第 42 卷 第 6A 期
2015 年 6 月
计 算 机 科 学
Computer Science
Vol. 42 No. 6A
June 2015

基于同态加密的密文全文检索技术的研究

程 帅 姚寒冰
(武汉理工大学计算机科学与技术学院 武汉 430063)

摘 要 随着社会信息化的高速发展,信息资源日益膨胀。全文检索技术为信息资源的检索利用提供了一种高效的手段,然而随之而来的信息安全问题也日益凸显。针对此问题,给出了一种改进的同态加密算法,将其称为 New Homomorphic Encryption 算法,并设计了一种基于 NHE 算法的密文全文检索方案。该方案将同态加密算法的特性运用在检索过程中,采用倒排索引结构并使用二分查找法进行检索,能有效解决密文的全文检索问题。

关键词 信息安全,密文全文检索,同态加密
中图法分类号 TP309 文献标识码 A

Study of Cipher Text Retrieval Based on Homomorphic Encryption

CHENG Shuai YAO Han-bing
(School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430063, China)

Abstract Along with the rapid development of the information society, the information resource is increased day by day. The full-text retrieval technique provides a high-efficient means for searching of information resource, but more problems occur to the information security. With respect to the aforesaid problems, we provided a scheme of the improved homomorphic encryption algorithm, called new homomorphic encryption algorithm, and designed a cipher text retrieval scheme based on NHE algorithm. The scheme uses the properties of homomorphic encryption algorithm in the retrieval process, and the inverted index structure and binary search method is used to search. And it can effectively solve the problem of cipher text retrieval.

Keywords Information security, Cipher text retrieval, Homomorphic encryption

1 引言

随着时代的发展和科技的进步,文献信息资源越来越多,呈指数级上涨趋势,但面临的问题是如何充分利用和管理这些文献信息资源。全文检索技术的发展给对文献信息资源的利用和管理带来了便利。目前,全文检索技术的应用范围越来越广泛,但是全文检索技术在安全性能方面存在一定的缺陷,为了保证信息的安全性,都需要对信息进行加密存储,但是加密后的信息给信息的检索带来了一定的难度。在信息检索领域,密文全文检索方面的研究还比较少。1978 年,Rivest 和 Dertouzos[1] 提出同态加密,同态加密技术允许人们对密文进行特定形式的代数运算,得到的仍是加密的结果,不需要先对密文进行解密,这与对明文进行运算再将结果加密一样。也就是说同态加密可以实现人们在加密数据中进行诸如检索、比较等操作,得出正确的结果。本文将同态加密的这个特性应用到检索过程中,提出一种基于同态加密的密文全文检索方案。

2 全同态加密概述

Graig Gentry 构造的同态加密方案包括 4 个算法[2],即密钥生成算法、加密算法、解密算法和额外的评估算法。

全同态加密包括两种基本的同态类型,即乘法同态和加法同态,加密算法分别对乘法和加法具备同态特性,而 2009 年之前的同态加密算法仅支持加法同态或者乘法同态[3]。

2.1 全同态加密原理

全同态加密的原理:加密操作为 E,明文为 m,密文为 e。即

$$e=E(m),m=E'(e)$$

如果针对明文有操作 f,针对 E 可构造 F,使得

$$F(e)=E(f(m))$$

这样 E 就是一个针对 f 的同态加密算法。若对于任意复杂的明文操作 f,都能构造出相应的 F,则称 E 为全同态加密算法[4]。全同态加密的目的在于找到一种能在加密的数据上进行任意数量的加法和乘法运算的加密算法,使得对加密数据进行某种操作所得到的结果恰好等于加密前的数据进行预期操作再加密后得到的密文。

2.2 全同态加密算法的介绍

2.2.1 DGHV 方案的加解密算法

DGHV 方案只使用基本的模块化算术,而不是在多项式环中使用理想格的概念,其具体过程如下:

KeyGen:选用一个随机的正素数作为密钥 p,其中 p 的长度接近 512bit。

本文受武汉理工大学自主创新研究基金(2013-IV-050)资助。

程 帅(1988—),男,硕士生,主要研究领域为网络与信息安全,E-mail:747126482@qq.com;姚寒冰(1976—),男,博士,副教授,主要研究领域为信息检索、信息安全。

