云计算中外包数据的可搜索加密和搜索结果完整性检验方案

1901210394 郭雨洁
1901210600 赵泽涵
1901210447 刘嘉欣

一、背景介绍

作为外包计算的一个重要分支，外包数据库（Outsourced Database , ODB ）近年来越来越吸引学术界的广泛关注。早在 2002 年， Hakan Hacigum 此等人就隐含地引入了外包数据库的概念。在外包数据库模型中，为了节省昂贵的数据库管理成本，数据拥有者将数据库在本地进行加密运算并将密文数据库外包给云服务器来管理。云服务器提供数据库访问所需的一切软硬件资源，确保在收到用户数据访问请求后，执行数据库检索并返回相应的结果给用户。这种模式下，既降低了数据拥有者的数据库维护开销，又可以为用户提供高质量的数据访问服务。然而，外包数据库在为人们带来诸多益处的同时，也不可避免地面临着一些新的安全挑战上。

首先，在云计算环境下，找到一个完全可信的云服务器几乎是不可能的（可能由于经济效益的考量、技术人员的失误以及系统遭受攻击）：而外包数据往往包括一些不能泄露给云服务器的敏感信息。因此，一个主要安全挑战是外包数据的秘密性，即云服务器不能知道存储的数据内容。需要指出的是，传统加密技术不能本质上解决这个问题，这是因为对密文进行有意义的操作是非常困难或效率非常低下的。尽管传统数据加密技术能够保证用户外包数据的机密性,然而,数据加密后使得在海量的密文文件中搜索特定的文件变得极为困难。对于明文信息,我们可以采用传统的搜索技术提取我们想要的数据。但是对于密文数据,服务器无法执行基于密文的高效检索,只能将整个加密数据库返回给用户,由用户自行解密并查找想要的数据显然这种方法所需的存储和计算开销是用户无法承受的,更重要的是这与数据外包的初衷相违背。为了在保证数据机密性的同时实现高效的密文检索功能,可搜索加密技术(Searchable Encryption) 于 2000 年被提出,用户将自己的数据加密存储在云服务器中,同时把关键词提取出来并进行加密处理,随后生成基于密文关键词的数据索引文件;当需要搜索云端存储的密文数据时,发送一个关键词陷门信息给云服务器,由其在索引上执行检索,并返回对应的数据密文给用户,用户在本地完成解密操作并最终获得所要查询的数据文件。

其次，由于云服务器是不完全可信的，出于自身经济利益（节省网络带宽和计算量）的驱动或者软硬件运行故障，它可能会返回给用户一些不正确 / 不完整的检索结果。因此，另一个重要的安全挑战是检索结果的可验证性。也就是说，用户能够高效地对云服务器返回的检索结果进行审计。特别是在外包数据库场景下，检索结果的可验证性主要包含如下两方面内容：（1）正确性：检索结果是用户上传的原始数据而没有被篡改过；（2）完整性：检索结果包含全部符合查询条件的数据记录。

二、可搜索加密

针对检索密文的方案提出了可搜索加密方案, 可搜索加密技术分为公钥可搜索加密和对称可搜索加密两类,其本质区别在于采用何种加密体制(对称/公钥)对数据进行加密。对称可搜索加密技术(SearchableSynmetric Encryption, SSE)能够实现用户加密数据的高效检索。作为 SSE 的补充,公钥可搜索加密技术(Public Key Encryption with Keyword Search, PEKS) 2004 年由

DarBoneh 等人首次提出，其与 SSE 的本质区别在于用户私钥用于生成检索陷门而公镇用于加密数据。PEKS 适用于多对一的数据检索场景(如加密邮件系统),功能更为强大。然而,采用公钥加密体制会导致方案效率低下。随着云存储服务的普及,可搜索加密的效率问题日益突显。下面将主要关注对称可搜索加密技术。

