计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2017.20150751
54(2):295-304,2017

一种可抵抗统计攻击的安全索引

惠 榛[1,2] 冯登国[1,3] 张 敏[1] 洪 澄[1]
[1](中国科学院软件研究所可信计算与信息保证实验室 北京 1000190)
[2](中国科学院大学 北京 100049)
[3](计算机科学国家重点实验室(中国科学院软件研究所) 北京 100190)
(huizhen@tca.iscas.ac.cn)

A Secure Index Against Statistical Analysis Attacks

Hui Zhen[1,2], Feng Dengguo[1,3], Zhang Min[1], and Hong Cheng[1]
[1](*Laboratory of Trusted Computing and Information Assurance, Institute of Software, Chinese Academy of Sciences, Beijing* 100190)
[2](*University of Chinese Academy of Sciences, Beijing* 100049)
[3](*State Key Laboratory of Computer Science (Institute of Software, Chinese Academy of Sciences), Beijing* 100190)

Abstract Most of current searchable encryption schemes suffer from the threat of statistical analysis attacks. Some related works design their keyword/document trapdoors in a one-to-one method to avoid the threat, but it could lead to a severe overhead in searching cost. In the present paper, we design an efficient secure index to defend against a kind of statistical analysis attack. This scheme uses a Bloom filter to build indexes for each document. In order to save searching cost, one unique trapdoor is built for one word. To satisfy the security requirement, this scheme treats indexes of all documents as a matrix, and then adopts forged indexes and interpolation to make sure the frequencies of different words are closed and all indexes in the matrix are indistinguishable between each other. As a result, a particular word in the matrix cannot be recognized, thus the statistical analysis attack is resisted. In implementation, this scheme uses inverted indexes to further improve querying performance. The scheme is proved to be semantic security. Experimental results show that the querying performance of our scheme is double of Z-IDX at large dataset and words cannot be recognized based on their frequencies.

Key words searchable encryption (SE); statistical leakage; inverted index; Bloom filter; access pattern

摘 要 现有的大部分可检索加密方案建立的安全索引面临着统计攻击的威胁.为了抵抗统计攻击,部分方案设计出关键词/文档一一对应的陷门,以检索时多次的陷门计算为代价保证安全性,但是这样又导致检索速度过于慢而无法接受.为此,研究了针对密文的安全检索方案,在克服已有方案缺点的同时保证对于统计攻击的安全性.该方案使用 Bloom 过滤器为文档的关键词构造索引.为了确保检索效率,对于相同的关键词构造唯一对应的陷门.通过增加伪造的文档索引,并且在索引中进行插值来确保每个关键词在文档集合中出现的次数相似,从而达到语义安全并且能够抵抗统计攻击.在实现中,对索引进行倒排进一步提高检索效率.证明了本方案的安全性,且采用实验验证了其有效性和高效性.

关键词 可检索加密;统计泄露;倒排索引;Bloom 过滤器;访问模式

中图法分类号 TP309.2

收稿日期:2015-08-13;**修回日期**:2016-10-10

基金项目:国家自然科学基金重点项目(61230005);国家自然科学基金项目(61402456)
This work was supported by the Key Program of the National Natural Science Foundation of China (61230005) and the National Natural Science Foundation of China (61402456).

大数据时代的到来使得企业内部的数据管理面临新的挑战. 面对爆炸式增长的数据,传统的存储和管理方法变得困难和低效. 越来越多的新生业务选择将他们的数据外包到数据服务提供商,减少存储和管理开销. 但是这也给用户带来了新的安全和隐私保护问题——外包的数据中可能包含企业的财务状况、人员记录等等敏感信息. 为了保护信息,特别是敏感信息的安全,常用的方式是将数据进行加密之后再外包存储.

数据加密存储能够有效地保护数据安全,不过这会使对数据进行选择性的查询变得困难. 如何对密文进行有效且安全的检索,即可检索加密(searchable encryption, SE)成为了研究的热点问题. 可检索加密允许用户将密文存储到半可信/不可信的服务器,并且提供关键词匹配检索,同时能够满足用户的安全和隐私需求[1]. 针对这方面已有大量的研究工作,文献[2-9]在安全性、高效性和可用性等不同立足点上给出了各自的解决方案. 虽然解决方案众多,但是他们的基本框架是类似的:用户将文档加密后存储在服务器. 同时,为了实现对于加密文档的检索,用户需要构造一个能够存储在服务器的安全索引结构用来查找到包含指定关键词的文档. 对于任意关键词,用户能够构造合理的陷门,陷门将被用于检索指定的关键词. 如果服务器不能够调用陷门生成方法,也不能够从索引中推理出关键词信息,就能保证密文的安全. 在实际中,存储的数据不限于文档,可以是视频、音频、图像信息等,只要能够提取出其中的关键字:如基于标题或者标签.

大部分的方案保护了陷门和索引的安全,即攻击者不能够从中直接获得关键词或者明文的信息. 然而,攻击者仍然可以利用统计知识从索引的结构或者用户的查询(即用户发送给服务器的检索陷门)中获知哪些文档包含了某个关键词. 为了抵抗对索引本身静态的统计分析攻击,已有的方案有 2 种解决方法:1)第 1 种是为每个关键词分别构造匹配不同文档的独立陷门,这种方法在查询某一关键词时,需要进行 d(d 为文档总数)次的陷门构造和索引查询;2)第 2 种只需要为关键词构造 1 个陷门,但索引中采用类似链表的方式,依次计算获取包含关键词文档. 查询时只需要依次构造陷门,但是需要多次的子陷门运算来获得文档. 显然,两种方式的检索的计算代价都比较大. 此外,这些方案的访问模式(access pattern)均容易受到动态的统计分析攻击. 访问模式是指对于用户的每次检索,哪些文档包含了查询的关键词. 最近的研究中,Islam 等人[8]提出了一种针对访问模式的具体攻防方案. 但是方案的构造不具一般性,实际上并没有完全解决访问模式的安全问题.

本文从效率和抵抗统计攻击 2 方面考虑来构造索引,提出一种基于 Bloom 过滤器的可检索加密的方案:将索引进行整体考虑,采用倒排[10]减少检索计算代价,同时通过构造伪索引填充来保证索引的安全性. 由于 Bloom 过滤器[11]可用于检测元素是否属于集合,而且其空间效率和查询时间都优于一般算法,常常被用于检索技术之中. Bloom 过滤器对集合采用一个位串表示,并支持元素的 Hash 查找,既能表示集合,支持集合元素查询,又能有效地过滤掉不属于集合的元素. 因此本文采用 Bloom 过滤器而非直接加密关键词的方式构造索引. 倒排索引技术具有实现相对简单、查询速度快以及支持同义词查询等扩展功能的优点,被广泛应用. 将索引整体考虑,就可以形成 Bloom 过滤器的倒排形式,进一步提升检索效率. 本文主要贡献如下:

1) 分析了索引整体构建的统计分析安全问题.

2) 设计了一种安全且高效的安全索引,能够满足可检索加密的一般安全要求,同时抵抗统计分析攻击.

3) 在真实数据上验证了方案的可行性和高效性.

