文章编号: 1001-0920(2019)01-0097-08 DOI: 10.13195/j.kzyjc.2017.0802

一种基于BloomFilter的改进型加密文本模糊搜索机制研究

吴 曦, 俞能海†, 张卫明

(1. 中国科学院 电磁空间信息重点实验室，合肥 230026；2. 中国科学技术大学 信息科学技术学院，合肥 230026)

摘 要: 随着云计算的日益普及,为实现共享计算资源、节约经济成本等目的,越来越多的重要数据被从本地外包迁移至云端. 出于对保护云端数据安全和用户隐私等方面的考虑,数据所用者一般倾向对敏感数据进行加密处理,在此基础上,如何能够对数据开展有效检索处理成为关注的重点. 为此,提出一种改进的密文数据多关键字检索机制,一方面,基于BloomFilter数据结构设计一种新的关键字转换方法,能够在保持模糊搜索功能及识别率的同时,有效降低数据索引规模;另一方面,基于动态混淆参数调节的思路改进相似度评估算法,以提高数据的加密强度,并且能很好地反映用户的检索偏好. 实验结果验证了所提机制是可行和高效的.

关键词: 云计算；隐私保护；可搜索加密；模糊检索；BloomFilter

中图分类号: TP37 **文献标志码:** A

An improved multi-keyword fuzzy search scheme based on BloomFilter over encrypted text

WU Xi, YU Neng-hai†, ZHANG Wei-ming

(1. Cyberspace Information Lab，Chinese Academy of Science，Hefei 230026，China；2. School of Information Science and Technology，University of Science and Technology of China，Hefei 230026，China)

Abstract: With the popularity of the cloud computing, more and more data owners are motivated to outsource their complex data from local sites to the cloud for great shared comuputing sources and economic savings. But for protecting data security and privacy, sensitive data have to be encrypted before outsourcing, which obsoletes some traditional data utilization, for example the multi-keywords search. In this paper, we develop a enhanced multi-keywords fuzzy search scheme. On one hand, a novel method is designed to transfor keywords based on BloomFilter, which can reduce the index length effectively while keeping the fuzzy search rate. On the other hand, the similarity measure method is improved based on dynamic adaption of confusion parameters to achieve various stringent privacy requirements, which shows the data user's favoritism better. Experiments on real-world data show that the proposed scheme is feasible, effective and accurate.

Keywords: cloud computing；privacy-preserving；searchable encryption；fuzzy search；BloomFilter

0 引 言

随着云计算技术的日益成熟,其架构与产品已惠及越来越多的用户,特别是云存储服务,成为大数据时代最为普及的云计算应用之一. 对于拥有和管理着海量数据的终端用户,可以将数据外包至云服务器上,以大幅减轻本地的计算和存储压力,降低因各类事故或人为因素造成的数据损坏与丢失风险. 与此同时,随着云存储的广泛应用,如何保护云端数据的安全和内容隐私成为最值得关注的问题之一. 特别是对于机密性或私密性较高的诸如健康、经济、军事等相关领域的数据,数据拥有者会倾向于将其进行加密之后再上传至云服务器,有许多传统的数据加密方法可以用以实现这个目的. 然而,在云计算环境和数据加密上传的背景下,产生了一些新的问题与挑战,主要体现在两个方面:一是数据通信量的大幅增加给云环境网络带宽及能耗带来的压力,这一问题在

收稿日期: 2017-06-22；**修回日期:** 2017-12-23.

责任编委: 刘德荣.

作者简介: 吴曦 (1980–), 男, 博士生, 从事信息安全及其应用的研究；俞能海 (1964–), 男, 教授, 博士, 从事网络安全、多媒体内容安全等研究.

†通讯作者. E-mail: ynh@ustc.edu.cn.

混合云模式下移动终端大量应用普及的情况下尤为突出,移动设备的计算能力和电池容量都相对有限,大数据量的加密信息传输和加解密运算会严重降低设备应用效率;二是传统加密方法对数据应用带来了挑战,即由于数据的加密,使得云服务器端除了简单的数据存储之外,无法执行更多的信息处理与运算功能,从而影响了云计算根本优势的发挥,这也是本文关注的重点.

总体而言,云计算环境下数据的隐私保护和安全保密要求与云计算的有效应用形成了矛盾. 聚焦应对上述两类问题,从实际应用最为广泛的数据搜索操作入手,面向能够在数据加密的前提下,由云服务器执行如模糊检索等功能相对丰富、能够反映用户查询偏好的数据检索需求开展研究. 近年来,研究者们针对加密数据搜索提出了一些良好的思路和方法[1-7]. 从存在不足的角度而言,这些机制有的只能支持单关键字搜索,有的执行效率较低,并且大多不支持模糊搜索,很大程度地制约了算法的实用性. Cao等[8]提出了基于Bloom Filter构建索引,运用局部敏感哈希技术(LSH)实现支持模糊查询的机制;Fu等[9]为了提升检索效率和精度,提出了一种改进的数据结构,用以表达数据的索引. 以上机制的不足在于数据索引生成时初始化数据结构的规模仍然较大,影响了实用效果,且检索算法无法有效反映用户的检索偏好,本文重点针对这些开展研究.