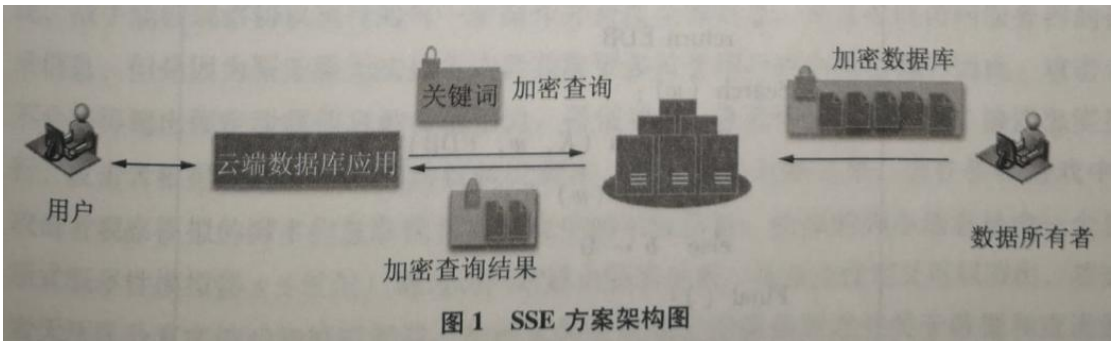


图 1　SSE 方案架构图

第一种对称可搜索加密的构造通常基于伪随机函数,具有计算开销小、算法简单、速度快的特点,除了加解密过程采用相同的密钥外,其陷门生成也需密钥的参与.单用户模型的单用户特点使得对称可搜索加密非常适用于该类问题的解决:用户使用密钥加密个人文件并上传至服务器,检索时,用户通过密钥生成待检索关键词陷门,服务器根据陷门执行检索过程后返回目标密文。

定义 1(对称可搜索加密). 定义在字典Δ={W1,W2,⋯,Wd}上的对称可搜索加密算法可描述为五元组: SSE=(KeyGen,Encrypt,Trapdoor,Search,Decrypt),

其中,

1) K=KeyGen(λ):输入安全参数λ,输出随机产生的密钥 K;

2) (I,C)=Encrypt(K,D), 输入对称密钥 K 和明文文件集 D=(D1,D2,⋯,Dn),Di∈$2^

Δ$,输出索引 I 和密文文件集 C=(C1,C2,⋯,Cn).对于无需构造索引的 SSE 方案(例如 SWP 方案),I=∅;

3) TW=Trapdoor(K,W):输入对称密钥 K 和关键词 W,输出关键词陷门 TW;

4) D(W)=Search(I,TW):输入索引 I 和陷门 TW,输出包含 W 的文件的标识符构成的集合 D(W);

5) Di=Decrypt(K,Ci):输入对称密钥 K 和密文文件 Ci,输出相应明文文件 Di.

如果对称可搜索加密方案 SSE 是正确的,那么对于∀λ∈ℕ,n∈Δ,D=(D1,D2,⋯,Dn)以及 KeyGen(K,D)输出的 K 和(I,C),都有 Search(I,Trapdoor(K,W))=D(W)和 Decrypt(K,Ci)=Di 成立.

这里,Ci∈C,i=1,2,⋯,n.

基于定义 1,对称可搜索加密流程如下:加密过程中,用户执行 KeyGen 算法生成对称密钥 K,使用 K 加密明文文件集 D,并将加密结果上传至服务器.检索过程中,用户执行 Trapdoor 算法,生成待查询关键词 W 的陷门 TW;服务器使用 TW 检索到文件标识符集合 D(W),并根据 D(W)中文件标识符提取密文文件以返回用户;用户最终使用 K 解密所有返回文件,得到目标文件.