1 相关工作

检索加密数据的问题已经得到广泛的研究,大多数的工作着重于加强安全性以及优化效率. 由 Goldreich 等人[12]和 Ostrovsky[13]提出的不经意 RAM 不会向第三方泄露任何的信息,是解决此问题的一种经典方法. 然而,这种方案的实际可行性有待解决,因为它们需要对数多项式时间的计算代价和交互代价. 研究者通过降低安全性要求,提高效率和可行性,提出了一系列的可检索加密方案. 可检索加密最早由 Song 等人[14]提出的,到目前已经过了 10 余年的研究. 在文献[14]中,作者明确指出他们的方案会遭受统计攻击,对于加密的文档会泄露有价值的信息. 但是,并没有对该问题进行更近一步的讨论.

Goh[2]的 Z-IDX 方案首先给出了索引安全的形式化定义,描述了安全模型即抵抗选择关键词攻击的语义安全(IND-CKA)以及更强的 IND-CKA2. 同时,文章考虑到统计攻击的问题,利用双重 Hash 来

构造 Bloom 过滤器,在第 2 次计算时加入文档标识符.这样使得相同关键对应的 Bloom 过滤器不相同,使得敌手不可比较.这也导致了在检索某个关键词时,需要计算其相对于每一个文档的陷门,降低了检索的效率.Chang 等人[4]提出的安全定义与 IND-CKA2 相似,但是要求陷门也不会泄露任何的信息.Kurosawa 等人[5]考虑了安全中除了隐私之前的另一个方面:可靠性,也就是要求服务器不能删除或修改用户的索引和密文信息.并且证明了隐私性和可靠性等价于通用可组合安全(UC 安全).Li 等人[9]提出了一种弱化第三方的可检索加密方案,允许多个用户之间进行加密文档的共享,而非对拥有唯一属主的文档进行检索.上述的方案都是使用对称密钥加密的数据,因此也被称作可检索对称加密(SSE).此外有工作研究了公钥的情况.Boneh 等人[15]提出了支持关键词检索的公钥加密(PEKS),该方案的陷门生成函数是概率算法.

除了不经意 RAM 等理论性方法之外,上述实际可行的可检索加密方案都会泄露访问模式信息,面临遭受统计分析统计的威胁.Islam 等人[8]对于可检索加密的访问模式泄露问题进行了研究,给出了一种针对访问模式信息泄露的攻击以及一种解决方案.但是其安全目标只是为了保护"两次搜索结果的交集"这一隐私信息,不能完全解决访问模式问题.为此,需要提出一种更具一般性的攻击模型,并针对性地设计安全索引.

2　问题描述

为了安全地存储和检索存储在云端的文档集合 $D=\{D_1,D_2,\cdots,D_d\}$,客户端上传加密的文档集 $C=\{C_i|C_i=\varepsilon(D_i),1\leqslant i\leqslant d\}$ 以及安全索引 $\boldsymbol{I}$.ε 是通用的对称加密算法,其安全性由算法本身保证.W 表示文档所包含的不相同的关键词,即关键词集合 $W=\bigcup_{i=1}^{d}W_i$,其中 W_i 表示 D_i 所包含的关键词.记 $t=|W|$,表示共有 t 个不同的关键词.当查询某指定关键词 w 时,用户构造查询对应的陷门 $\boldsymbol{I}_w$ 并发送到服务器,服务器进行检索并且返回检索结果.

2.1　统计分析攻击

由于索引信息和明文信息的统计特征密切相关,这就为攻击者分析索引推理出明文提供了可能性.对于推理攻击可以用不同的方式进行建模,文献[16]给出了常用的模型,假设攻击者知道关键词在明文中的分布.从而,攻击者可以推理出明文内容,即某个关键词是否存在于特定的文档,这将为攻击者定位攻击目标提供便利.统计分析攻击可以依据静态的索引进行,也可以依据动态的检索结果,即访问模式进行.两者的区别在于,前者可直接依据索引直接分析包含某个关键词的信息,而后者需要监控查询过程,从检索结果分析包含某个关键词的信息,然而两者的分析方法都是相同的.

下面形式化地描述统计分析攻击的模型.记 q 为某一个攻击者感兴趣的关键词(或者对该关键词的查询).对于文档集合中的每个关键词 w_i,其在文档集合中出现的次数为 f_{w_i}.假设攻击者可以获得关于关键词出现频次的一个分布:$\mathcal{D}=(w_i,f_{w_i})_{i=1}^{t}$.该分布和用户存储的文档关键词的真实分布一般而言大体相似,但是存在一些差异.因此在计算关键词频次的差距时,需要考虑的不是频率完全相同的,而是与目标关键词频率相差一定范围内的.对于 q 的统计分析攻击,记作 ATK^{stats},过程如下.

$ATK^{\text{stats}}(\mathcal{D},q,\theta)$:

1) 令 $U=\{i|i\in\{1,2,\cdots,t\},|f_q-f_{w_i}|\leqslant\theta\}$;

2) 返回 $G_q=\{w_i|i\in U\}$.

攻击的效果取决于关键词的词频分布,G_q 表示与 q 词频相同或者相近(相差不超过 θ)的关键词集合,$1/|G_q|$ 为攻击准确率.与查询 q 词频相同或者相近的关键词越少,则集合 G 越小,攻击效果越好;反之,当所有关键词频率都在范围 θ 内时,ATK^{stats} 等同于随机猜测,此时攻击失效,此时攻击准确率为 $1/t$.

定义 1. ATK^{stats} 攻击优势.对于查询 q 的 ATK^{stats} 攻击优势为:$M(q)=1/|G_q|-1/t$,如果 $M(q)$ 是可忽略的,那么 ATK^{stats} 攻击无效.

文档关键词的词频已被证实符合 Zipf 法则[17]——单词的频次 f 与其频率排名 r 成反比.那么,对于高频词,其攻击返回集合 G 较小;对于低频次,其攻击返回集 G 较大.也就是说对于高频词,ATK^{stats} 攻击准确率可接近 100%;对于低频词,攻击准确率降低,但是仍然能够有效减少需要攻击的文档数目.

为了说明统计分析攻击,考虑如表 1 和表 2 所示的例子.表 1 表示用户的文档及其分别包含的关键词.表 2 为对这个文档所构建的最简单的密文索引.该索引对所有的关键词进行加密,用户检索时,提交所要检索的关键词密文(即陷门)到服务器,服务器返回所有包含该关键词的文档.