目前的主要相关研究可分为不支持模糊查询的密文检索机制和支持模糊查询的密文搜索机制两类. 其中不支持模糊查询的密文检索机制主要包括以下两小类: 1) 单关键字搜索: 文献[3]提出了一种基于对称密码学算法的可搜索加密方案,其核心思想是对密文文件进行按序扫描,计算是否存在与给定的关键字的加密形式相匹配的内容,该算法的问题在于检索效率不高. 针对此问题,文献[4]提出了基于安全索引的快速可搜索加密方案,索引结构是基于Bloom Filter来实现的;文献[5]研究提出了基于公钥密码的可搜索加密方案PEKS,提升了密钥传递的安全性,但仍然面临着计算开销很大的难题;文献[6]基于身份加密(Identity based encryption)和对称密码学分别提出了两种可搜索加密的方案,其应用场景为在加密后的审计日志上进行关键字搜索,该方案的特点为需要服务器端对文件进行加密,而且用户在进行检索时需要访问服务器并自己承担搜索开销. 为了进一步减少用户在检索时的计算量并提高用户的检索体验,文献[7]运用保序加密(Order preserving encryption)来实现可支持结果排序的密文检索,云服务器可在不掌握明文信息的情况下对密文索引进行快速排序并按关联性返回结果,保序加密泄露了一定程度的密文信息,其安全性有待进一步加强. 2) 多关键字搜索: 为满足实际应用中常见的多关键字搜索需求,相关研究得到了广泛的关注. 文献[10]基于对称密码学和公钥密码学分别提出了支持连接关键字搜索的可搜索加密机制;文献[11]提出了多接受者公钥加密方法(Multi-receiver public key encryption);文献[12]在先前的单关键字可排序搜索的工作上进一步拓展,采用kNN的思想实现了多关键字可排序搜索;文献[13-14]改进地提出了基于相似性排序的可验证的隐私保护多关键字的文本搜索;文献[15]提出了一个安全的多关键字排序的可搜索加密方案,支持索引动态更新.

支持模糊查询的密文搜索机制主要针对用户可能的错误输入以及其他情况导致的无法精确进行关键字搜索的场景. 文献[13]研究了密文域上的模糊关键字搜索问题,采用编辑距离的概念来界定哪些词属于模糊检索范围内的词,即在某个编辑距离之内的词都视作该关键字的变形,在上传和检索时所有该关键字的变形都会被一并操作,这样的方案一方面实现了其所提出的模糊检索,另一方面却线性增长了存储开销和用户方面因为需要对不精确的结果进行进一步筛选而带来的网络和计算开销. 此后,文献[9]提出了使用局部敏感哈希来实现多关键字的模糊搜索,此方案是直接的模糊匹配,不需要将关键字的所有可能拼写列举出来,不会增加索引的大小,但检索效率受到一定影响.

1 问题建模

1.1 系统模型

作为研究基础,定义一个云存储和数据搜索系统模型,其中包含3类角色: 数据所有者、数据查询者和云环境. 云环境由多个私有和公有云构成,如图1所示. 在该系统模型中,数据所有者拥有数据文件集,出于对本地存储容量不足、计算能力有限以及数据集中管理规定等多方面的考虑,将该数据集上传至云环境中;出于对安全保密的考虑,数据集要以加密的形式上传,同时,这些上传的加密数据可能会根据管理和使用的具体要求存储在云环境中不同的私有或公有云服务器上. 经授权的数据查询者可以向云环境发起数据搜索请求,获取云环境反馈的相关数据文件. 假定云服务器是“诚实且好奇”的[16],即云服务器在准确完成用户请求的操作同时,可能会试图从用户

的搜索请求及返回结果等数据中分析获取其他敏感的信息.

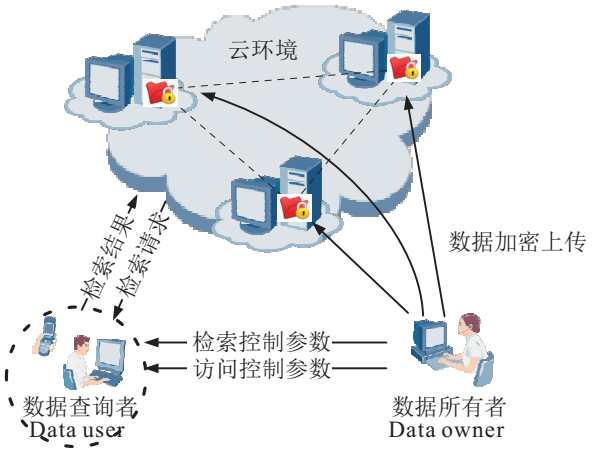

图 1　系统模型

1.2　应用模型

在上述系统模型之下,围绕主要的应用需求,应用模型可以作以下两类描述.

1.2.1　安全搜索模型

每个云服务器可以获得加密的数据文件和索引,记录用户的历史请求陷门,并统计相关密文频率信息,云服务器可以据此利用比值分析(Scale analysis)等方法推测搜索关键字等信息,因此安全的搜索模型需要有效地抵御这一风险. 在此基础上,搜索模型应尽可能提供更为友好的功能,比如模糊查询、反映用户偏好等.

