软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, [doi: 10.13328/j.cnki.jos.000000]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

一种基于动态累加器的去中心化加密搜索方案

张 琰[1,2], 王瑾璠[1,2], 齐竹云[2,3], 杨镕玮[1,2], 汪 漪[1,2]

[1](南方科技大学未来网络研究院,广东 深圳 518055)

[2](鹏城实验室,广东 深圳 518055)

[3](北京大学电子与计算机工程学院,广东 深圳 518055)

通讯作者: , E-mail: wy@ieee.org

摘 要: 近年来区块链技术取得广泛关注，涌现出众多基于区块链技术的新型应用，其中以 StorJ，Filecoin 为代表的去中心化存储应用取得了较好的市场反响。对比传统中心化存储，去中心化存储为用户提供了全新的数据存储思路，令用户在获得更好的服务伸缩性的同时，有效降低数据存储的成本。但是我们发现在现有的去中心化存储方案中，用户的隐私不能得到有效的保护。本文介绍了一种利用加密搜索技术对去中心化存储方案进行加强的方法。新方法将动态累加器算法引入到到加密搜索过程中，保障用户存储内容隐私并提供了更好的加密搜索的性能。

关键词: 区块链；去中心化存储；加密搜索；动态累加器

A Decentralized Searchable Encryption Scheme based on Dynamic Accumulator

ZHANG Yan[1,2], WANG Jin-Fan[1,2], QI Zhu-Yun[2,3], YANG Rong-Wei[1,2], WANG Yi[1,2]

[1](SUSTech Institute of Future Networks, Southern University of Science and Technology, Shenzhen 518055, China)

[2](Pengcheng Laboratory, Shenzhen 518055, China)

[3](School of Electronic and Computer Engineering, Peking University, Shenzhen 518055, China,)

Abstract: Since the flourish of the blockchain technology, a series of applications based on blockchain technology are emerging, and the decentralized storage service becomes the killer app in the decentralized markets, such as Storj, Filecoin. Comparing with the centralized storage, decentralized storage gives people secure, low cost and more scalable. But, in existing decentralized storage apps, client's privacy cannot be protected. Therefore, we propose the idea that implementing the searchable encryption scheme with decentralized storage to improve the user's privacy. Also, we utilize the dynamic accumulator to improve the search efficience of the searchable encryption scheme.

Key words: Blockchain; Decentralized Storage; Searchable Encryption; Dynamic Accumulator

0 引言

区块链概念的被提出和首次实现来自于中本聪(Satoshi Nakamoto)于 2009 年 1 月 3 日所发表的比特币[1][2]。近年来，区块链技术取得持续发展，其中最具代表性的成果之一是由 Vitalik Buterin 在 2013 年提出的

以太坊（Ethereum）及太坊虚拟机（EVM），令区块链网络中的节点具备执行图灵完备代码的能力，从而使区块链网络成为去中心化应用（DApp）的部署平台。区块链技术当前仍处于探索及持续发展的状态，但基于区块链技术构建的新型应用已经开始部署并取得了一定的成功。以 StorJ[4,6],FileCoin[5]为代表的去中心化存储应用，是一类重要的区块链应用。

去中心化存储为用户提供了更灵活的扩展存储方式以及更低的存储成本，但用户存储隐私的相关问题也被逐渐暴露出来。加密搜索[7,8,9,10]技术是解决这一问题的关键方法，其基本原理是通过关键词搜索及对关键词信息、文件信息进行隐藏达到保护隐私的目的。在传统中心化存储方案中，加密搜索在保护文件安全及客户隐私方面已经获得了成功的运用。目前已有一些工作[11]尝试将加密搜索引入到去中心存储方案中，但我们认为加密搜索方法在去中心化运用中的使用仍具有较大的可优化空间。

本文阐述了一种基于动态累加器的去中心化加密搜索方案，该方案涉及区块链智能合约、加密搜索技术以及动态累加器技术。本文的主要贡献包括：

(1) **保护用户隐私的安全去中心化数据存储**：将加密搜索技术运用与去中心化存储中，证明了加密搜索技术在去中心化存储场景中运用的技术可行性，在保障存储数据安全的同时保护了用户隐私。

(2) **改善加密搜索效率**：引入动态累加器 [12, 13]到加密搜索方案中，将搜索令牌（Token）比对次数由 O(n)=m*n 次降低到 O(n)=m 次（m 为文件数，n 为每个文件拥有的平均搜索关键字数）。