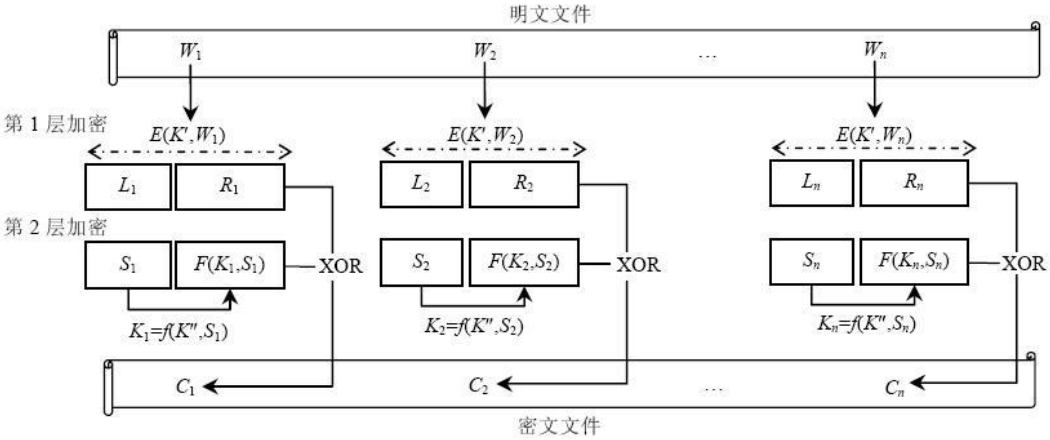
SWP 方案:

SWP 方案在预处理过程中根据文件长度产生伪随机流 S1,S2,⋯,Sn(n 为待加密文件中“单词”个数),然后采用两个层次加密:在第 1 层,使用分组密码 E 逐个加密明文文件单词;在第 2

层,对分组密码输出 E(K',Wi)进行处理:① 将密文等分为 Li 和 Ri 两部分;② 基于 Li 生成二进制字符串 SiIIF(Ki,Si),这里,Ki=f(K'',Li),II为符号串连接,F 和 f 为伪随机函数;③ 异或 E(K',Wi)和 SiIIF(Ki,Si)以形成 Wi 的密文单词.

查询文件 D 中是否包含关键词 W,只需发送陷门 TW=(E(K',W),K=f(K'',L))至服务器(L 为 E(K',W)的左部),服务器顺序遍历密文文件的所有单词 C,计算 C XOR E(K',W)=SIIT,判断 F(K,S)是否等于 T:如果相等,C 即为 W 在 D 中的密文;否则,继续计算下一个密文单词.

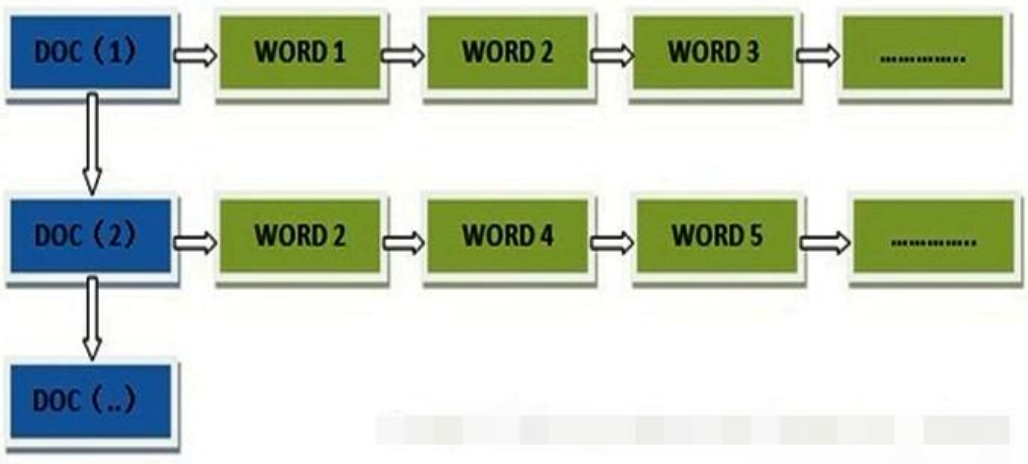
SWP 方案通过植入“单词”位置信息,能够支持受控检索(检索关键词的同时,识别其在文件中出现的位置).例如,将所有“单词”以 WIIW 在文件中出现的位置,仍按图所示加密,但查询时可增加对关键词出现位置的约束.SWP 方案存在一些缺陷:① 效率较低,单个单词的查询需要扫描整个文件,占用大量服务器计算资源;② 在安全性方面存在统计攻击的威胁.例如,攻击者可通过统计关键词在文件中出现的次数来猜测该关键词是否为某些常用词汇。



