted computing, and critical technology in electronic commerence.

Wang Quanqi, born in 1992. Master. His main research interest is mobile computing (wqq_7622@126.com).

2017 年《计算机研究与发展》专题(正刊)征文通知
——车联网关键技术与应用研究

随着传感器和无线通信等技术的发展,车联网作为物联网和移动互联网发展的代表性产物,成为现代智能交通的重要组成部分. 车联网在提高交通运行效率和减少环境污染的同时,还能够提供辅助安全驾驶和安全消息广播等措施来保障生命财产安全,此外车载导航、娱乐和 Internet 接入等服务改善了驾驶体验. 尽管我国的车联网研究起步稍晚,但近几年得到了政府、企业和研究机构的大力支持和高度重视. 2013 年,工信部电信研究院和中国移动研究院联合发布《车联网产业发展白皮书》; 2014 年,国务院出台《关于促进智慧城市健康发展的指导意见》,将智慧交通上升到国家战略; 2016 年,工信部批准国家智能网联汽车(上海)试点示范区封闭测试区在上海国际汽车城正式开园,同时国家重大专项 LTE-V 车联网专用通信标准化项目启动. 在学术界,车联网已经成为一个备受关注的新兴研究领域,对其关键技术的研究有利于推进智慧交通的建设步伐.

《计算机研究与发展》拟于 2017 年 11 月出版"车联网关键技术与应用研究"专题,讨论车联网领域最新的突破性进展,交流车联网领域新的学术思想和方法,展望车联网前沿技术未来的发展趋势. 欢迎相关领域的专家学者和科研人员踊跃投稿. 现将专题论文征集有关事项通知如下.

征文内容

本专题包括(但不限于)下列主题:

- 车联网移动通信与接入技术
- 车联网移动模型与性能评估
- 车联网智能交通控制
- 车联网资源调度与路径规划
- 车联网消息安全与隐私保护
- 车联网紧急消息广播
- 车联网协同控制(例如协助下载、协作式安全应用)
- 车联网各层协议(例如 MAC 层协议、路由算法等)

投稿要求

1) 论文应属于作者的科研成果,数据真实可靠,具有重要的学术价值与推广应用价值,未在国内外公开发行的刊物或会议上发表或宣读过,不存在一稿多投问题. 作者在投稿时,需向编辑部提交版权转让协议.
2) 论文应包括题目、作者信息、摘要、关键词、正文和参考文献,论文一律用 Word 排版,论文格式请参考《计算机研究与发展》近期文章.
3) 论文需附通讯作者的联系方式、联系地址、及 E-mail 信息.
4) 论文请通过期刊网站(http://crad.ict.ac.cn)进行投稿,并在作者留言中注明"车联网 2017 专题"(否则按自由来稿处理).

重要日期

征文截稿日期: 2017 年 5 月 31 日　　录用通知日期: 2017 年 7 月 18 日
最终稿提交日期: 2017 年 7 月 25 日　　出版日期: 2017 年 11 月

特邀编委

吴黎兵　教授　武汉大学　wu@whu.edu.cn　　郭得科　教授　国防科学技术大学　dekeguo@nudt.edu.cn
蒋洪波　教授　华中科技大学　hongbojiang@hust.edu.cn

联系方式

编 辑 部: crad@ict.ac.cn, 010-62620696, 010-62600350
通信地址: 北京 2704 信箱《计算机研究与发展》编辑部　　邮政编码: 100190