本文结构如下：第 1 章，介绍区块链、加密搜索及动态累加器技术等背景知识;第 2 章，阐述基于动态累加器的去中心化存储的加密搜索方案;第 3 章，实验验证及性能分析；第 4 章，总结全文。

1 背景知识

1.1 区块链技术概述

从数据形态上看，区块链是一串有序串联的数据块（Block）所形成的链条（Chain），且每个新增数据块都会包含前一数据块的特征信息（即区块内容的哈希值）。当包含了前一区块特征信息的新增数据块被加入到区块链上，前一区块中所包含的数据的特征值将被唯一确定并被记录。因此，数据一旦被写入区块链，将被认为是不可篡改、不可删除的，从而形成了前后相连、环环相扣的链状结构。

与传统的分布式数据库（Distributed Database）改善高并发数据访问性能的诉求不同，区块链基于密码学理论及创新的数据结构，能够在缺乏信任的分布式网络（对等网络）环境中保证极致的链上数据的强一致性和不可篡改特性。目前，区块链最成功的应用是以比特币为代表的加密货币（Cryptocurrency）。在加密货币场景中，区块链是所有用户所共同编写的“超级账本”，所有加密货币交易（transaction）都会被永久记录在区块链上。

1.2 加密搜索

传统的搜索技术是基于明文的，用户提交的查询关键字及存储信息均以明文形式存在，恶意服务提供商(Service Provider)、运营商、黑客都有机会获取或截获用户的查询关键字、查询结果及存储的明文数据等信息，造成严重的隐私泄漏及数据安全风险。 为了解决这一问题，加密搜索技术[7,8,9,10]应运而生，提供了对密文进行搜索查询的方案，在这种模式下，搜索过程将不会泄露查询关键字等任何与明文有关的信息，有效保护了用户数据安全及隐私。目前，加密搜索技术已经在云计算场景中得到了较好的应用。

1.3 动态累加器

动态累加器是由 Camenisch 和 Lysyanskaya[12]首先提出，其作用是将一组值累加成一个值，并且能够使输入的任意一个值证明自己被累加到这个累加值中。并且，动态累加器允许动态的添加和删除一些值。2008 年，Peishun Wang 等[13]人对动态累加器作了正式的定义。以下对动态累加器作形式化描述：

（1） KeyGen(k,M)：概率多项式算法，输入参数 1^k 和累加元素的上限 M，返回动态累加器的参数 $P_{uda}=\left(P_{uda-pk},P_{uda-sk}\right)$，其中 P_{uda-pk} 是动态累加器的公钥，P_{uda-sk} 是动态累加器的私钥。

（2）AccVal(L,P)：概率多项式算法，计算出一个累加值。输入参数 P_{uda} 和一组元素 $L\{c_1,\ldots,c_m\}(1<m\le M)$，返回一个累加值 v 和辅助信息 a_c 和 A_i。

（3） WitGen(a_c,A_i,P_{uda})：概率多项式算法：生成每个元素对应的证据值。算法输入辅助信息 a_c 和 A_i 和参数 P_{uda}，输出对每一个 $c_i(i=1,\ldots,M)$ 的证据 $W_i(i=1,\ldots,M)$。

（4） Verify(c,W,v,P_{uda})：确定多项式算法，验证给定的元素是否在累加值 v 当中。输入元素 c、证据 W_i、累加值 v 和公钥 P_{uda-pk}，如果证据 W_i 构成 c 被累加在 v 中的证明，输入 Yes，否则输出 No。

（5） AddEle$(\mathrm{L}^+,a_c,v,P_{uda})$：概率多项式算法，添加新的元素并生成新的累加值。输入一组新的元素 $\mathrm{L}^+=\{c^+_1,\ldots,c^+_k\}(\mathrm{L}^+\subset C,1\le k\le M-m)$，辅助信息 a_c、累加值 v 和参数 P，返回新的累加值 v' 与集合 $\mathrm{L}^+\bigcup\mathrm{L}$ 相一致，返回 $\{W^+_1,\ldots,W^+_k\}$ 为新插入的元素 $\{c^+_1,\ldots,c^+_k\}$ 对应的证据，同时更新辅助信息 a_c 和 a_u，在以后操作中使用；