Encrypt(m)：选用两个随机数 q 和 r，其中 q 是一个大的正整数，$|2r| < p/2$，q 的长度接近 512bit，要加密一比特明文 $m \in \{0,1\}$，计算 $c = pq + 2r + m$，即得到相应的密文。

Decrypt(c)：$(c \bmod p) \bmod 2 = (c - p * \lceil c/p \rfloor) \bmod 2 = \mathrm{LSB}(c)\ \mathrm{XOR}\ \mathrm{LSB}(\lceil c/p \rfloor)$。

下面，我们重点分析其检索过程。

我们将检索算法记为 $Retrieval(c)$，使用 DGHV 算法对关键字 M_{index} 的检索过程如下：

a. 用户用同态加密算法 $c_{index} = (m_i + 2r_i + p \times q_i)$ 将 m_i 进行加密，得到密文 c_{index}，其中，对于同一个用户来说，p 是固定的，r_i 和 q_i 是随机数，每次都发生变化；

b. 服务器接收到检索词密文 c_{index}，然后从索引文件中读取密文索引词 c_i；

c. 服务器使用检索算法 $Retrieval = ((c_i - c_{index}) \bmod p) \bmod 2$，若 $Retrieval$ 为 0，则检索到 c_i 为关键字的密文。

可以发现，在检索算法 $Retrieval$ 中，用户想要让服务器端检索一个关键字就必须将密钥 p 发送给服务器。显然，这会使得用户存储在服务器上的加密数据完全暴露给服务器，密文对于服务器而言是完全透明的，因为服务器一旦获得了密钥 p，就可以直接对密文信息进行解密操作。因此，在服务器不可信的条件下，该算法不具有实用性[5]。

2.2.2 CAFED 方案的加解密算法

CAFED 方案描述了一种“完全同态”的保护私有数据的加密方案，将数据访问和数据处理分离，其具体实现过程如下：

首先选取一个安全参数 λ，设 $N = \lambda$，$P = \lambda^2$，$Q = \lambda^5$。

Keygen(λ)：选用一个 P 位的随机素数作为密钥 p。

Encrypt(p,m)：选取一个 Q 位的随机数 q，要加密一比特明文 $m \in \{0,1\}$，设 m' 是一个 N 位的随机数，$m' = m \bmod 2$，计算 $c = m' + pq$。

Decrypt(p,c)：计算 $(c \bmod p) \bmod 2$。

下面我们重点分析其检索过程。

CAFED 算法对关键字 M_{index} 的检索过程如下：

a. 用户用同态加密算法 $c_{index} = (m_i + p \times q_i)$ 将 m_i 进行加密，得到密文 c_{index}，其中，对于同一个用户来说，p 是固定的，q_i 是随机数，每次都发生变化；

b. 服务器端接收到检索词密文信息 c_{index}，然后从索引文件中读取密文索引词 c_i；

c. 服务器使用检索算法 $Retrieval = (c_i - c_{index}) \bmod p$，若 $Retrieval$ 为 0，则检索到 c_i 为关键字的密文。

可以发现，与 DGHV 方案一样，在检索算法 $Retrieval$ 中，用户想要让服务器端检索一个关键字，就必须将密钥 p 发送给服务器。因此，在服务器不可信的条件下，该算法不具有实用性[6]。

3 改进的同态加密算法

3.1 算法介绍

本文在原始 DGHV 算法和 CAFED 算法[7]的基础上提出一种新的同态加密算法 New Homomorphic Encryption，为了便于比较和使用，我们将其简称为 NHE 算法，算法如下。

KeyGen：选用一个 P 位的随机产生的安全正素数作为密钥 p。

Encrypt(m)：选取一个 Q 位的固定的安全大素数 q，其中 $P > Q$，随机产生一个 R 位的随机数 r，要加密一比特明文 $m \in \{0,1\}$，计算 $c = m + p + rpq$，即可得到相应的密文。