1.2.2　效率提升模型

在云环境中,降低云服务器与终端的通信量、提升云服务器计算能力使用率是很重要的挑战. 按照传统的方式,在搜索过程中,终端需要对从云服务器返回的数据索引进行解密,计算相关度并排序,再将排序结果反馈给云服务器提取相关数据文件. 在这样的搜索过程中,云服务器与用户交互次数过多,必然会使带宽压力陡增,同时也影响了云服务器计算优势的发挥. 效率提升模型应充分体现云计算高效率低能耗的设计理念,将更多的计算压力释放到云服务器端[17],同时减少应用通信的数据量,降低终端与服务器的通信传输压力.

1.3　设计目标

本研究的主要目标是:在已有多关键字密文搜索算法[9,12,18]的研究基础上,聚焦保护数据安全、拓展搜索功能、反映查询者偏好等方面,设计实现一个有效的密文搜索机制,解决上述模型中面临的主要问题,并对其中的密文搜索核心算法开展有针对性的设计和改进.

2　核心算法设计

2.1　改进的密文搜索算法

2.1.1　预备知识

1) 布隆过滤器(Bloom Filter). Bloom Filter是一种具有很高空间效率的数据结构,它利用队列来表示一个集合,可以确定某个元素是否属于该集合. 初始化的时候,将队列中所有的位置都置为0,对于一个给定的集合 $S = (a_1, a_2, \cdots, a_n)$,使用 l 个独立的哈希函数,形如 $H = \{h_i | h_i : S \to m, 1 \leqslant i \leqslant l\}$,通过将队列相应的位置置为1,将属于 S 的元素 a 逐一插入到Bloom Filter中. 当要检查某个元素 q 是否属于 S 时,通过哈希函数获取 q 在Bloom Filter队列中对应的 l 个位置,如果有任何一个位置为0,则 $q \notin S$;对应的 l 个位置均为1,则要么 $q \in S$,要么出现了“假阳性”,实际上 $q \notin S$. 假阳性概率为 $(1 - \mathrm{e}^{-\frac{ln}{m}})^l$,当 $l = \dfrac{m}{n} \cdot \ln 2$ 时,最理想的假阳性概率为 $(1/2)^l$.

2) 局部敏感哈希(Locality-sensitive hashing). LSH是一个用于解决在高维空间中近似距离搜索问题的算法,该算法能够将相似的输入参数以较大的概率映射到同一区间中. 对于一个局部敏感哈希函数 H,其是 (r_1, r_2, p_1, p_2) 敏感的,则对于任意两个输入参数 x 和 y,满足

$$\begin{cases} \Pr[h(x) = h(y)] \geqslant p_1,\ d(x, y) \leqslant r_1; \\ \Pr[h(x) = h(y)] \leqslant p_1,\ d(x, y) \geqslant r_2. \end{cases}$$

其中 $d(x, y)$ 为参数 x 与 y 之间的距离.

2.1.2　主要技术

1) 关键字集合压缩. 目标数据集 $F = (F_1, F_2, \cdots, F_n)$ 中往往包含大量文件,其中有海量的关键字 $W = (w_1, w_2, \cdots, w_m)$. 为了提高索引生成效率(特别是在云环境中,对网络带宽和数据传输量要求较高的情况下),尽可能地控制索引规模. 采用文献[9]的方法,首先进行数据预处理,运用传统的Porter分词算法[20]对从数据集中提取出的关键字集合进行“词根”过滤,例如“walks”、“walking”、“walked”等近似的关键字,词根均为“walk”. 后续的检索操作基于词根进行,大幅减少了计算量,且由于该操作只是在索引生成时执行一次,对整个系统的运行效率不会产生过大的影响.

2) 关键字转换与索引生成. 关键字转换是本研究所提机制中最关键的步骤. 与文献[9,17]中类似,利用映射向量来表示关键字. 在文献[9]中,使用了 26×26 比特的大向量来对关键字进行表示. 在文献[17]中,将相关向量的长度降为130比特,但在数据集合检索量较大的情况下,这样的索引规模仍将对算法

执行效率产生负面影响,存在进一步优化的空间. 因此,本文针对进一步缩小映射向量长度开展研究. 首先通过统计研究,构建一个普适的关键字匹配特征体系,这些特征能有效反映索引关键字与检索关键字之间的近似关系;然后依据这个特征体系,将关键字转换到映射向量中去;进一步,为了更好地反映数据查询者的偏好,向索引及查询的关键字映射向量中分别嵌入相关统计信息,进而基于这样的关键字映射向量进行可搜索加密.

i) 特征体系. 注意到,作为数据查询者,输入检索关键字时可能会出错,但在大多数情况下不会错得很“离谱”,也就是说,即使在错误拼写的情况下,准确关键字与有误关键字之间仍存在一定的关联关系,通过突出体现这种关联关系的特征,能够有效地表达准确关键字与有误关键字的相似性. 经过统计分析,提出两项最有代表性的特征,一是准确关键字与有误关键字首字母一致且长度相同;二是准确关键字与有误关键字包含某个字母的位置和数量相同. 因此,将上述两点作为设计判断关键字近似特征体系的基本依据. 需要强调的是,特征体系各成员之间是相互关联的有机整体,通过动态调节不同拼写出错情况下的特征权值,应尽可能降低对某些特征成员的依赖程度. 接下来的问题在于,如何在关键字的映射向量中以尽量短的长度有效表达这些特征,以及特征之间的权值如何分配.