（6） DelEle$(\mathrm{L}^-,a_c,v,P_{uda})$：概率多项式算法，从累加值中删除某些元素。输入一个元素集合 $\mathrm{L}^-\{c^-_1,\ldots,c^-_k\}(\mathrm{L}^-\subset\mathrm{L},1\le k<m)$ 表示是被删除的元素，辅助信息为 a_c，累加值 v 和参数 P，返回新的累加值 v'，保持与集合 $\mathrm{L}-\mathrm{L}^-$ 相一致，更新辅助信息 a_c 和 a_u，在以后操作中使用；

（7）UpdWit(W_i,a_u,P_{uda})：确定多项式算法，更新已经被累加在 v 和 v' 的证据。输入证据 W_i、辅助信息 a_u 和公钥 P_{uda-pk}，输出一个已更新的证据 W_i'，用来证明元素 c_i 已被累加在新的累加值 v' 中。

2 基于动态累加器的加密搜索方案设计

本文尝试将动态累加器引入到加密搜索方案中，对现有基于区块链的去中心化存储的查询方案进行改良。新方案利用了动态累加器中 *witness* 的高效可验证性及累加值中元素可动态添加和删除的特性，兼顾了效率与灵活性。我们的工作基于 CCS’14 Hahn 的加密搜索方案，引入动态累加器并针对基于区块链的区中心化存储应用场景进行改进。在本章节中，我们首先详细阐述了新方案的各个操作步骤的基本定义，然后结合实例展示在新方案中进行加密及搜索的详细流程。

2.1 功能定义

算法 1. 客户端初始化算法

Algorithm 1 SetUp

1: Client generates symmetric encryption Key key_1,
2: Client generates HMAC key key_2
3: Client generates Dynamic accumulator Key $P_{uda}=(P_{uda-pk},P_{uda-sk}),P_{uda-pk}$ is the public key of the dynamic accumulator,P_{uda-sk} is the private key

定义 1 SetUP：如算法 1 所示。客户端在初始化阶段将应用真随机数生成算法生成加密文件所需密钥 key_1，HMAC 使用的密钥 key_2，动态累加器所需密钥 $p_{uda}=(p_{uda-pk},p_{uda-sk})$，其中 p_{uda-pk} 是动态累加器的公钥，p_{uda-sk} 是动态累加器的私钥。同时，客户端生成搜索历史 σ。

算法 2. 客户端添加搜索令牌及累加值算法

Algorithm 2 AddToken

1: Client encrypts *file* by key_1 to generate the cipher c
2: Client chooses keywords $w_1,w_2,...,w_n$,n is the length of the keywords
3: Client uses HMAC to generate the Token, $T_{w_i}=HMAC_{key_2}(w_i)$
4: Client inits a search history σ
5: Client inits a empty list X
6: **if** This is a new file f and doesn't have a dynamic accumulator value v **then**
7: Client generates the v by (v, a_c,A_l)=**AccVal**$(P_{uda},T_{w_1},...,T_{w_n}),a_c$ and A_l are the additional information
8: **else**
9: Client generates the v by (v, a_c,A_l)=**AddEle**$(T_{w_1},...T_{w_n},v,a_c,P_{uda})$
10: **end if**
11: Client generates witness value of each token by **WitGen**(a_c,A_l,P_{uda})
12: **if** T_{w_i} belongs to σ **then**
13: $X = X \cup witness_i$
14: **end if**
15: Client generates $Trans_{add}(\mathrm{ID}(f),v,X,URL)$ to interact with the blockchain, $\mathrm{ID}(f)$ is the id of the file, URL is the storage url of the encrypted file

定义 2 AddToken：如算法 2 所示。客户端首先采用 $ENC_{key_1}(file)$ 生成密文 c，客户端可以采用任何安全的对称加密算法，本方案对文件加密算法不作限制。从文件中选取一组关键词集合 $keywords\{w_1,\cdots,w_n\}$。客户端通过 $T_{w_i}=HMAC_{key_1}(W_i)$ 生成搜索令牌 T_{w_i}。在客户端对所有关键词进行相同操作后，若该文件为新增文件，则客户端对所有的搜索令牌进行累加值生成操作 $(a_c,A_i,v)=AccVal(T_{w_1},\cdots,T_{w_n},P_{uda})$，$v$ 是动态累加器的累加值，a_c 以及 A_i 是辅助信息。若客户端是为已经存储的文件添加关键字，则通过 $(a_c,A_i,v)=AddEle(T_{w_1},\cdots,T_{w_n},P_{uda},v,a_c)$ 生成新的动态累加器累加值 v。在累加值生成后，客户端通过 $WitGen(a_c,A_i,P_{uda})$ 生成每个 T_{w_i} 的 $Witness_i$。若 T_{w_i} 属于搜索历史 σ，则 $X \cup Witness_i$，X 是一个空的列表。此后客户端生成添加的搜索令牌的交易 $Trans_{add}(ID(f),v,X,URL)$,其中 $ID(f)$ 为文件 ID,URL 为文件存储位置的链接。