Table 1 Keywords of Each Document

表 1 每个文档的关键词

Document	Keyword
D_1	w_1, w_{21}, w_2
D_2	w_3, w_{21}, w_{31}
D_3	w_4, w_5, w_6
D_4	w_7, w_8, w_{31}
D_5	w_{22}, w_9, w_{10}
D_6	w_{22}, w_{11}, w_{31}

Table 2 Encrypted Keywords of Each Document

表 2 每个文档加密后的关键词

Document	Keyword
D_1	π, α, μ
D_2	ω, α, κ
D_3	ξ, π, η
D_4	α, γ, κ
D_5	ζ, δ, θ
D_6	ζ, τ, κ

然而,密文索引有明显的统计特征.如果攻击者明确知道文档索引的明文,或关键词在文档中的出现次数,他将很容易推测出密文所对应的明文关键词,从而定位到感兴趣的文档.在这个上例中,攻击者可以直接推断出 κ 为 w_{11} 对应的检索陷门,且以50%的概率推测出 α, ζ 分别对应于 w_{21}, w_{22}.此时,攻击优势分别为:

$$M(\kappa)=1-\frac{1}{14}=92.9\%,$$

$$M(\zeta)=M(\alpha)=\frac{1}{2}-\frac{1}{14}=42.9\%.$$

2.2 研究目标

现有的密文检索方案,为了抵抗统计分析攻击,对于不同的文档会采用不同的密钥或者加入标志信息构造索引,如图 1(a)所示. Goh[2] 构造的 Z-IDX 方案,在构建索引时,需要经过 2 轮的伪随机置换,并且在第 2 轮的置换中加入文档标志,生成最后的索引. Cash 等人[7] 的 Π_{2lev} 方案,在索引构造时利用了一个计数器,通过累加来区分不同的关键词/文档对应关系.此类方案构造的索引不再泄露这种统计信息,但是在检索时,需要为关键词计算 $|D|$ 个不同的陷门,增加了检索代价.在文献[7]中对于 q 的查询,需要进行 f_q 次解密操作;在文献[2]中对于任意关键词的检索,都需要进行 $|D|$ 次 Bloom 过滤器查询操作.因此,我们的研究目标是抵抗统计分析攻击,同时提高检索效率,将检索过程中的计算次数降低为常数级别.

我们的方案构造了一种直接的关键词到陷门的映射,如图 1(b)所示.该陷门对于所有文档的索引同等适用,从而可以进一步整合单个文档的索引,提高检索效率.

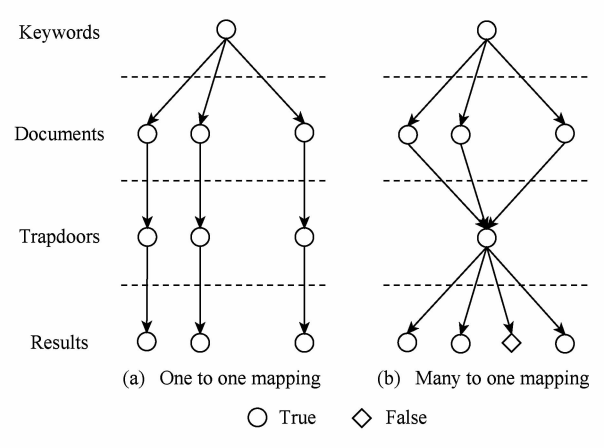


Fig. 1 Relations of keywords, documents, trapdoors and results

图 1 关键词、文档、陷门和检索结果的关系

3 安全需求

本节讨论可检索加密方案的安全需求,将其分为 2 部分:可检索加密的基本安全需求和抵抗统计分析攻击的 δ-安全性.首先,介绍需要用到的符号.

$x \xleftarrow{R} S$ 表示随机变量 x 是从集合 S 中均匀随机选取的,集合 A 和 B 的对称差表示为 $A\Delta B=(A-B)\bigcup(B-A)$, $|A|$ 表示集合 A 的势.

3.1 可证明安全

可证明安全是可检索加密的基本要求,现有的绝大多数方案采用对选择密钥攻击的语义安全[2-4,9]作为索引方案的安全定义,这要求对于索引具有不可区分性.这里,我们考虑同样的安全需求,利用对选择关键词攻击的不可区分性 IND-CKA 来证明方案对于选择关键词攻击的语义安全.

定义 2. IND-CKA. IND-CKA 是挑战者 $\mathcal{C}$ 和攻击者 $\mathcal{A}$ 之间的游戏,由如下 4 个过程组成:

1) 设置. 攻击者 $\mathcal{A}$ 选择一个文档集合 $\{D_1, D_2, \cdots, D_d\}$ 并且从挑战者 $\mathcal{C}$ 获取对应的索引.

2) 查询. $\mathcal{A}$ 可以向 $\mathcal{C}$ 查询关键词 x,获得对应的陷门 T_x,利用陷门,$\mathcal{A}$ 可以搜索包含 x 的文档索引.

3) 挑战. 在进行了一系列的陷门查询之后,$\mathcal{A}$

选择 2 个文档 D_0,D_1 使得 $|D_0 \Delta D_1| \neq 0$. 并且,$\mathcal{A}$ 不得向 $\mathcal{C}$ 查询属于 $D_0 \Delta D_1$ 的关键词的陷门. 之后,$\mathcal{A}$ 将 D_0,D_1 发送给 $\mathcal{C}$. $\mathcal{C}$ 选择 $b \xleftarrow{R} \{0,1\}$,建立 D_b 的索引 I_{D_b},将 I_{D_b} 返回给 $\mathcal{A}$. 对于 $\mathcal{A}$ 的挑战就是判断 b 的值. 在挑战发起之后,$\mathcal{A}$ 不得向 $\mathcal{C}$ 查询属于 $D_0 \Delta D_1$ 的关键词的陷门.

4) 应答. 最终,$\mathcal{A}$ 输出 b' 代表其对 b 的猜测. 称 $\mathcal{A}$ 赢得该游戏,如果 $b'=b$. $\mathcal{A}$ 赢得该游戏的优势为 $Adv_{\mathcal{A}}=|Pr[b=b']-1/2|$,即 $\mathcal{A}$ 的猜测概率与随机猜测的差. 称 $\mathcal{A}$ 以 ε-优势赢得游戏,如果 $Adv_{\mathcal{A}} \geqslant \varepsilon$.

IND-CKA 是可检索加密方案基本的安全需求.

3.2 δ-安全

由 2.1 节对于统计分析攻击的描述可知,统计分析攻击得以实现的根本原因是关键词词频分布不相同. 从而,消除索引中关键词的词频差异是抵抗统计分析攻击的一种有效方法.

定义 3. δ-安全. 从索引中获知的文档关键词的词频相差都不超过 $\delta \in \mathbb{N}$,并且服从相同的分布 $\mathcal{D}$.

定理 1. 满足 δ-安全的索引能够抵抗攻击推理攻击.

证明. 由定义 3 可知,在满足 δ-安全的索引中,对于任意查询 q,q 与其他所有关键词的词频差距 $|f_{w_i}-f_q| \leqslant \delta$. 而且由于所有关键词词频分布相同,攻击者需要令 $\theta \geqslant \delta$,才能确保 q 对应的关键词出现在 G 中. 此时,对任意查询 q 的攻击优势均为 $M(q)=1/t-1/t=0$. 因此,满足 δ-安全的索引能够抵抗统计分析攻击. 证毕.

4 索引方案构造

为了提高查询效率,同时满足通用安全需求并且能够抵抗统计分析攻击,本文提出了一种基于 Bloom 过滤器的倒排索引方案.