Decrypt(c)：计算 $c \bmod p$。

(1)加密过程

首先将明文进行比特分组（分组长度可以根据安全需求来确定），然后对每个明文分组 m_i 做加密运算，具体的过程分为如下几步：

a. 选取随机产生 P 位的安全大素数 p，选取 Q 位固定的安全大素数 q（$P > Q >$ 明文分组长度）；

b. 把消息 M 划分成长度为 L 的明文分组 $M = m_1 m_2 m_3 \cdots m_t$，其中 L 应该小于 Q；

c. 生成一个 R 位的随机数 r；

d. 使用加密算法 $c_i = m_i + p + rpq$，计算出密文 $C = c_1 c_2 c_3 \cdots c_t$；

e. 将密文消息 C 和安全大素数 q 发送给服务器。

(2)解密过程

a. 用户收到密文 C 后，对密文消息 C 进行分组，得到 $C = c_1 c_2 c_3 \cdots c_t$；

b. 使用密钥 p 和解密算法 $m_i = c_i \bmod p$，计算 m_i；

c. 得到明文消息 $M = m_1 m_2 m_3 \cdots m_t$。

(3)检索过程

NHE 算法对关键字 M_{index} 的检索过程如下：

a. 用户用同态加密算法 $c_i = m_i + p + r_i pq$ 将 M_{index} 进行加密，得到密文 c_{index}，其中安全大素数 q 是一个固定的大素数，对于云服务器来说 q 是已知的；

b. 服务器接收到检索词密文信息 c_{index}，然后从存储密文的文件中读取密文信息 $C = c_1 c_2 c_3 \cdots c_t$；

c. 服务器使用检索算法 $Retrieval = (c_i - c_{index}) \bmod q$，若 $Retrieval$ 为 0，则检索到 c_i 为关键字的密文。

使用此算法计算 $Retrieval$ 时，只需要将大素数 q 上传到云存储服务器，避免了将密钥 p 发送给服务器的现象。对于密文 $c_i = m_i + p + r_i pq$ 而言，即使服务器在拥有 q 后，使用 q 对明文进行解密，也只能得到 $c_i \bmod q = (m_i + p) \bmod q$，明文依然没有泄漏，因此服务器在检索的过程中，并不会知道用户的任何数据信息。在这种情况下，服务器能够直接对用户的密文进行检索等操作，用户的数据内容对于服务器来说，完全不可知。这样既保证了用户数据的安全性，又实现了直接对密文信息进行检索的功能，所以这个改进算法在服务器不可信的情况下具有一定的实用价值。

3.2 同态性验证

(1)同态加法特性验证

假设有两组明文 m_1 和 m_2，对它们分别进行加密可得密文

$$c_1 = m_1 + p + r_1 pq$$

$$c_2 = m_2 + p + r_2 pq$$

则对于明文

$$m_3 = m_1 + m_2$$

应有

$$c_3 = c_1 + c_2 = (m_1 + m_2) + 2p + pq(r_1 + r_2)$$

则

$$m_3 = c_3 \bmod p = (c_1 + c_2) \bmod p = m_1 + m_2$$

即该算法满足加法同态条件。

(2) 同态乘法特性验证

对于明文

$$m_4 = m_1 \times m_2$$

应有

$$c_4 = c_1 \times c_2 = (m_1 + p + r_1 pq) \times (m_2 + p + r_2 pq)$$
$$= m_1 m_2 + p(m_1 + m_2 + p + r_1 pq + r_2 pq) + pq(m_1 + m_2 + r_1 r_2)$$

则

$$m_4 = c_4 \bmod p = (c_1 \times c_2) \bmod p = m_1 \times m_2$$

即该算法满足乘法同态条件。

3.3 同态方案的安全性

方案所基于的困难问题是文献[8]中所提到的部分近似最大公因子问题(Partially Approximate Common Division Problem,PACDP)。定义如下:

定义1(部分近似最大公因子问题(PACDP)) 给定两个整数 $a_0 + x_0, b_0$,其中 a_0, b_0 都是大素数 p 的倍数,且 $|x_0| \ll p$。根据输入的两个大整数 $a_0 + x_0, b_0$,求其素数 p。

目前 PACDP 问题的解决方法主要是格归约求解。关于格归约求解,文献[9]等已对这种方法求解部分近似最大公因子进行各种参数设置和说明,根据文献中得出的结论可知使用格归约法求解 PACDP 问题是困难的。

本节将把 NHE 算法方案语义安全性规约到部分近似最大公因子问题。为此得到定理1,由于该定理的证明和文献[2]中的定理2证明过程类似,此处只对该定理的证明过程做简要的描述(详见文献[2])。