算法 3. 区块链添加搜索令牌算法

Algorithm 3 Add

1: After blockchain receives $Trans_{add}(\mathrm{ID}(f),v,X,URL)$
2: **if** X is not null **then**
3: Add every witness of the X to the inverted index λ_ω, each entry looks like $\{ witness_i : \mathrm{ID}(\overline{f})\}$, $\mathrm{ID}(\overline{f})$ the the set of file ids that files contains the keyword
4: **end if**
5: Update the index λ_f, the entry look likes $\{\mathrm{ID}(f{:}v)\ \}$

定义 3 Add：如算法 3 所示。区块链在接收到 $Trans_{add}(ID(f),v,X,URL)$ 后，首先检查 X 是否为空。如果 X 为非空,则添加 X 中的每一个 *Witness* 到 λ_{ω} 中，其中 λ_{ω} 是一个倒排索引。随后，区块链将 $ID(f)$ 与 v 添加到索引 λ_f 中。

算法 4. 客户端生成搜索令牌算法

Algorithm 4 SearchToken

1: Client chooses keyword w_i, then generate the searchToken by $T_{w_i} = HMAC_{key_2}(w_i)$
2: **if** T_{w_i} doesn't belong to the σ **then**
3: Let $\sigma = \sigma \cup T_{w_i}$
4: **end if**
5: Client generates $witness_i$ through the **WitGen** function
6: Client sends the $Trans_{search}(witness_i, T_{w_i}, P_{uda-pk})$ to the blockchain

定义 4 SearchToken：如算法 4 所示。客户端选择关键词 w_i，并通过 $T_{w_i} = HMAC_{key_2}(w_i)$ 生成搜索令牌 T_{w_i}，然后检查搜索历史 σ，如果搜索令牌 T_{w_i} 不在 σ 中，$\sigma = \sigma \cup T_{w_i}$。客户端调用 *WitGen* 方法生成 $Witness_i$，随后客户端发送 $Trans_{search} = (witness_i, T_{w_i}, p_{uda-pk})$ 到区块链。

算法 5. 区块链执行搜索算法

Algorithm 5 Search

1: After blockchain receives $Trans_{search}(witness_i, T_{w_i})$, blockchain searchs the inverted index λ_{ω} first
2: **if** Blockchain finds an entry match the $witness_i$ **then**
3: return the $\mathbf{I}_w$ to the client, which $\mathbf{I}_w$ means a list of files contains this keyword separately and equals ID($\overline{f}$)
4: **end if**
5: Blockchain searchs each entry of the index λ_f,
6: Blockchain runs function **Verify**$(T_{w_i}, witness_i, v, P_{uda-pk})$ for each entry, and return ID(f)
7: Blockchain make a set $\mathbf{I}_w = \mathbf{I}_w \cup \mathrm{ID}(f)$ and return the $\mathbf{I}_w$ to the client

定义 5 Search: 如算法 5 所示。区块链通过智能合约首先检查倒排索引 λ_{ω} 的每一项，看 $Witness_i$ 是否存在。如果存在，返回 I_w，其中 I_w 代表一组文件的 ID 及存储信息。若不存在，调用 *Verify* 方法，对索引中每一项的累加值 v 与 $witness_i$ 和 I_{w_i} 进行匹配。在验证完所有项后，将结果为真的项组成集合 I_w 返回给发起搜索用户。

2.2 系统流程

图 1 展示了典型的文件上传至基于区块链的去中心化存储设施并进行加密索引及对加密索引搜索并下载的流程。用户在本地调用 SetUp 方法初始化所需要的密钥，做好准备工作。当用户需要在第三方存储文件时，用户会将文件进行加密然后上传至 storage peer 并获取 URL。其后，用户会从文件中筛选出若干个关键词，随后调用 AddToken 方法，生成与文件 ID 对应的动态累加器累加值及与搜索历史相关的参数 X。而后，用户发送 $Trans_{add}(ID(f),v,X,URL)$ 交易到区块链，调用区块链中对应合约 Add 方法。区块链合约将对 $Trans_{add}(ID(f),v,X,URL)$ 中的参数进行存储，生成相关的索引信息。当用户需要请求某个已上传文件时，用户发送 $Trans_{search}(witness_i, T_{w_i}, P_{uda-pk})$ 交易请求调用区块链合约中的 Search 方法，区块链将利用动态累加器验证 witness 的方法，对索引进行遍历和比对操作，返回用户一系列与文件相关 URL。用户依据 URL 从 storage peer