ii) 基本算法. 在所提的关键字转换算法中,映射向量的总长为87个比特,将每个关键字分割为4个部分表示,如图2所示. 其中:首字母标识字段由5个比特构成,用于通过二进制数表示关键字的首字母是哪一个;关键字长标识字段由4个比特构成,用于通过二进制数表示关键字总长度;包含字母标识字段由26个比特构成,用于表示关键字中包含哪些字母;包含字母数量标识字段由52个比特构成,用于表示关键字所包含的字母数量分别是几个.

图 2 映射向量构成

对于关键字“hello”,可以转换为$\{'8';\ '5';\ 'e',\ 'h',\ 'l',\ 'o';\ 1,1,2,1\}$,进而通过算法向87比特映射向量中的相应位置赋值. 关键字转换成映射向量的细节由以下伪代码描述.

input: A plain-text keyword set W; A null vector $\{0,1\}^{87}$

output: A mapping-vector V

for $W_i \leftarrow W_1$ to W_0 do

Stem W_i to ST_i whose length is st_i, Translate ST_i to $\mathrm{ST}_i[j]$;

and generate a vector$\{y|y|[j]=1, 0<j<\mathrm{st}_i$;

and generate a vector $\{\mathrm{STY}_i|\mathrm{STY}[j]=0, 0\leqslant k\leqslant 86\}$;

for $\mathrm{ST}_i[j] \leftarrow \mathrm{ST}_i$ do

set $\mathrm{ST}_i[1]$ to a new element $\mathrm{STY}_i[]$;

set st_i to a new element $\mathrm{STY}_i[]$;

for $k=1$ to $k=j-1$ do

if $\mathrm{ST}_i[j]=\mathrm{ST}_i[k]$ then

$y[j]++$;

end

end

for $\mathrm{STY}_i[0]$ to $\mathrm{STY}_i[86]$ do

Set all corresponding position in $\{0,1\}^{87}$ to 1;

Set the rest position to 0;

Output the vector V_{W_i} for the keyword W_i;

end

Output the $V=\{V_{W_i}|W_i\in W\}$;

end

end

iii) 索引生成. 通过上述算法的表示,一个有拼写错误的关键字可以大概率地与准确的关键字保持近似性,由于用向量来表示关键字,这种近似性可以通过欧氏距离进行测量. 接下来,将关键字映射向量作为输入,利用布隆过滤器生成关键字索引. 通过选用合适的局部敏感哈希函数,两个相近的输入能够被大概率地映射到布隆过滤器的相同索引输出向量,最终使得关键字模糊搜索得以实现. 为了更好地反映用户偏好,检索到更有用的文档,本文提出在布隆过滤器的输出向量中嵌入统计信息,对索引向量嵌入关键字的词频(TF)均值$\overline{\mathrm{TF}}$,对查询向量嵌入关键字的文档频率倒数(IDF)均值$\overline{\mathrm{IDF}}$. 这样,当查询向量中包含文档频率较低即重要性更强的关键字时,对应的$\overline{\mathrm{TF}}$和$\overline{\mathrm{IDF}}$数值将会增大,并在密文搜索过程中通过$\overline{\mathrm{TF}}\times\overline{\mathrm{IDF}}$体现在最终的查询相似度数值中. 至此,完成关键字转换和索引生成的全部过程.

3) 基于改进的安全内积相似度评估的搜索. 在关键字转换为映射向量,并通过局部敏感哈希函数插入布隆过滤器的基础上,为保护数据在云服务器端的安全和隐私,需要对索引和搜索关键字映射向量进行加密,同时在云服务器端不解密的前提下进行搜

索. 采用安全内积相似度计算作为基本思路,主要步骤如下.

i) 密钥生成. 密钥SK由数据所有者生成,可以表示为三元组$\{S, M_1, M_2\}$,其中包括一个$|T|$bit的随机生成向量$S \in \{0,1\}^m$和两个$|T| \times |T|$的可逆随机矩阵$\{M_1, M_2\}$,$|T|$为存储索引的布隆过滤器长度. 如果数据查询者即为数据所有者,则密钥在本地生成并处理;如果数据查询者为云环境中其他用户,则需采用离线拷贝、广播加密或其他方式进行安全传递,还需考虑管理与传输的效率问题,关于这方面内容非本论文重点,可另行研究.

ii) 关键字向量加密. 利用随机向量作为指示器,将每个索引关键字向量I按以下规则随机分割成两个向量$\{I', I''\}$:如果$S_i \in S$为1,令$I'_i = \frac{1}{2}I, I''_i = \frac{1}{2}I$,否则令$I'_i = I''_i$. 利用$\{M_1, M_2\}$分别对这两个随机向量进行乘运算,生成加密的索引向量$\overline{I}: \{M_1^{\mathrm{T}} I', M_2^T I''\}$,利用同样的方法对整个文档集$F$生成加密索引集$\overline{I}_F$. 类似地,对每个查询关键字向量$Q$进行加密,生成加密查询向量$\overline{Q}: \{M_1^{-1}Q', M_2^{-1}Q''\}$.