另一种可搜索对称加密使用了倒排索引。在搜索引擎中每个文件都对应一个文件 ID,文件内容被表示为一系列关键词的集合。例如:“文档 1”经过分词,提取了 20 个关键词,每个关键词都会记录它在文档中出现的次数和出现位置。得到正向索引的结构如下:

“文档 1”的 ID > 单词 1:出现次数,出现位置列表;单词 2:出现次数,出现位置列表;……。

“文档 2”的 ID > 此文档出现的关键词列表。一般是通过 key,去找 value。



当用户在主页上搜索关键词“华为手机”时,假设只存在正向索引(forward index),那么就需要扫描索引库中的所有文档,找出所有包含关键词“华为手机”的文档,再根据打

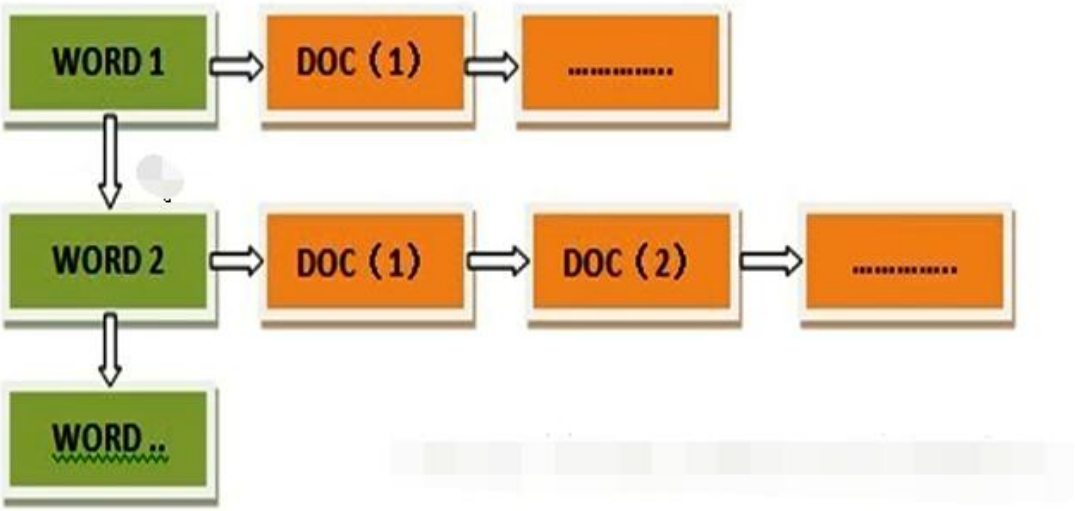
分模型进行打分，排出名次后呈现给用户。因为互联网上收录在搜索引擎中的文档的数目是个天文数字，这样的索引结构根本无法满足实时返回排名结果的要求。

所以，搜索引擎会将正向索引重新构建为倒排索引，即把文件 ID 对应到关键词的映射转换为关键词到文件 ID 的映射，每个关键词都对应着一系列的文件，这些文件中都出现这个关键词。

得到倒排索引的结构如下：

“关键词 1”：“文档 1”的 ID，“文档 2”的 ID，……。

“关键词 2”：带有此关键词的文档 ID 列表。



在 Curtmola-Garay-Kamara-Ostrovsky-ccs06 中提出的可搜索对称加密中，就用到了倒序索引。首先为每一个关键字 Wi 建立一个链接列表，其中列表的元素是包含 Wi 的所有文档的标识符。之后为整个数据集建立一个数组 A 和一个查找表 T，然后将所有链接列表中的元素加密并放入数组 A 中。其中有两个参量 P 是 PRP，F 是 PRF，PRP 用来作为地址，PRF 即伪随机数生成函数，作为 Key。查询的时候就通过 P 和 F 来定位一个元素。