获取加密的文件到本地并本地密钥对密文文件进行解密。

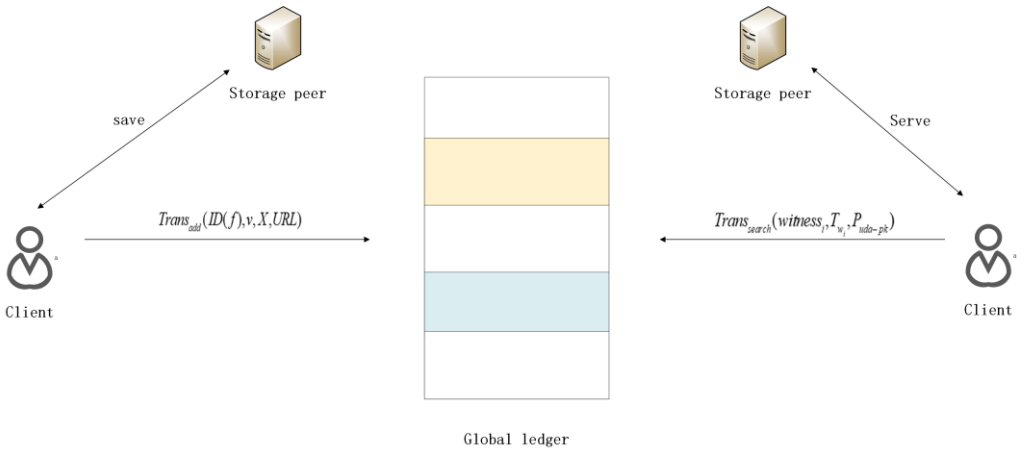

Fig.1 系统基本流程

3 方案分析

3.1 安全性分析

3.1.1 令牌、动态累加器的安全性分析

(1) 令牌安全性：

- 单向性：选取任意 *keyword*，可由 $T_w = HMAC_{key}(keyword)$,但已知 T_w *keyword* 及 *key* 不可反推出 *keyword*
- 抗碰撞性：选择任意 $keyword_1$ 和 $keyword_2$，令 $HMAC_{key}(keyword_1)=HMAC_{key}(keyword_2)$ 是概率极低近乎不可能的。同样的，选取任意 key_1 和 key_2，令 $HMAC_{key_1}(keyword)=HMAC_{key_2}(keyword)$ 是概率极低近乎不可能的。

(2) 动态累加器安全性

- 安全性：生成任意两组元素 $L_1\{c_1,\cdots,c_m\}$ 和 $L_2\{n_1,\cdots,n_m\}$，当 $L_1 \neq L_2$ 时，令 $AccVal(L_1,P)=AccVal(L_2,P)$ 是概率极低近乎不可能的。同样的，当 $P_1 \neq P_2$，令 $AccVal(L_1,P_1)=AccVal(L_1,P_2)$ 是概率极低近乎不可能的。同样的，令 $AccVal(L_1,P_1)=AccVal(L_2,P_2)$ 也是极低概率事件。

3.2 性能分析

本方案在引入动态累加器对加密搜索方案进行改进后，将加密搜索的空间复杂度进一步降低，显著提升加密搜索效率。其原因如下：在原方案 CCS'14 Hahn 中，每次搜索将对索引 λ_f 和倒排索引 λ_ω 进行遍历。尤其是在索引 λ_f 中，每一行的都是以 $\{ID(f):HMAC_{T_{w_1}}(S_1)\|S_1,\cdots,HMAC_{T_{w_n}}(S_n)\|S_n\}$ 这种形式存在，当服务器端接收到搜索令牌后，每一行都需要进行 n 次的 *HMAC* 运算(假设平均每个文件拥有 *n* 个关键词 *keyword*)，若当前索引 λ_f 中存在 m 个文件，则 λ_f 中执行一次搜索需要 $O(n)=m*n$ 次比对。在本方案中，由于动态累加器的引入，在索引 λ_f 中的每一行将以 $\{ID(f):v\}$ 这种聚合值的形态出现，在进行一次搜索时，每一行仅需要进行一次校验操作验证 $Witness_i$ 是否在 *v* 中存在。因此，在对 m 个文件进行关键字搜索时，仅需要比对 $O(n)=m$ 次。如图 2 所示，给定文件的平均 keyword 越多时，本方案的效率越高。同理，若给定文件的 keyword 数量一定，文件数量越大，本方案的效率优势越明显，如图 3 所示。