4.1 Bloom 过滤器

Bloom 过滤器[11]用 1 个 m 位数组表示包含有 n 个元素的集合 $S=\{s_1,s_2,\cdots,s_n\}$. 开始阶段,数组的所有元素都被初始化为 0. Bloom 过滤器使用 k 个独立的 Hash 函数 $h_1,h_2,\cdots,h_k$,其中 $h_i:\{0,1\}^* \rightarrow [1,m]$,$i \in [1,k]$,对于每一个元素 $s \in S$,在数组中的 $h_1(s),h_2(s),\cdots,h_k(s)$ 这些位置的值都被置为 1. 同一个位置可以被置 1 多次,然而只需要注意第 1 次. 为了判定元素 a 是否属于集合 S,需要检查 $h_1(a),h_2(a),\cdots,h_k(a)$ 这些位置的值. 如果所有的值都为 1,那么就认为 a 是该集合的成员. 然而,这种方式存在着误判,即实际上不是集合 S 成员的元素 a 会被判定为属于该集合. 导致出现误判的原因是:a 对应的每一个位置都可能被其他元素而不是 a 本身置 1. 另一方面,如果有一个位置的值是 0,那么 a 一定不是集合 S 的成员.

4.2 符　号

为方便说明,现将索引构造中用到的符号定义如下:

D 表示系统中的全部文档,即有 d 个元素的文档集 $D=\{D_1,D_2,\cdots,D_d\}$,其中 $D_i=W_i$,$W_i \subseteq \{0,1\}^*$ 为关键词的集合. 规定 $n=|W_i|$,即从每个文档中抽取 n 个关键词.

W 表示文档所包含的不相同的关键词,即关键词集合 $W=\bigcup_{i=1}^{d} W_i$. 记 $t=|W|$,表示共有 t 个不同的关键词.

$\boldsymbol{BF}$ 表示 m 位长的 2 元向量,即 Bloom 过滤器.

SK 表示用户私钥,包含 k 个私钥,$SK=\{sk_1,sk_2,\cdots,sk_k\}$.

$D(w)$ 表示包含关键词 w 的文档集合.

$\boldsymbol{BF}_1 \& \boldsymbol{BF}_2$ 表示 2 个 $\boldsymbol{BF}$ 按位与运算.

4.3 基本方案构造:SE-1

为了构造索引,需要处理 2 部分的内容:一部分是文档对应的索引信息,另一部分是填充的索引信息. 对于每一个文档 D_i,构造 1 个 m 位的 Bloom 过滤器 $\boldsymbol{BF}_i$. 为文档 D_i 中每个关键词在 $\boldsymbol{BF}_i$ 中插入 k 个伪随机值,形成文档对应的索引 $\boldsymbol{I}_i$. 统计关键词在文档中出现的频次并且按词频排序,记作 $(w_{ri},f_i)_{i=1}^{t}$. 对于填充部分,构造 l 个 m 位的 Bloom 过滤器 $\boldsymbol{BF}_j$,$j \in [1,l]$,并且为每个 $\boldsymbol{BF}_j$ 维护一个计数器 ctr_j. 对于每一个关键词 w_{ri},随机选择 $c=f_1-f_i$ 个计数器值不为 n 的 Bloom 过滤器,插入 k 个伪随机值,并且将对应计数器加 1. 插入最后可能使 l 增大,这时更新 l 即可. 此外,可能有的 Bloom 过滤器的计数器小于 n,此时选择集合 W 之外的关键词插入,使得所有填充的 Bloom 过滤器的计数器值为 n,即都包含 n 个元素. 此后,将这些共 $d+l$ 个 Bloom 过滤器进行转置,构成最终的索引 $\boldsymbol{I}_D$. 可以看出,$\boldsymbol{I}_D$ 为一个 $m \times (d+l)$ 的矩阵. 此外,为了能区分真实文档和伪造的填充,需要维护一个填充列表 $FList$ 记录填充的索引.

检索关键词 w 时,首先生成对应的检索陷门 T_w,即 m 位的位串,其中 k 位为 1. 在索引 $\boldsymbol{I}_D$ 中取出

T_w 中为 1 的 k 个位置对应的行 $\boldsymbol{I}_{y_1},\boldsymbol{I}_{y_2},\cdots,\boldsymbol{I}_{y_k}$. 例如,$T_w=001100$,则取出 $\boldsymbol{I}_D$ 中的第 3 行和第 4 行. 之后计算$\widetilde{\boldsymbol{BF}}=\boldsymbol{I}_{y_1}\&\boldsymbol{I}_{y_2}\&\cdots\&\boldsymbol{I}_{y_k}$,返回$\widetilde{\boldsymbol{BF}}$中为 1 的位所对应的序号 $R=\{id_1,id_2,\cdots,id_q\}$. 最后,利用 $FList$ 过滤掉伪造填充文档,获得检索结果. 可以看出,SE-1 的查询用时是常数级的. 因为对于一个查询,只需要进行 k 次按位与预算和一次筛选.

方案 SE-1 包括 5 个概率多项式时间算法:$Keygen(m,k)$,$BuildIndex(D,SK)$,$Trapdoor(w,SK)$,$SearchIndex(T_w,\boldsymbol{I}_D)$和 $Filter(R,FList)$.

算法 1. $Keygen(m,k)$.

给定参数 m 和 k.

构造 $SK=\{sk_1,sk_2,\cdots,sk_k\}\xleftarrow{R}\{0,1\}^{k\text{lb}m}$作为密钥,选择伪随机函数 $h:\{0,1\}^*\times\{0,1\}^{\text{lb}m}\to\{0,1\}^{\text{lb}m}$.

算法 2. $BuildIndex(D,SK)$.

给定文档集合 D,密钥 $SK=\{sk_1,sk_2,\cdots,sk_k\}$.

1) 构造文档对应的 Bloom 过滤器

对于 $D_i(1\leqslant i\leqslant d)$中的每一个关键词 w_{ij},计算陷门$(y_1=h(w_{ij},sk_1),y_2=h(w_{ij},sk_2),\cdots,y_k=h(w_{ij},sk_k))$,将其对应的 Bloom 过滤器 $\boldsymbol{BF}_i$,对应位置置 1.

2) 构造填充的 Bloom 过滤器

① 计算关键词频次 $f_1=|D(w_{r1})|$,$f_2=|D(w_{r2})|,\cdots,f_t=|D(w_n)|$.

② 初始化填充的 Bloom 过滤器 $S=\{\boldsymbol{BF}_1,\boldsymbol{BF}_2,\cdots,\boldsymbol{BF}_l\}$,$l=\sum_{i=1}^{t}(f_1-f_i)$,以及集合 $V=\varnothing$.

③ 初始化计数器. 对于 S 中的每一个 $\boldsymbol{BF}_i$,构造计数器 $ctr[\boldsymbol{BF}_i]=n$.

④ 插值和填充 Bloom 过滤器.