定理1 假设攻击者可以用一个优势为 ε 的算法 A 攻破 NHE 算法,则存在一个算法 B,通过调用 A,能够解决部分近似最大公因子问题,其优势为 $\varepsilon/2$。

证明 一个挑战者 C 产生一个部分近似最大公因子问题实例,即选取一个 η 比特大素数 p 和 θ 比特的大素数 q,随机选取整数 $r \in (-2^\rho, 2^\rho)$,计算 $N = pqr$,并将 N 发送给攻击者 B,B 使用 N 作为公钥。B 首先访问加密算法,产生许多0的密文,记为 $z_1, z_2, \cdots, z_n$,则存在某两个数 r_i, k_i 使得 $z_i = 2r_i + k_i p$。从这些0的密文中随机选择两个数 z_1^*, z_2^*,调用算法 Binary GCD。最后将输出这样一个元素 $z = p + r$,然后用 $z_1 = z_1^*$ 和 $z_2 = z$ 调用算法 Binary GCD,便可以输出 $q_p(z_1^*)$ 的二进制比特。最后计算 $p = \lceil z_1^* / q_p(z_1^*) \rfloor$,便可以恢复 p。类似文献[3],得到 B 的优势为 $\varepsilon/2$。

对方案的任何攻击方案都可以转换为部分近似最大公因子问题的解决方案,若攻击者可以破解 NHE 算法,则攻击者可以解决部分近似最大公因子问题,而上文已经说明了使用格归约求解部分近似最大公因子问题是困难的,所以本文方案是安全的。

另外,在该算法中,需要产生两个大素数 p 和 q,其中 p 和 q 的长度都是 $512bit$。由于密钥 p 的长度是512bit,因此密钥空间也是 2^{512},对于一个不知道密钥的攻击者而言,要想通过暴力破解得到密钥,他需要尝试 2^{512} 次,这种尝试在现实中是难以实现的。

4 一种改进的密文全文索引结构

4.1 传统的倒排索引结构及其安全隐患

如图1所示,传统的倒排索引结构由两种元素组成:索引文件(也称词汇表)和倒排文件(也称事件表)[10]。索引文件是文本中所包含的所有不同单词的集合,每条记录包含索引词及与该词有关的指针,这些指针即指向倒排文件。倒排文件存储了每一个单词在文本中出现的位置信息,根据这些位置信息又可以自动算出其频率信息。

图1 传统倒排索引模型

全文检索时,用查询词查找索引文件,再由索引文件指向倒排文件,倒排文件指向主文件。其安全隐患在于:

(1)索引文件的索引词表须保序以实现高效检索,这给攻击者提供了可乘之机;

(2)为了保证全文检索功能的全面性和扩展性,倒排文件保留了很多原文信息,一种极端的情况是仅根据索引文件就可以还原出完整的原始文档,这是攻击者所乐意看到的,必须采取强有力的措施加以防范[11]。

4.2 一种改进的安全倒排索引结构

本节将根据上文对密文索引在安全性方面的分析和研究,一步步提出对应的改进方案,最终形成一种改进的安全倒排索引结构。

首先,从全文索引倒排词表角度分析,其有序性是保证高效全文检索的前提,不能屏蔽。但是这种顺序容易遭受语义分析攻击。可以事先对文档进行中文分词并逐条加密分词结构,然后针对密文词条构建倒排索引,其保持的就是密文序,该密文序在语义上没有任何意义,这样就屏蔽了倒排词表明文语义上的顺序,这也是密文全文检索系统研究领域现阶段的做法。

其二,从全文索引静态存储角度分析,虽然上述索引结构中索引文件里存储的都是密文词条,但攻击者依然可以通过统计分析攻击获得关键信息。例如,倒排文件里的地址串越长,其对应的索引词就一定是该应用场景中的热点词。基于第一步的改进,可以进一步将索引文件中的密文词条数据与指向倒排文件的地址指针进行捆绑加密,并将索引文件与倒排文件分离存储,同时将倒排文件中的词条位置信息和频率信息在倒排文档地址指针集合中屏蔽掉,这样就可以完全防范统计攻击了。