三、搜索结果完整性检验方案

针对结果完整性问题也提出了一些解决方案。现有的外包数据库检索方案根据验证方法的不同可以分为三类：

第一类方法是利用认证数据结构（如 Merkle Hash Tree,MHT）来验证检索结果的完整性：其主要思想是将数据库的所有数据记录作为叶子节点，创建一棵全局的 MHT ，根节点经用户签名存储在服务器上。想要验证某条数据记录时，用户通过重新计算 MHT 根节点的签名来完成。然而，基于 MHT 的方法的缺点在于验证过程需要较大的通信和计算开销。也就是说,为了完成单个数据记录的验证,服务器需要返回根节点到当前(要验证)节点路径上的所有节点的兄弟节点的哈希值给用户(logn 个哈希值)。

第二类方法是概率性完整性验证方案，主要技巧在于数据拥有者事先将少量“间谍”数据记录插入到数据库里,然后通过分析检索结果中“间谍”数据来完成验证。如果满足某个查询条件的“间谍”数据没有返回,则用户可以认定服务器存在作弊行为。这非常类似于 Ringers 的思想，但是,这种方法有两个缺点:首先,为了实现结果可验证性，“间谍”数据必须

共享给所有授权用户。因此,服务器通过与某个授权用户勾结就可以获得所有“间谍”数据,从而在之后的检索中只要将所有要验证的数据记录返回,就能够轻松地达到欺骗用户的目的。其次,这种方法需要服务器返回整条数据记录,因此不支持投影查询等传统的数据库查询方式。

第三类方法是基于签名链技术的验证方案。与基于 MHT 的方法相比,该类方法降低了检索验证过程的通信和计算开销。有人提出了一个聚合签名的变型（Immutable Aggregated Signature),不仅能够节省通信和验证开销,而且使得敌手在即使已经获得了一定数量签名时仍然不能计算出新的可用的聚合签名。

常规的外包数据库验证是用户在将数据存入数据库之前对每一个数据进行签名,而服务器会存储数据以及其对应的每一个签名。在用户进行查询时,服务器将对应的数据和签名一并返回给用户,使得用户可以在本地有效检验数据的完整性。这个方法是可用的,但是它的缺点在于,对用户而言,无论是接收全部的数据和对应的签名,还是计算数据和验证签名,都开销极大,效率极低。因此,需要在数字签名的基础上用到改进的聚合签名的验证方案。在收到用户的查询请求后,服务器执行该查询,得到满足条件的一系列数据,但它不马上将这些数据和对应的签名全部返回给用户,而是将这些独立的签名聚合成一个签名,再将数据的组合和这个聚合的签名返回给用户。通过这个歌方法,用户可以十分便捷的检验数据的完整性。

聚合签名是一种用来将任意多个签名聚合成一个签名的变体签名方案,假定系统中有 n 个用户{u1,u2,...,un},对应得,存在 n 个公钥{pk1,pk2,...pkn},n 个消息{m1,m2,...mn},以及 n 个对消息的签名{∂ 1,∂ 2,...∂ n},聚合签名的生成者(这里的聚合签名生成者可以是任意的, 不需要在集合{u1,u2,...,un}中）可以将{∂ 1,∂ 2,...∂ n}聚合成一个唯一的短签名∂ 。重要的是,产生的聚合签名是可验证的,即给定∂ ,参与生成∂ 的公钥集合{pk1,pk2,...pkn}及其签名的原始消息集合{m1,m2,...mn},可以验证得到用户 ui 分别对消息 mi 做了签名∂ i。下面对聚合签名的执行进行详细的介绍。