(i) 对于每一个 $w\in W$:

计算其陷门$(y_1=f(w,sk_1),y_2=f(w,sk_2),\cdots,y_k=f(w,sk_k))$,以及需要填充的个数 $c=f_1-f_w$;

若 $|S|<c$,在 S 中添加 $\boldsymbol{BF}$,使得 $|S|=c$,并构造对应计数器;

从 S 中随机选择 c 个 $\boldsymbol{BF}$,即 $\boldsymbol{BF}_{s1},\boldsymbol{BF}_{s2},\cdots,\boldsymbol{BF}_{sc}$,将每个 $\boldsymbol{BF}$ 的对应位置置 1;

对于 $i=1,2,\cdots,c$,更新计数器 $ctr[\boldsymbol{BF}_{si}]\leftarrow ctr[\boldsymbol{BF}_{si}]-1$;若 $ctr[\boldsymbol{BF}_{si}]=0$,将 $\boldsymbol{BF}_{si}$从 S 移至 V.

(ii) 若 $|S|\neq 0$,对于每个 $\boldsymbol{BF}\in S$:

选择 $ctr[\boldsymbol{BF}]$个关键词 $\tilde{w}\in\{0,1\}^*\backslash W$;

计算$(y_1=f(\tilde{w},sk_1),y_2=f(\tilde{w},sk_2),\cdots,y_k=(\tilde{w},sk_k))$,将其在 $\boldsymbol{BF}$ 中的对应位置置 1;

将 $\boldsymbol{BF}$ 从 S 移至 V.

(iii) 令 $l\leftarrow|V|$.

3) 构建倒排索引 $\boldsymbol{I}_D$

将生成的 $d+l$ 个 $\boldsymbol{BF}$ 重新随机排序,排序后记作 $\boldsymbol{I}'_1,\boldsymbol{I}'_2,\cdots,\boldsymbol{I}'_{d+l}$. 记录填充的 $\boldsymbol{BF}$ 对应的索引编号 $FList=\{id_{f1},id_{f2},\cdots,id_{fl}\}$. 从每一个 $\boldsymbol{I}'_i$ 中选择各自的第 v 比特 $b_{i,v}$,构造 $\boldsymbol{I}_v=[b_{1,v},b_{2,v},\cdots,b_{d+l,v}]$,$v\in[1,m]$. 最终 $\boldsymbol{I}_D=[\boldsymbol{I}_1,\boldsymbol{I}_2,\cdots,\boldsymbol{I}_m]^{\mathrm{T}}$.

算法 3. $Trapdoor(w,SK)$.

给定关键词 w 和密钥$\{sk_1,sk_2,\cdots,sk_k\}$.

计算关键词 w 的陷门为 $T_w=(y_1=f(w,sk_1),y_2=f(w,sk_2),\cdots,y_k=f(w,sk_k))$,$T_w\in\{0,1\}^{k\text{lb}m}$.

算法 4. $SearchIndex(T_w,\boldsymbol{I}_D)$.

给定检索关键词 w 的陷门 $T_w=(y_1,y_2,\cdots,y_k)$以及文档的倒排索引 $\boldsymbol{I}_D$.

计算$\widetilde{\boldsymbol{BF}}=\boldsymbol{I}_{y_1}\&\boldsymbol{I}_{y_2}\&\cdots\&\boldsymbol{I}_{y_k}$,返回其中值为 1 的位所对应的序号 $R=\{id_1,id_2,\cdots,id_q\}$.

算法 5. $Filter(R,FList)$.

给定查询结果集 R 和填充记录列表 $FList$.

计算真实文档 $finalR=R\backslash FList$.

4.4 安全分析

本节分析方案是否满足对于选择关键词攻击的不可区分性以及是否 δ-安全.

定理 2. 索引方案 SE-1 满足 IND-CKA,如果 f 是一个伪随机函数.

证明. 在上文中 SE-1 的构造所示,$\boldsymbol{I}_D$ 是$\boldsymbol{I}'$的倒排,因此两者具有相同的安全性. 并且,对于填充的索引部分,填充的方式决定了其和正常索引的不可区分,因此 $\boldsymbol{I}_D$ 的安全性就等同于 $\boldsymbol{I}''=[\boldsymbol{I}'_1,\boldsymbol{I}'_2,\cdots,\boldsymbol{I}'_{|D|}]$的安全性,下面证明 $\boldsymbol{I}''$的安全性.

假设 SE-1 不是对于选择关键词攻击语义安全的,那么,存在算法 $\mathcal{A}$ 能够在多项式时间内以 ε 一优势赢得游戏. 我们构造另一个算法 $\mathcal{A}'$,该算法将算法 $\mathcal{A}$ 当作子算法来判断函数 f 是伪随机函数或是随机函数. 在调用 $BuildIndex$ 和 $Trapdoor$ 算法时,$\mathcal{A}'$询问预言机 $\mathcal{O}_f$ 得到未知函数 $f:\{0,1\}^*\to\{0,1\}^m$. $\mathcal{A}'$利用 $\mathcal{A}$ 的步骤如下:

设置. $\mathcal{A}'$随机选择一个多项式数目的关键词的集合 S,将 S 发送给 $\mathcal{A}$. $\mathcal{A}$ 选择 S 的一些子集返回给 $\mathcal{A}'$,记作 S^*. 对于 S^* 中的任意元素 D_i,$\mathcal{A}'$构造对应的索引 $\boldsymbol{I}'_i$. 在生成所有的索引之后,$\mathcal{A}'$将索引以及其对应的关键词子集发送给 $\mathcal{A}$.

查询. 收到 $\mathcal{A}$ 对于关键词 x 的查询之后,$\mathcal{A}'$ 运行 $Trapdoor$ 算法,返回陷门 T_x.

挑战. 在进行过一定数目的查询之后,$\mathcal{A}$ 选择 2 个关键词集合 $D_0, D_1 \in S^*$,使得 $|D_0 \Delta D_1| \neq 0$. 并且,$\mathcal{A}$ 不可向 $\mathcal{A}'$ 查询过任何属于 $D_0 \Delta D_1$ 的关键词. 之后,$\mathcal{A}$ 将 D_0, D_1 发送给 $\mathcal{A}'$,$\mathcal{A}'$ 选择 $b \xleftarrow{R} \{0,1\}$,构建 D_b 对应的索引 $\boldsymbol{I}_b$ 将其发送给 $\mathcal{A}$. 在挑战发起之后,$\mathcal{A}$ 不得向 $\mathcal{A}'$ 查询属于 $D_0 \Delta D_1$ 的关键词的陷门.

应答. 最终,$\mathcal{A}$ 输出 b' 代表其对 b 的猜测. 称 $\mathcal{A}$ 赢得该游戏,如果 $b'=b$. $\mathcal{A}'$ 输出 0,表示其猜测 f 为伪随机函数;否则,$\mathcal{A}'$ 输出 1.

下面,我们证明 $\mathcal{A}'$ 能够以大于 ε 的优势判断 f 为伪随机函数或者随机函数.