iii) 相似度评估. 利用索引关键字与查询关键字向量的内积,即计算两向量之间的余弦夹角,能够实现在不对关键字解密的前提下,对两者的相似度进行评估,有

$$\begin{aligned}&\mathrm{Sim\_Score}(D_I, Q) =\\ &\mathrm{Sim\_Score}(\overline{D_I}, \overline{Q}) =\\ &\overline{I} \cdot \overline{Q} =\\ &M_1^T I' \cdot M_1^{-1} Q' + M_2^{\mathrm{T}} I'' \cdot M_2^{-1} Q'' =\\ &I \cdot Q = \cos(I, Q).\end{aligned}$$

从安全性角度分析,对于索引和查询关键字向量,只要保证密钥SK三元组的机密性,索引I_F或查询向量Q就难以通过计算被逆推出来. 但是,上述算法的问题在于,同样的检索关键字得到的相似度数值是相同的,如果云服务器能够跟踪访问节点和记录相似度结果,则可以发起已知密文攻击,根据相同的相似度数值统计,分析某个关键字的相似度分布,并可能关联推测和识别出该关键字. 因此,基本思路存在着较大的安全隐患,这在对机密性要求较高的云环境中是不可接受的. 本文提出根据每次检索的相对随机性动态变化地调节原始密钥和插入哑元关键字数的思路,通过引入与每个独立检索行为相关的更为随机的混淆参数,对数据加密和安全检索算法加以改进.

iv) 算法改进. 两个混淆参数定义如下:检索目标云服务器的数据集合保密等级$M_j = \{m_j\}, m_j = 1, 2, \cdots, 5$;检索关键字数目$C_s = \{c_s\}, c_s = 1, 2, \cdots$.

a) 密钥生成与加密阶段的改进. 对于索引向量I,将密钥三元组$\{S, M_1, M_2\}$扩展为四元组$\{S, M_j, M_1, M_2\}$,其中每个元素$I_i \in I$,按以下规则分割成$\{I'_i, I''_i\}$:如果$S_i \in S$为1,则令$I'_i = \frac{1}{M_j}I + r, I''_i = \frac{M_j - 1 M_j}{I} + r$,否则令$I'_i = I''_i$. 进而通过随机可逆矩阵$\{M_1, M_2\}$将索引向量$I$加密为$\overline{I}: \{M_1^{\mathrm{T}} I', M_2^{\mathrm{T}} I''\}$,实现对索引向量更为随机的分割与加密.

对于查询向量Q,将密钥三元组$\{S, M_1, M_2\}$扩展为五元组$\{S, M_j, C_s, M_1, M_2\}$,首先计算$M_j$与$C_s$的平均值$\mu = \frac{1}{2}(M_j + C_s)$,进而计算均方差$\sigma = \sqrt{\frac{1}{2}[(M_j - \mu)^2 + (C_s - \mu)^2]}$,查询向量$Q$中的每个元素$Q_i \in Q$,按以下规则分割成$\{Q'_i, Q''_i\}$:如果$S_i \in S$为0,则令$I'_i = \frac{1}{\sigma}I + r, I''_i = \frac{\sigma - 1}{\sigma}I + r$,否则令$Q'_i = Q''_i$. 进而,通过随机可逆矩阵$\{M_1, M_2\}$将查询向量$Q$加密为$\overline{Q}: \{M_1^{-1}Q', M_2^{-1}Q''\}$.

b) 相似度计算阶段的改进. Cheng等[2]介绍了一种安全的kNN算法,通过引入新的随机数,将相似度计算改进为$r\overline{I} \cdot \overline{Q}$,提高了算法的加密强度,但仍存在遭受比值攻击的隐患;Cao等[8]修改了这种安全kNN技术,对加密三元组向量及矩阵进行维度扩展,向每个查询向量的扩展维中引入新的“哑元”随机数,将相似度计算改进为$r\overline{I} \cdot \overline{Q} + \sum \varepsilon_i^v$,进一步增强了安全强度,但由于每次检索过程中选取的随机变量ε_i数量是固定的,使得$\sum \varepsilon_i^v$的分布仍存在规律性,在面临大数据分析时仍存在较大安全隐患. 为了进一步抵消这种相对固定属性,提出利用前述引入的“混淆参数”对插入哑元关键字数进行扰动的改进思路,即根据每次查询的目标文件集合保密等级M_j和查询关键字数目C_s,对加密三元组向量及矩阵进行动态维度扩展,即每次相似度计算插入的关键字数量为$M_j \& C_s = (M_j + C_s)/2$,从而将相似度计算改进为

$$\begin{aligned}&\mathrm{Sim} - \mathrm{Score}(D_I, Q) = \overline{I} \cdot \overline{Q_\varepsilon} =\\ &M_1^{\mathrm{T}} I' \cdot M_1^{-1} Q' + M_2^{\mathrm{T}} I'' \cdot M_2^{-1} Q'' + \sum_{i=1}^{M_j \& C_s} \varepsilon_i^v =\\ &r\overline{I} \cdot \overline{Q} + \sum_{i=1}^{M_j \& C_s} \varepsilon_i^v.\end{aligned}$$