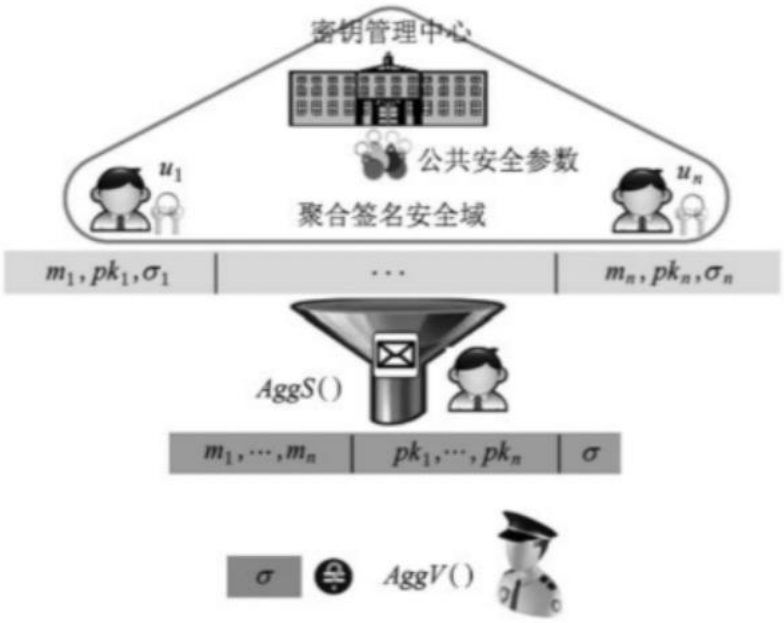
AS=(Gen,Sign, Venfy, AggS, Aggv)是多项式时间算法五元组,具体说明如下:

DS=(Gen, Sign,Verify)是普通的签名方案,亦称为聚合签名的基准签名。

聚合签名生成 AggS。在 Gen, Sign 的基础上,实现普通签名功能、消息向量(m1,...,mn),用户向量(u1,...,un)和个体签名向量(∂ 1,...,∂ n)的聚合功能以及聚合追加新的签名∂ n+1

聚合签名验证 AggV。假设每一个 ui 对应一个公私钥对{pki,ski},如果:AggV(pk1,...,pkn,m1,...,mn,AggS(pk1,...,pkn,m1,...,mn,Sign(sk1,m1),...,Sign(skn,mn)))=1,则输出 1,否则输出 0。

聚合签名还可以支持增量聚合,签名∂ 1,∂ 2 可以聚合成∂ 12,然后∂ 12 可以和∂ 3 继续聚合成∂ 123。



聚合签名的特点有：

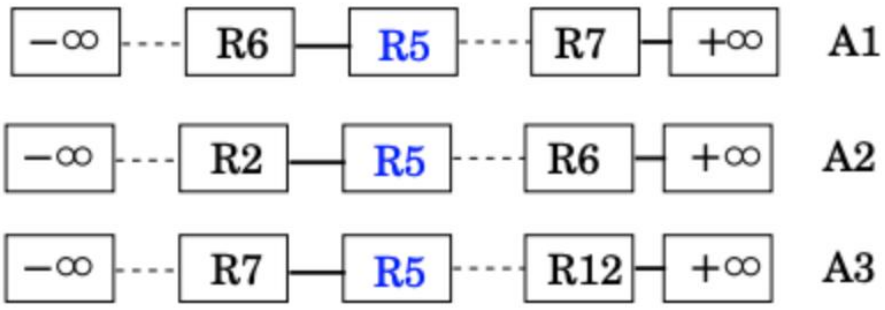
1.压缩性：聚合签名提供了一种将 n 个用户对 n 个消息的签名合成为一个单一的签名的方法，聚合签名可以同时给多个消息，多个用户提供不可否认服务，可以把任意多个用户的签名压缩成一个签名，这大大减少了签名的存储空间，同时也降低了传输签名的网络带宽的要求。