情况 1. 当 f 为伪随机函数. 此时 $\mathcal{A}'$ 完全模拟了 IND-CKA 游戏中的挑战者,那么由 $\mathcal{A}$ 的定义可知

$$|Pr[\mathcal{A}'^{(\cdot)}_{f_k}=0 \mid k \xleftarrow{R} \{0,1\}^m] - 1/2| \geqslant \varepsilon.$$

情况 2. 当 f 为随机函数,首先证明分析中只需要考虑挑战涉及的 2 个关键词集合. 也就是 S^* 中其他的元素不会泄露关于挑战集合的信息. f 作为一个随机函数,因此对于 $\mathcal{A}$ 来说在索引中关联同一个关键词的编码是不可行的,否则 $\mathcal{A}$ 能预测一个随机函数的输出. 根据这个推论,加上对于选择集合 D_0, D_1 的限制,以及发起挑战之后的查询限制,从 $\mathcal{A}$ 的角度,对于关键词 $z \in D_0, D_1$,其陷门和 S^* 中其他关键词的陷门是相互独立的. 即 $\mathcal{A}$ 从 S^* 的其他元素中不能学习到关于 D_0, D_1 的知识.

假设 D_0, D_1 包含 2 个关键词 x, y 并且 $x \in D_0$, $y \in D_1$. 假设 $\mathcal{A}$ 以优势 $\theta > 0$ 猜对 b. 那就是说,给定 $f(z)$,$\mathcal{A}$ 能够以优势 $\theta > 0$ 判断 $z=x$ 或 $z=y$. 即 $\mathcal{A}$ 能够以优势 $\theta > 0$ 区分随机函数的输出. 然而这是不可能的. $\mathcal{A}$ 最多只能以 1/2 的概率猜对 b. 由此可见

$$Pr[\mathcal{A}'=0 \mid g \xleftarrow{R} \{F:\{0,1\}^* \to \{0,1\}^m\}] = Pr[b'=b] = 1/2.$$

综上,

$$|Pr[\mathcal{A}'^{(\cdot)}_{f_k}=0 \mid k \xleftarrow{R} \{0,1\}^m] - Pr[\mathcal{A}'=0 \mid g \xleftarrow{R} \{F:\{0,1\}^* \to \{0,1\}^m\}]| \geqslant \varepsilon.$$

证毕.

下面讨论 SE-1 对于统计分析攻击的安全性,需要分 2 步进行. 首先需要证明填充的索引项和由文档生成的索引项是不可区分的. 在此基础之上,证明方案是 δ-安全性的,即能够抵抗统计分析攻击.

定义 4. 文档索引与填充索引不可区分性. 若文档索引和填充索引都具有良好的随机性,则它们不可取分.

定理 3. SE-1 具有文档索引与填充索引不可区分性.

证明. 从构造方式可知,SE-1 的文档索引和填充索引不可区分. 对任意文档索引,其构造看作是 $n \times k$ 个伪随机函数的联合输出,即

$$F(w_1, w_2, \cdots w_n, sk_1, sk_2, \cdots, sk_k) = f(w_1, sk_1) \| f(w_1, sk_2) \| \cdots \| f(w_1, sk_k) \| \cdots \| f(w_n, sk_1) \| f(w_n, sk_2) \| \cdots \| f(w_n, sk_k),$$

显然这个输出也是伪随机的.

对于任意的填充索引,有 2 种情况:1) 未使用不属于 W 的关键词作为输入构造;2) 使用不属于 W 的关键词作为输入构造. 然而,二者最终都是选择了 n 个属于 $\{0,1\}^*$ 的值作为输入. 因此输出结果同样可表示为:

$$F(\dot{w}_1, \dot{w}_2, \cdots \dot{w}_n, sk_1, sk_2, \cdots, sk_k) = f(\dot{w}_1, sk_1) \| f(\dot{w}_1, sk_2) \| \cdots \| f(\dot{w}_1, sk_k) \| \cdots \| f(\dot{w}_n, sk_1) \| f(\dot{w}_n, sk_2) \| \cdots \| f(\dot{w}_n, sk_k).$$

证毕.

定理 4. 索引方案 SE-1 具有 δ-安全性.

证明. 由填充算法可知,对于任意 $w \in W$,有 f 个索引包含该关键词. 考虑到 Bloom 过滤器的误判率,可知,在剩下的 $d+l-f$ 个索引中,可能出现误判的情况,导致关键词 w 的词频增加. 已知在一个 Bloom 过滤器中,误判率为:

$$f_p = [1-(1/m)^{kn}]^k \approx (1-\mathrm{e}^{-kn/m})^k,$$

那么 w 被误判为属于原本不包含它的索引的概率为:

$$p = \frac{1}{\mathrm{C}_m^k} \times f_p = \frac{(1-\mathrm{e}^{-kn/m})^k}{\mathrm{C}_m^k},$$

则 w 因误判而增加的频次 $\tilde{f}$ 服从参数为 $d+l-f$ 和 p 的二项分布. 令 $\delta = \tilde{f}$,从而 w 的总频次在大小为 δ 的区间 $[f, f+\delta]$ 内且满足分布 $(\dot{f} = f + \tilde{f}) \sim f + B(d+l-f, p)$. 证毕.

4.5 改进方案构造 SE-2

SE-1 的构造能够有效地提供索引的安全性,同时保证检索的效率. 然而,将所有关键词都填充到与词频最高的关键词一样的数目,会增加索引的存储代价. 同时,这样的安全性也过于严格. 考虑到 Zipf 法则描述的词频曲线,可知,对于低词频部分的关键词,攻击的优势已经可以很小,可以忽略,容易被攻击的只是高词频部分的关键词. 因此对于 SE-1 在构建

索引的改进主要在于对关键词进行选择性填充,从而降低存储代价.我们需要能够保证抵抗统计分析攻击的更加宽松的安全定义.

定义 5. (α,δ)-安全.从索引中获知的任意文档关键词,至少存在一个大小为 $\alpha\in[1,t]$ 且包含该关键词的集合,使得它们词频都在某个的 $\delta\in\mathbb{N}$ 区间内,并且服从相同的分布 $\mathcal{D}$.

显然,满足(α,δ)-安全的索引也能够抵抗统计分析攻击.由于此时,对任意查询 q 的攻击优势最大为$M(q)=1/\alpha-1/t$.只要 α 足够大,则攻击优势可以忽略.

SE-2 的构造与 SE-1 十分相似,区别只出现在构造索引算法的填充部分.不失一般性,假设 t 整除 α,记$b=t/\alpha$.将词频排序的关键词按词频从大到小依次分为 b 段,则每段共有 α 个关键词.对于第 j 段中的关键词w_{ji},需要在 $c_j=f_{ji}-f_{j\alpha+1}$,$j\in[0,b-1]$ 个 Bloom 过滤器中进行填充.填充的具体方法与 SE-1 相同.

方案 SE-2 由 5 个概率多项式时间算法组成:$Keygen(m,k)$,$BuildIndex(D,SK)$,$Trapdoor(w,SK)$,$SearchIndex(T_w,\boldsymbol{I}_D)$ 和 $Filter(R,FList)$.其中,$Keygen(m,k)$,$Trapdoor(w,SK)$,$SearchIndex(T_w,\boldsymbol{I}_D)$,$Filter(R,FList)$ 与 SE-1 完全相同,下面给出 $BuildIndex(D,SK,\alpha)$算法.