2.2 与现有算法比较

就关键字的表达和索引规模而言,关键字转换是对其产生影响的最关键环节. 在现有的主要研究

中,一种思路[16]是先将关键字转换到一个相邻字母集合,然后再将该集合映射到一个 26 × 26 比特的 bi-gram 向量中,集合中包含的元素在该向量中对应位置为 1,步骤较为繁琐且索引规模很大;另一种思路[9]在此基础上进行了改进,使用一个 130 比特的 uni-gram 向量进行关键字转换. 本文从关键字特征体系的角度设计提出的关键字映射向量长度仅为 87 比特,与现有技术相比,能够使生成索引的初始向量总长度下降 50 %∼70 %. 就针对不同拼写错误的表达而言,可以从以下几类示例来分析.

1) 有一个字母错误拼写: 比如,关键字“work”$\{'23';'4';'w','o','r','k';1,1,1,1\}$被错误拼写为“wark”$\{'23';'4';'w','a','r','k';1,1,1,1\}$,首字母和长度未变,而该特征权值占比较大,因此正确与错误的关键字的欧式距离小于 $\sqrt{2}$,优于现有技术中的数值.

2) 两个字母拼写颠倒: 仍以“work”为例,如除首字母外的任意两字母拼写颠倒,映射向量的表达完全一致,因此正确与错误的关键字的欧式距离为 0.

3) 少写或多写一个字母: 比如,关键字“work”$\{'23';'4';'w','o','r','k';1,1,1,1\}$被错误拼写成“wok”$\{'23';'3';'w','o','k';1,1,1\}$,则通过权重调节,欧氏距离小于 3,接近现有技术中的数值;如被错误拼写成“worrk”$\{'23';'5';'w','o','r','r','k';1,2,1,1\}$,则通过权重调节,欧氏距离小于 $\sqrt{2}$,优于现有技术中的数值. 对比结果也体现了前述通过权重调节平衡关键字相似度对特征体系各成员依赖程度的观点.

就检索机制的安全性而言,在加密阶段,通过密钥元组的扩展和算法改进,使得对索引和查询向量的分割更加随机,实现了更为安全的数据加密. 在相似度计算与评估阶段,本研究与现有机制的思路都是通过插入随机数来提升安全性,因此可以针对以下两种情况进行分析.

1) 唯密文可知模式.

定理 1 在唯密文可知模式下,本文所提机制比现有机制更加安全.

证明 现有机制中,若通过插入一个随机数来混淆相同索引及查询之间的关联性,则两次查询中该随机数相同的概率 $p = \frac{1}{v}$;若插入固定的 n 个随机数 ε_i,则两次查询中随机数之和相同的概率仍为固定值. 本文提出的对每次查询中插入随机数进行动态调节,任意两次查询中 $\sum_{i=1}^{M_j\&C_s}\varepsilon_i^v$ 相同的概率为非固定值,这显然增大了相同索引及查询密文之间的不可区分性,提升了唯密文可知模式下的抗分析攻击能力. □

2) 背景可知模式.

定理 2 在背景可知模式下,本文所提机制比现有机制更加安全.

证明 现有机制下,对于相同查询向量,插入固定数量的随机数,经过大量的分析积累,会泄露与明文的关联性. 本文所提机制通过每次检索目标集和关键字数组合,对插入哑元数进行扰动,使得加密查询向量与明文的差异性和无关联性增大,增加了关联分析的难度,提升了背景可知模式下的抗分析攻击能力. 同时,针对相同关键字的历次检索,相似度数值会有较大差异,既不会对于单次检索的相似度排序造成影响,又有效增加了云服务器对所收集的查询相似度数值与陷门之间关系的学习难度. □

3 实验结果

在 2.90 GHz 主频的 Core2 i7 处理器,Windows 7 操作系统下,使用 C 语言实现所提机制. 选用 3 000 篇近年来的 IEEE INFOCOM 出版物文档,关键字约 4 000 个. 基于文献 [20] 的 LSH 机制,使用一个 2 阶 $(\sqrt{3},2,0.56,0.28)$-LSH 来构建索引,Bloom Filter 长度设为 $m = 6\,000, l = 30, k = 10$. 与现有研究类似,在查询关键字中随机选择替换一个字母,并在一个查询中设置多于两个关键字.

3.1 效 率

1) 索引生成与加密. 虽然索引生成主要是在初始化时运行,但在实际应用中,随着新的文档和关键字的补充,不可避免会进行更新,陷门生成则是在每个检索过程都要做的操作. 索引生成针对每个文件(包括词根提取和布隆过滤器)生成两个步骤,如图 3 所示. 根据文件包含的关键字数目不同,这两个步骤的时间都呈线性增长,但由于使用了本文所提的改进索引生成机制,每个关键字映射向量长度大幅缩短. 因此,与现有机制相比,索引生成效率有较大提升,同时实现了对构建索引所需布隆过滤器长度的适当缩减. 如图 4 所示,索引加密主要通过矩阵的相乘,其时间消耗随文件数量的增加呈线性增长,但本机制加密效率有 20 % 左右的提升.