2.简便性：把任意多个签名的验证简化到一次验证，大大减少了签名验证的工作量，所以聚合签名在很大程度上提高了签名的验证与传输效率。

但是，聚合签名只保证了数据的完整性，那么要如何能够保证检索的完整性呢？这里就要用到签名链。签名的生成算法如下：

Sign(r) = h(h(r)||h(IPR1(r))||...h(IPRl(r)))SK

其中 h()是哈希函数，IPRi 表示前一个直接关联的元组数据，l 表示搜索的维度的编号，SK 则是签名的私钥。对于不同的维度而言，给定不同的属性，各元组数据根据该属性进行排序，从而数据在不同的维度有着不同的可搜索的排序序列。如图所示，有三个维度的可搜索排序序列，对于 R5 而言，其在三个维度的前一个元素分别为 R6，R2，R7，因此，R5 的签名为：



SIgn(R5) = h(h(R5)||h(R6)||h(R2)||h(R7))SK

基于如上的签名链，服务器在收到请求后，经过检索释放所有匹配的元组，两个超出查

询范围（以提供完整性证明）的边界元组以及对应于结果集的聚合签名 。 签名链向查询者证明服务器确实返回了查询范围内的所有元组。

Mykleun 等人首次提出了检索结果完整性验证问题。随后, Pang 等人分别提出了静态和动态数据库检索完整性验证方案。在他们的方案中,所有的数据记录被假定按照某个检索属性进行排序,然后,数据拥有者为每条记录创建一个包含其前后相邻数据记录信息的签名。为了遵循数据库访问控制策略,两个虚拟的边界记录被插入到可搜索属性域中。然而,这种方法很难处理检索区域不连续的情况。最近, Yuan 等人 3 提出一种新的外包数据库可验证聚合查询方案。在他们的方案中,每条数据记录被分配一个基于多项式的认证标签,该标签用于检查聚合查询结果的正确性。但是,上述方案均没有考虑当服务器有意返回空集的情形,从而均达不到完整性校验的完备性。

现有的验证方法都不能完全解决检索结果的可验证问题。尤其是在云服务器有意返回空集时,现有的方法均不能很好地解决检索结果的正确性和完整性验证问题。最大的难点在于用户无法有效地验证空集的有效性。当云服务器返回空集时,有两种可能:一种情况是云服务器是诚实的,确实没有符合条件的数据记录;另一种情况是云服务器根本没有执行查找运算。在对数据库内容不知情的情况下,用户无法区分这两种情况。因此,当检索结果为空集时,如何验证检索行为的正确性是一个非常具有挑战性且有价值的研究课题。在动态数据检索方面,研究支持前/后向安全的对称可搜索加密方案,设计支持数据安全更新的动态可搜索加密方案是未来重要的研究方向。

参考文献:

[1]. E. Bertino, B. Carminati, E. Ferrari,等. Selective and authentic third-party distribution of XML documents[J]. IEEE Transactions on Knowledge & Data Engineering, 16(10):1263-1278.

[2]. Devanbu P , Gertz M , Martel C , et al. Authentic Data Publication over the Internet[J]. Journal of Computer Security, 2002, 11(3):291-314.

[3]. H. Hacigumus, B. Iyer, S. Mehrotra. Providing database as a service[C]// Proceedings 18th International Conference on Data Engineering. IEEE, 2002.

[4]. Li, Jin, Chen, Xiaofeng, Li, Mingqiang,等. Secure Deduplication with Efficient and Reliable Convergent Key Management[J]. IEEE Transactions on Parallel & Distributed Systems, 25(6):1615-1625.

[5]. Pang H H, Tan K L. Authenticating query results in edge computing[C]// 2004.

[6]. Raluca Ada Popa, Frank H. Li, Nickolai Zeldovich. An Ideal-Security Protocol for Order-Preserving Encoding[C]// 2013 IEEE Symposium on Security and Privacy. IEEE, 2013.

[7]. Yan Zhu,, Gail-Joon Ahn,, Hongxin Hu,, 等. Dynamic Audit Services for Outsourced Storages in Clouds[J]. IEEE Transactions on Services Computing, 6(2):227-238.

[8]. SONG,D. X. Practical techniques for searches on encrypted data[C]// Proc. 2000 IEEE Symposium on Security and Privacy (SP'00). IEEE Computer Society, 2000.