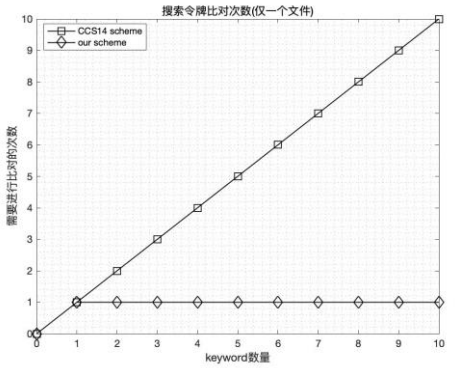

Fig.2 给定文件，keyword 数量增加，搜索令牌比对次数对比

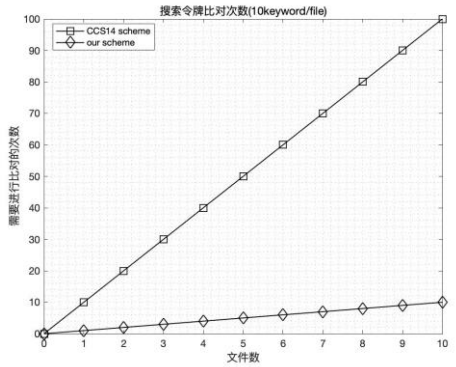

Fig.3 keyword 数量不变，文件数量增加，搜索令牌比对次数对比

4 结论

本文首先阐述了区块链技术、加密搜索和动态累加器的相关背景知识。针对现有的去中心化存储场景阐述了基于加密搜索方法的去中心化存储数据安全及用户隐私保护方案，并提出了基于动态累加器算法的加密搜索改进方案并通过实验证明了该方案的可行性及有益效果。实验结果表明新方案有效提升了去中心化存储场景中的加密搜索效率，有效保护用户数据安全及隐私。

References:

[1] SATOSHI N． Bitcoin:A Peer-to-Peer Electronic Cash System [EB/OL]．https://bitcoin．org/bitcoin．pdf

[2] The Bitcoin Project, “Bitcoin.” On line at: https://bitcoin.org/en/.

[3] The Ethereum Project, “Ethereum.” On line at: https://ethereum.org.

[4] The Storj Project, “Storj.” On line at: https://storj.io/storj.pdf.

[5] The Filecoin Project, “Filecoin.” On line at: http://filecoin.io/filecoin.pdf.

[6] C, Gray. “Storj vs. dropbox: Why decentralized storage is the future.” On line at: https://bitcoinmagazine.com/articles/storj-vs-dropbox-decentralized-storage-future-1408177107, 2014

[7] R, Curtmola, J, A, Garay, S, Kamara, and R, Ostrovsky, “Searchable symmetric encryption: Improved definitions and efficient constructions,” Journal of Computer Security, vol. 19, no. 5, pp. 895-934, 2011

[8] D, Cash, J, Jaeger, S, Jarecki, C, Jutla, H, Krawczyk, M,-C. Rosu, and M, Steiner, “Dynamic searchable encryption in very large databases: Data structures and implementation,” in Proc. of NDSS, 2014.

[9] S, Kamara, C, Papamanthou, and T, Roeder, “Dynamic searchable symmetric encryption,” in Proc. of ACM CCS, 2012.

[10] F, Hahn and F, Kerschbaum, “Searchable encryption with secure and efficient updates,” in Proc. of ACM CCS, 2014.

[11] ChengJun, C, XinLiang, Yuan, Cong, Wang. “Towards Trustworthy and Private Keyword Search in Encrypted Decentralized Storage,” in Communication and Information Systems Security Symposium, 2017.

[12] Camenisch, J, Lysyanskaya, A. “Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials[M]Advances in Cryptology,” — CRYPTO 2002. Springer Berlin Heidelberg, 2002:61-76.

[13] Wang, P, Wang, H, Pieprzyk, J. A New Dynamic Accumulator for Batch Updates.”Information and Communications Security, International Conference”, ICICS 2007, Zhengzhou, China, December 12-15, 2007, Proceedings. DBLP, 2007:98-112.