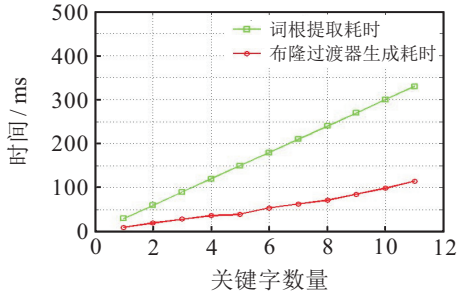

图 3 单文件关键字词根提取及索引构建耗时

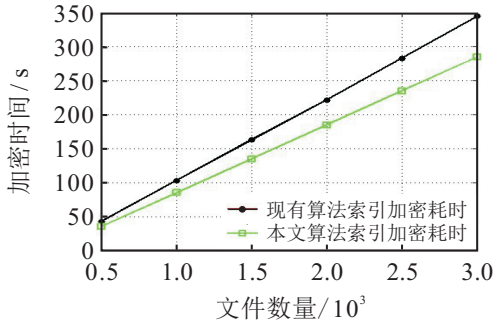

图 4　完整索引加密时间

2) 搜索时间. 由于本文基于对每个文件对应的索引进行搜索,影响搜索时间的重要因素在于目标文件数量. 图5显示,在查询关键字$k = 10$的情况下,搜索时间随文件数量的增加呈线性增长,由于索引规模的下降,搜索效率略有提升. 同时,不论查询关键字数量为多少,其都将被映射到一个布隆过滤器中,因此关键字数量的多少对搜索时间的影响很小,图6显示了这一结论.

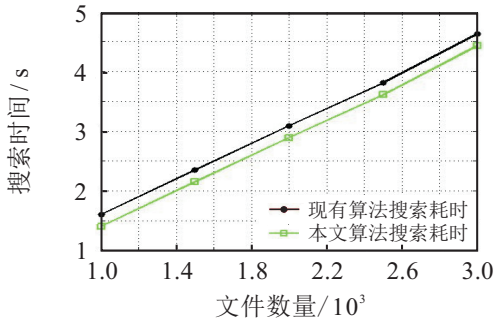

图 5　不同文件集规模下的搜索时间

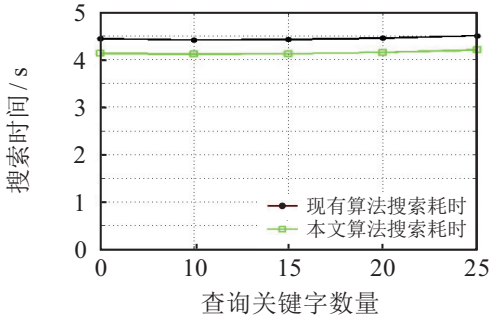

图 6　不同查询关键字数量下的搜索时间

3.2　准 确 率

本文重点关注的是向数据查询者返回top-k个相关的结果,因此主要使用准确率P_r来衡量结果,根据前述的规则对查询关键字进行模糊. 其中,$P_r = t_p/t_p + f_p$,t_p为“真阳性”结果,f_p为“假阳性”结果. 图7显示: 在关键字准确的情况下,随着搜索关键字数量的增加,由于“假阳性”的产生,搜索准确率P_r逐步小幅下降; 在引入模糊关键字的情况下,随着搜索关键字数量的增加,准确率P_r同样开始有小幅下降,但随着关键字数量的进一步增加,准确率又逐步回升. 这是因为受模糊关键字引起“假阳性”的影响,随着关键字数量增加,准确率反而会逐步下降.

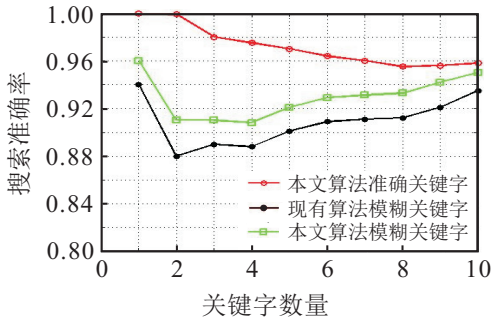

图 7　准确及模糊关键字条件下的搜索准确率

同时,对于两种模糊关键字情况,图7显示: 1) 当关键字中有两个字母顺序颠倒时,由于构建的映射向量是相同的,转换的索引与准确关键字是一致的,因此对准确率不产生影响; 2) 当关键字中增加或减少一个字母时,总体上看,与准确关键字搜索相比,会一定程度地降低准确率,且增加字母比减少字母降低的作用更大,但与现有研究中的模糊关键字搜索准确率相比,仍有一定程度提高,主要原因是本文所提关键字转换机制使得模糊关键字与准确关键字之间的距离更接近.