其三,从全文索引动态访问角度分析,如果攻击者能动态跟踪全文检索系统的访问过程,则有可能找出一系列词条与该词条在索引中相关信息的对应关系从而破解部分关键信息。基于第二步的改进,可以进一步将索引文件中的倒排词表划分区间,一个区间里包含 M 个词条,然后再对词条区间进行整体加密,这样就将单个词条的安全性转化为该词条所在区间的整体安全性,在一定程度上防范了动态访问过程中的明密文比对攻击。

其四，经过上述改进后，为了进一步提高性能，基于第三步的改进，可以进一步引入两级索引结构。一级索引包含全部密文词条元组集合，每一个元组由密文词条和指向倒排地址文件的指针组成，然后再将这些元组按词条区间分块加密。取每一区间的首个词条再次建立索引即得二级索引，可在系统初始化时一次性导入内存以加速索引的访问，从而弥补索引加密带来的性能损失。二级索引的每个元组由密文词条和指向一级索引中该词条所处区间的首地址指针构成。二级索引采取整体加密的方式存储。经过最终改进后的索引结构如图 2 所示。

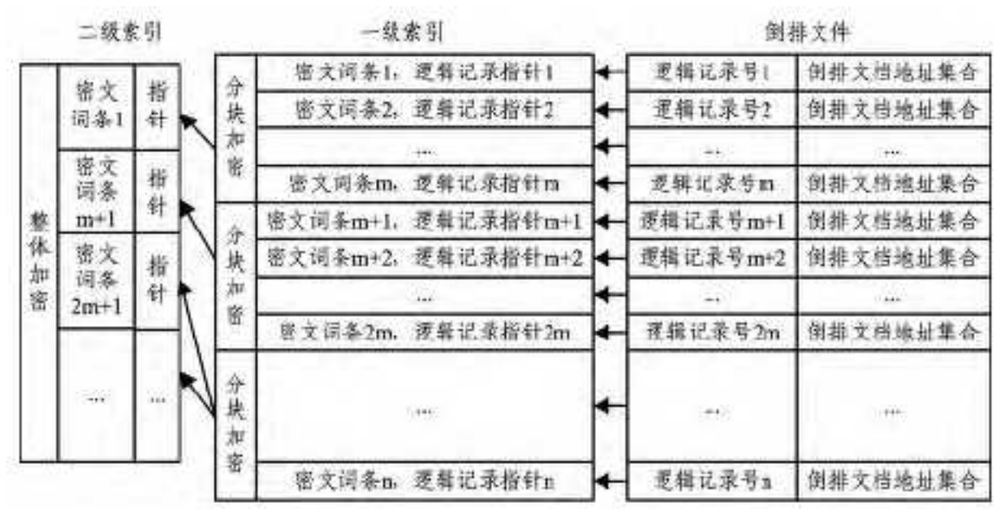

图 2　两级索引结构的倒排索引

5　密文全文检索算法

5.1　加密过程

首先对文档进行分词，建立明文索引，然后将索引中的词条分为若干长度为 L 的分组 $M=m_1m_2m_3\cdots m_l(L<P)$，最后按照前面所述的同态加密算法加密，$c_i=m_i+p+r_ipq$，将加密得到的分组密文依次合并，得到加密密文。

5.2　检索过程

检索时，需要提交检索需要的关键词密文 key，假设 $key=m_{index}+p+r_1pq$，m_{index} 是对应的明文关键词。将二级索引一次性调入内存，采用二分查找法对二级索引进行查找。假设每次对比的二级索引的密文是 $c_i=m_i+p+r_ipq$，计算 $key \bmod q$ 和 $c_i \bmod q$，将计算结果进行比对就能知道两者大小关系，从而进行二分查找。当找到 key 所在的二级索引的某个区间之后，通过此区间所对应的二级索引中的指针将一级索引块调入内存，一级索引的查找可以直接通过顺序查找，比对过程和二级索引的比对相同，当找到 $key \bmod q$ 和 $c_i \bmod q$ 相等时，检索成功，通过一级索引中的逻辑记录指针就可以找到倒排文件中倒排文档地址集合，通过该集合中的地址就能找到包含关键词的所有文件；若在二级索引的比对过程中没有找到 c_i 使 $key \bmod q$ 与 $c_i \bmod q$ 相等，则检索失败。