算法 6. $BuildIndex(D,SK,\alpha)$.

给定文档集合 D,密钥 $SK=\{sk_1,sk_2,\cdots,sk_k\}$,安全参数 α.

1) 构造文档对应的 Bloom 过滤器

对于 $D_i(1\leqslant i\leqslant d)$中的每一个关键词 w_{ij},计算该关键词的陷门$(y_1=h(w_{ij},sk_1),y_2=h(w_{ij},sk_2),\cdots,y_k=h(w_{ij},sk_k))$,将其对应的 Bloom 过滤器 $\boldsymbol{BF}_i$ 对应位置置 1.

2) 构造填充的 Bloom 过滤器

① 计算关键词频次 $f_1=|D(w_{r1})|$,$f_2=|D(w_{r2})|$,$\cdots$,$f_t=|D(w_{rt})|$.

② 初始化填充的 Bloom 过滤器 $S=\{\boldsymbol{BF}_1,\boldsymbol{BF}_2,\cdots,\boldsymbol{BF}_l\}$,$l=\sum_{j=0}^{b-1}\sum_{i=j\alpha+1}^{(j+1)\alpha-1}(f_{j\alpha+1}-f_i)$,以及集合 $V=\varnothing$.

③ 初始化计数器.对于 S 中的每一个 $\boldsymbol{BF}$,构造计数器 $ctr=n$.

④ 将 W 按频次高低分为 $b=t/\alpha$ 个集合 W_1,W_2,$\cdots$,W_b.

(i) 对于每一个 $w\in W_j$,$j\in[1,b]$:

计算其陷门$(y_1=f(w,sk_1),y_2=f(w,sk_2),\cdots,y_k=(w,sk_k))$,需要填充的个数 $c=f_{(j-1)\alpha}-f_w$.

若$|S|<c$,在 S 中添加 $\boldsymbol{BF}$ 使得$|S|=c$,并构造对应计数器 $ctr=n$.

从 S 中随机选择 c 个 $\boldsymbol{BF}$,即 $\boldsymbol{BF}_{s1},\boldsymbol{BF}_{s2},\cdots,\boldsymbol{BF}_{sc}$,将每个 $\boldsymbol{BF}$ 的对应位置置 1.

对于 $i=1,2,\cdots,c$,更新计数器 $ctr[\boldsymbol{BF}_{si}]\leftarrow ctr[\boldsymbol{BF}_{si}]-1$;如果 $ctr[\boldsymbol{BF}_{si}]=0$,将 $\boldsymbol{BF}_{si}$ 从 S 移至 V.

(ii) 若$|S|\neq 0$,对于每个 $\boldsymbol{BF}\in S$:

选择 $ctr[\boldsymbol{BF}]$个关键词 $\tilde{w}\in\{0,1\}^*\backslash W$.

计算$(y_1=f(\tilde{w},sk_1),y_2=f(\tilde{w},sk_2),\cdots,y_k=(\tilde{w},sk_k))$,将其在 $\boldsymbol{BF}$ 中的对应位置置 1.

将 $\boldsymbol{BF}$ 从 S 移至 V.

(iii) 令 $l\leftarrow|V|$.

3) 构建倒排索引 $\boldsymbol{I}_D$

将生成的 $d+l$ 个 $\boldsymbol{BF}$ 重新排序,记作 $\boldsymbol{I}'_1,\boldsymbol{I}'_2,\cdots,\boldsymbol{I}'_{d+l}$.记录填充的 $\boldsymbol{BF}$ 对索引的索引编号 $FList=\{id_{f1},id_{f2},\cdots,id_{fl}\}$.从每一个 $\boldsymbol{I}'_i$ 中选择各自的第 v 比特 $b_{i,v}$构造 $\boldsymbol{I}_v=[b_{1,v},b_{2,v},\cdots,b_{d+l,v}]$,$v\in[1,m]$.最终 $\boldsymbol{I}_D=[\boldsymbol{I}_1,\boldsymbol{I}_2,\cdots,\boldsymbol{I}_m]^{\mathrm{T}}$.

SE-2 与 SE-1 一样满足 IND-CKA 安全.由构造可知,SE-2 满足(α,δ)-安全,因此能够抵抗统计分析攻击.相比于 SE-1,SE-2 有效地减少了索引的存储空间.

5 实验结果与性能分析

实验选取了来自新浪网的公开新闻数据,其中包括了 100 万条新闻的文档数据.实验在有 1 个 Master Node 和 5 个 Slave Nodes 的 Hadoop Hbase 实验集群上进行.

5.1 统计分析攻击效果实验

从数据集中抽取了 10 万条新闻记录作为文档进行实验.将关键词按照出现频次进行排序,用对应的序号表示关键词.10 万个文档包含了 74 702 个不同的关键词,词频最高的关键词为出现 4 184 次的“中国”,词频最低的关键词为出现 1 次的“生产国”等共 27 987 个不同的关键词.图 2(a)为文档中提取的关键词频率曲线:横轴是关键词频排名,纵轴表示对应关键词出现的频次,这个信息也是攻击者所能够知道的背景知识.可以看出,高词频部分的关键词比较少,并且相邻的两个关键词之间频次的差距较大.

低词频部分的关键词数目较多,而且相邻的关键词之间频次差距不大且相等.

实验对比了 Z-IDX 方案和 SE-1,SE-2 方案,分别遍历了所有的关键词,记录不同方案的返回集合表现出的词频分布特征.图 2(b)为 Z-IDX 方案的关键词词频统计信息,可以看出,该曲线和图 2(a)中的曲线几乎完全重合.这意味着,对于高频部分的词攻击几乎能以 100%的概率推测;对于低频部分的关键词,猜测的准的概率仍然较低.图 2(c)为 SE-1 方案的关键词词频统计信息,可以看出,每个关键词出现的频次很接近,都在 4 200 左右,此时统计分析攻击的效果和随机猜测效果一样.图 2(d)为 SE-2 方案的关键词词频统计信息,实验选择 $\alpha=1\,000$,可以看出低频词部分差异较小,前 10 000 个高频词出现频率几乎相同.此时相比 SE-1 方案,节省了近 4/5 的索引存储空间.