3.3　相似度变化率

关于相似度计算结果的加密保护,采用“相似度变化率”来评估,即在运用固定数目的哑元插入和根据混淆参数动态调节数目的哑元插入两种算法返回同样数量k个查询结果的前提下,实际属于top-k中的文档相似度排序发生变化的比率. 图8显示,根据混淆参数动态调节数目的哑元插入算法能够使更大比例的相似度排序发生变化,从而为查询提供更强的隐私保护.

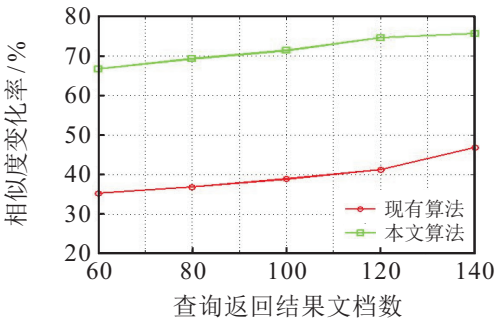

图 8　不同算法条件下的相似度变化率

4　结　论

本文研究了在云环境中数据加密保护条件下,针对可支持模糊检索的多关键字搜索问题. 基于现有技术,提出了一种能够在有效处理各种拼写错误的基础上,更为节约存储空间的关键字映射向量表达方

法,并针对如何更准确检索数据和进一步降低安全隐患,研究改进了相似度评估算法. 实验结果表明,所提机制和算法是可行和实用的. 未来的工作将重点聚焦在: 1) 如何更高效地进行索引添加更新; 2) 如何进一步提升密文搜索效率; 3) 如何借鉴明文检索技术,在密文域从语义的角度表达用户检索意图,从而进一步提升检索算法可用性; 4) 如何支持对加密中文本的检索.

参考文献(References)

[1] Wang B, Yu S, Lou W, et al. Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud[C]. 2014 Proc IEEE INFOCOM. Noida: IEEE, 2014: 2112-2120.

[2] Cheng X, Su S, Teng Y, et al. Enabling secure and efficient kNN query processing over encrypted spatial data in the cloud[J]. Security & Communication Networks, 2015, 8(17): 3205-3218.

[3] Song D X, Wagner D, Perrig A. Practical techniques for searches on encrypted data[C]. IEEE Symposium on Security and Privacy. Berkeley: IEEE, 2000: 44-55.

[4] Goh E J. Secure indexes[EB/OL]. (2004-05-16) [2017-04-15]. http://eprint.iacr.org/2003/216.

[5] Boneh D, Di C G, Ostrovsky R, et al. Public Key Encryption with Keyword Search[C]. Int Conf on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2004: 506-522.

[6] Boneh D, Waters B. Conjunctive, subset, and range queries on encrypted data[C]. Theory of Cryptography Conf. Berlin: Springer, 2007: 535-554.

[7] Wang C, Cao N, Li J, et al. Secure ranked keyword search over encrypted cloud data[C]. Int Conf on Distributed Computing Systems. Genova: IEEE, 2010: 253-262.

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

[9] Fu Z, Wu X, Guan C, et al. Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement[J]. IEEE Trans on Information Forensics & Security, 2016, 11(12): 2706-2716.

[10] Ballard L, Kamara S, Monrose F. Achieving efficient conjunctive keyword searches over encrypted data[C]. Int Conf on Information and Communications Security. Berlin: Springer-Verlag, 2005: 414-426.

[11] Hwang Y H, Lee P J. Public key encryption with conjunctive keyword search and its extension to a multi-user system[C]. Int Conf on Pairing-Based Cryptography. Tokyo: Springer-Verlag, 2007: 2-22.

[12] Cao N, Wang C, Li M, et al. Privacy-preserving multi-keyword ranked search over encrypted cloud data[C]. 2011 Proc IEEE INFOCOM. Shanghai: IEEE, 2011: 829-837.

[13] Li J, Wang Q, Wang C, et al. Fuzzy keyword search over encrypted data in cloud computing[C]. Conf on Information Communications. New York: IEEE Press, 2010: 441-445.

[14] Li M, Yu S, Cao N, et al. Authorized private keyword search over encrypted data in cloud computing[C]. Int Conf on Distributed Computing Systems. Minneapolis: IEEE, 2011: 383-392.

[15] Tang Q. Nothing is for free: Security in searching shared and encrypted data[J]. IEEE Trans on Information Forensics and Security, 2014, 9(11): 1943-1952.

[16] Wang J, Yu X, Zhao M. Privacy-preserving ranked multi-keyword fuzzy search on cloud encrypted data supporting range query[J]. Arabian J for Science and Engineering, 2015, 40(8): 2375-2388.

[17] Kumar K, Lu Y H. Cloud Computing for Mobile Users[J]. Computer, 2010, 43(4): 51-56.

[18] Sun W, Wang B, Cao N, et al. Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking[J]. IEEE Trans on Parallel & Distributed Systems, 2014, 25(11): 3025-3035.

[19] Porter M F. An algorithm for suffix stripping[M]. San Francisco: Morgan Kaufmann Publishers Inc, 1997: 130-137.

[20] Datar M, Immorlica N, Indyk P, et al. Locality-sensitive hashing scheme based on p-stable distributions[C]. The 20th Symposium on Computational Geometry. New York: ACM, 2004: 253-262.

(责任编辑: 齐 霁)