6　密文全文检索算法实验结果及分析

为了考察本文提出的检索算法效率，将其与文献[12]提出的检索算法做了对比实验。我们选取了一个有 17 万多个单词的词库（词库来自武汉华工达梦数据库有限公司开发的数据库管理系统）作为测试集，先后从中抽取 10 组不同数量的单词作为单词集，分别做实验。

实验环境描述如下：

(1) 操作系统：Win7 旗舰版操作系统。

(2) 硬件平台：Intel Core i7 处理器（主频 2.10GHz，二级缓存 6MB）、内存 4GB，实验结果如表 1 所列。

表 1　检索算法查询时间对比

单词数量 n(个)	文献[12]算法		本文算法		后者占前者的百分比
	总时间（毫秒）	平均每个单词耗时（毫秒）	总时间（毫秒）	平均每个单词耗时（毫秒）	
10	0.005	0.0005000	0.006	0.000600	120%
50	0.031	0.000620	0.027	0.000540	87.097%
100	0.072	0.000720	0.059	0.000590	81.944%
500	0.519	0.001038	0.353	0.000706	68.015%
1000	1.180	0.001180	0.742	0.000742	62.881%
5000	7.817	0.001563	4.793	0.000959	61.315%
10000	18.415	0.001842	10.686	0.001069	58.029%
50000	104.743	0.002095	59.024	0.001180	56.351%
100000	207.199	0.002072	118.153	0.001182	57.024%
170000	378.919	0.002229	211.452	0.001244	55.804%

由表 1 数据可以看出，当数据量很小时，两者所用时间相差不大，甚至在第一组实验中文献[12]的算法所用时间更少，但是当数据量逐渐增大时，本文使用的检索算法效率远远高于文献[12]中使用的检索算法。由此可知，当文档数量非常大的时候，文献[12]提出的方案效率将会非常低，相比之下，本方案的检索算法虽然要花费时间去建立索引，但是建立索引的优势就是在检索时能够快速找到用户需要的文档，因此本方案与文献[12]中的方案相比具有一定的优势。

结束语　本文将同态加密技术运用到密文全文检索中，设计了一个既能确保用户数据安全，又能避免传统加密算法需要解密才能进行检索的弊端，提高了密文的检索效率。本方案在加密过程中用到大数之间的操作，这在存储过程中会产生一定的开销，但是我们可以通过接下来的优化实验发现并设计出效率更高的检索算法，并加入访问控制机制，使数据存储更安全。

参 考 文 献

[1] Rivest R, Adleman L, Dertouzos M. On data banks and privacy homomorphisms[C] // Foundations of Secure Computation. 1978:169-180

[2] van Dijk, Gentry, Halevi, et al. Fully homomorphic encryption over the integers[J]. LNCS, 2010, 6110:24-43

[3] 何文才，杜敏，刘培鹤，等. 基于 Paillier 同态的无线自组网组密钥管理方案[J]. 计算机科学，2013，40(10)：114-118

[4] 黄永丰，张九岭，李星. 云存储应用中的加密存储及其检索技术[J]. 中心通讯技术，2010，16(4)：33-35

[5] 林如磊，王箭，杜贺. 整数上的全同态加密方案的改进[J]. 计算机应用研究，2013，30(5)：1515-1519

[6] Gentry. A Full Homomorphic Encryption Scheme[D]. Stanford: Stanford University, 2009

[7] Gentry. Fully homomorphic encryption using ideal lattices[M]. New York, Association for Computing Machinery, 2009:169-178

[8] Howgrave-Graham. Approximate integer common divisors[C] // Volume 2146 of Lecture Notes in Computer Science (CALC' 01). Springer, 2001:51-66

[9] Nick Howgrave-Graham. Approximate inter common divisors [C] // Silverman J H. CALC, LANCS, 2001, 2146:51-66

[10] 郭利刚，姚寒冰. 基于倒排索引的密文数据库检索方法研究[J]. 计算机安全，2010，09：13-15

[11] 宋赛. 密文全文检索系统的安全索引结构研究[D]. 武汉：华中科技大学，2009

[12] Li Jian, Chen Si-cong, Song Dan-jie. Security structure of cloud storage based on homomorphic encryption scheme[C] // Proceedings of IEEE CCIS 2012. 2012