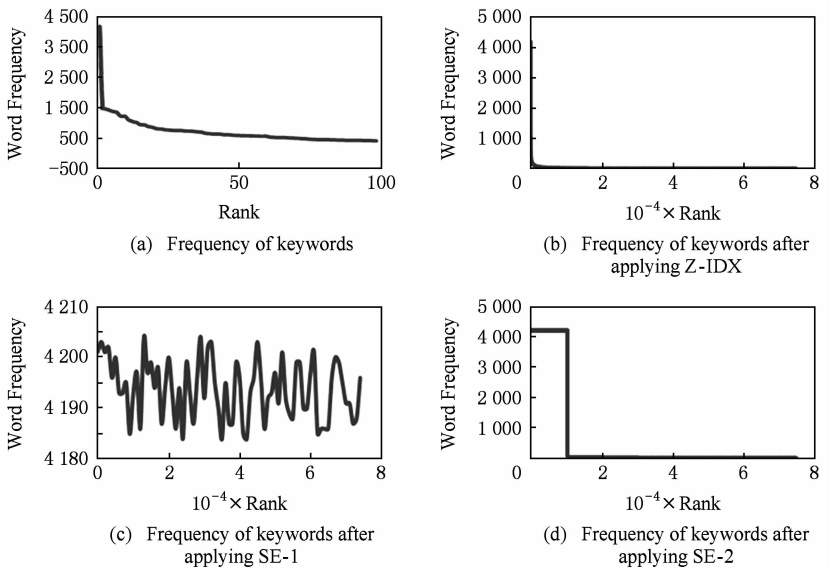


Fig. 2　Querying results

图 2　检索结果

5.2　性能实验

性能实验是在大小为 20 万、70 万和 100 万的 3 个不同的新闻文档的数据集上分别进行的.针对 3 个不同的数据集,分别建立 Z-IDX 和 SE-1,选择相同的关键词,在 2 个索引中分别检索,测试检索用时.文档大小对查询的影响,如图 3 所示.随着文档集的增加,Z-IDX 和 SE-1 索引检索用时都会增长.

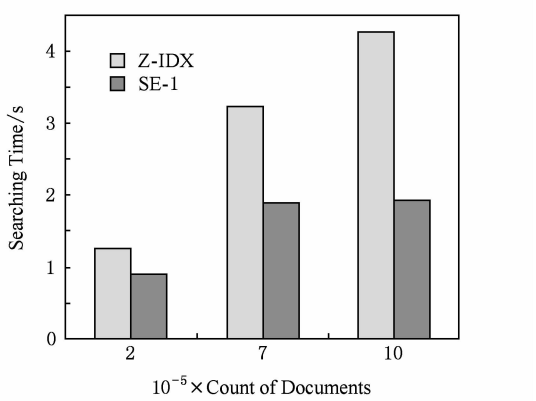


Fig. 3　Query performance

图 3　查询性能

从图 3 中可以看出,随着文档集大小的增加,使用 Z-IDX 检索用时的增加比使用 SE-1 索引检索用时的增加要快.由此可以推断,SE-1 索引在文档集较大时,有更好的检索优势,也更加适合处理海量数据.

6　总　　结

用户外包数据往往包含了敏感信息,为了保护数据,通常会将数据加密之后再进行外包.为了实现对外包加密数据的检索,需要构造安全的密文索引.本文分析该场景下索引的需求,提出了一种基于 Bloom 过滤器的安全索引:该索引利用 Bloom 过滤器减少索引空间占用;结合改进的倒排技术,该索引能够提供较好的检索效率;此外,利用索引的填充和过滤技术,在保证常数级检索效率的同时确保了索引能够抵抗统计分析攻击.

在面对分布式存储的数据时,文中的算法可以扩展到利用 MapReduce 实现,从而进一步提高索引的建立和检索效率.这些是今后需要考虑的工作.

参 考 文 献

[1] Li Hui, Sun Wenhai, Li Fenghua, et al. Secure and privacy-preserving data storage service in public cloud [J]. Journal of Computer Research and Development, 2014, 51(7): 1397-1409 (in Chinese)
(李晖, 孙文海, 李凤华, 等. 公共云存储服务数据安全及隐私保护技术综述[J]. 计算机研究与发展, 2014, 51(7): 1397-1409)

[2] Goh E J. Secure indexes [EB/OL]. IACR Cryptology ePrint Archive, 2003 [2012-08-15]. http://eprint.iacr.org/2003/216

[3] Curtmola R, Garay J, Kamara S, et al. Searchable symmetric encryption: Improved definitions and efficient constructions [J]. Journal of Computer Security, 2011, 19(5): 895-934

[4] Chang Y C, Michael M. Privacy preserving keyword searches on remote encrypted data [C] //Proc of the 3rd Int Conf on Applied Cryptography and Network Security. Berlin: Springer, 2005: 442-455

[5] Kurosawa K, Yasuhiro O. UC-secure searchable symmetric encryption [C] //Proc of the 16th Int Conf on Financial Cryptography and Data Security. Berlin: Springer, 2012: 285-298

[6] Kamara S, Papamanthou C, Roeder T. Dynamic searchable symmetric encryption [C] //Proc of the 19th ACM Conf on Computer and Communications Security. New York: ACM, 2012: 965-976

[7] Cash D, Jaeger J, Jarecki S, et al. Dynamic searchable encryption in very large databases: Data structures and implementation [C] //Proc of NDSS'2014. Reston, Virginia: The Internet Society, 2014 [2015-10-01]. http://www.internetsociety.org/access-pattern-disclosure-searchable-encryption-ramification-attack-and-mitigation

[8] Islam M S, Kuzu M, Kantarcioglu M. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation [C] //Proc of NDSS'2012. Reston, Virginia: The Internet Society, 2012 [2015-10-01]. http://www.internetsociety.org/doc/dynamic-searchable-encryption-very-large-databases-data-structures-and-implementation

[9] Li Zhen, Jiang Han, Zhao Minghao. A discretionary searchable encryption scheme in multi-user settings [J]. Journal of Computer Research and Development, 2015, 52(10): 2313-2322 (in Chinese)
(李真, 蒋瀚, 赵明昊. 一个自主授权的多用户可搜索加密方案[J]. 计算机研究与发展, 2015, 52(10): 2313-2322)

[10] Zobel J, Moffat A. Inverted files for test search engines [J]. ACM Computing Surveys, 2006, 28(2): No.6

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

[12] Goldreich O, Ostrovsky R. Software protection and simulation on oblivious RAMs [J]. Journal of the ACM, 1996, 43(3): 431-473

[13] Ostrovsky R. Efficient computation on oblivious RAMs [C] //Proc of the 22nd Annual ACM Symp on Theory of Computing. New York: ACM, 1990: 514-523

[14] Song D X, Wagner D, Perrig A. Practical techniques for searches on encrypted data [C] //Proc of IEEE S&P'2000. Piscataway, NJ: IEEE, 2000: 44-55

[15] Boneh D, Di Crescenzo G, Ostrovsky G, et al. Public key encryption with keyword search [C] //Proc of EUROCRYPT'2004. Berlin: Springer, 2004: 506-522

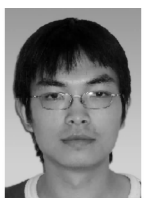
[16] Damiani E, Vimercati S D C D, Jajodia S, et al. Balancing confidentiality and efficiency in untrusted relational DBMSs [C] //Proc of the 10th ACM Conf on Computer and Communications Security. New York: ACM, 2003: 93-102

[17] Zipf G K. Human behavior and the principle of least effort [J]. Journal of Clinical Psychology, 1949, 6(3): 306-306

Hui Zhen, born in 1987. PhD candidate. His main research interests include accesss control and data protection.

Feng Dengguo, born in 1965. Professor and PhD supervisor. His main research interests include cryptography and information security.

Zhang Min, born in 1975. PhD and senior engineer. Her main research interests include theories and techniques about trusted computing and database security.



Hong Cheng, born in 1985. PhD and associate professor. His main research interests include outsourcing database security.