基于动态布谷鸟过滤器的多关键词可搜索加密方案

王谦[1,2]，赵圣楠[2]，赵川[2]，荆山[1,3]，陈贞翔[1,3]
（1. 济南大学信息科学与工程学院，山东 济南 250022；2. 泉城省实验室，山东 济南 250103；
3. 济南大学山东省网络环境智能计算技术重点实验室，山东 济南 250022）

摘　要：已有多关键词可搜索加密方案在服务器端使用布谷鸟过滤器（CF）实现了多关键词搜索。CF 的容量是预设的，这无法满足服务器端持续增长的扩容需求。为解决上述问题，提出了动态多关键词可搜索加密（DMKSE）方案。DMKSE 使用动态布谷鸟过滤器来解决服务器端的扩容需求，同时改进了文档-关键词对的存储方式，提高了服务器端的空间利用率，且没带来额外的时间开销。实验结果表明，DMKSE 的空间利用率至少比原方案提高了 1.348 1 倍。

关键词：可搜索加密；多关键词查询；动态布谷鸟过滤器；空间利用率
中图分类号：TP309
文献标志码：A
DOI：10.11959/j.issn.1000–436x.2024219

Multi-keyword searchable encryption scheme based on dynamic cuckoo filter

WANG Qian[1,2], ZHAO Shengnan[2], ZHAO Chuan[2], JING Shan[1,3], CHEN Zhenxiang[1,3]
1. School of Information Science and Engineering, University of Jinan, Jinan 250022, China
2. Quan Cheng Laboratory, Jinan 250103, China
3. Shandong Provincial Key Laboratory of Network-based Intelligent Computing, University of Jinan, Jinan 250022, China

Abstract: Existing solutions have utilized Cuckoo Filter (CF) on the server side to achieve multi-keyword searching.The capacity of CF is preset, which can not satisfy the continuous growth of the capacity expansion demand on the server side. To solve the above problems, a dynamic multi-keyword searchable encryption (DMKSE) was proposed, which used the dynamic cuckoo filter to solve the server-side capacity expansion demand, and improved the storage of document-keyword pairs, which improved the space utilization of the server-side, without additional time overhead. Experimental results show that DMKSE improves space utilization by at least 1.3481 times over the original scheme.

Keywords: searchable encryption, multi-keyword, dynamic cuckoo filter, space utilization

0　引言

由于本地存储的缺点（地域限制、昂贵的劳动力成本和机器维护成本等）和云计算的优点（灵活性、资源无限、可承受性等），个人和组织更倾向于将数

收稿日期：2024–10–10
通信作者：赵川，ise_zhaoc@ujn.edu.cn
基金项目：国家自然科学基金资助项目（No.62472252, No.62172258）；泰山学者工程基金资助项目（No.tsqn202211280）；山东省自然科学基金资助项目（No.ZR2023LZH014, No.ZR2022ZD01, No.ZR2022MF264, No.ZR2021LZH007）；山东省重点研发计划基金资助项目（No.2021SFGC0401, No.2021CXGC010103）；山东省科学技术厅基金资助项目（No.SYS202201）；泉城省实验室重大项目（No.QCLZD202302）
Foundation Items: The National Natural Science Foundation of China (No.62472252, No.62172258), TaiShan Scholars Program (No.tsqn202211280), Shandong Provincial Natural Science Foundation (No.ZR2023LZH014, No.ZR2022ZD01, No.ZR2022MF264, No.ZR2021LZH007), Shandong Provincial Key R&D Program of China (No.2021SFGC0401, No.2021CXGC010103), Department of Science & Technology of Shandong Province (No.SYS202201), Quan Cheng Laboratory (No.QCLZD202302)

据存储在服务器中。然而，服务器不可信会导致数据泄露问题。为了保护数据隐私，可搜索加密技术应运而生，这是一种支持关键词密文搜索的加密原语。

根据搜索过程中关键词数量，可搜索加密可分为单关键词可搜索加密和多关键词可搜索加密。多关键字可搜索加密方案[1-3]可以通过单关键字可搜索加密和过滤器实现，比较经典的过滤器是布隆过滤器（BF, Bloom filter）[4]和布谷鸟过滤器（CF, cuckoo filter）[5]。过滤器用于存储加密文档索引及该文档对应的关键字，这样能够快速判断一个文档中是否含有当前搜索关键字。动态可搜索加密支持动态添加和删除文档和关键字。BF和CF的容量是固定的，不能满足动态可搜索加密的实际需求。现有基于过滤器的多关键字可搜索加密方案[1-3]没有考虑过滤器容量固定问题。

CF支持元素删除操作，因此CF比BF能够更好地支持动态可搜索加密。链式动态布谷鸟滤波器（C-DCF, chain dynamic cuckoo filter）[6]是第一个扩展CF容量的方案。Zhang等[7]设计了对数动态布谷鸟滤波器（L-DCF, logarithmic dynamic cuckoo filter）提高搜索效率。符鹏涛等[8]提出跳跃动态布谷鸟滤波器（J-DCF, jump dynamic cuckoo filter）实现了更高的空间效率。上述动态布谷鸟过滤器均由多个CF组成，通过增减CF数量来调整容量。

在文献[2]中，CF的一个桶中仅存储一个文档索引和相应的加密关键字。由于CF中的桶大小需要满足最大需求，而各文档所含关键词数量不一，使关键词较少的文档在对应桶中留有一些空闲槽，这导致了大量的空间浪费。

为了解决上述问题，本文提出了动态多关键字可搜索加密（DMKSE, dynamic multi-keyword searchable encryption）方案。本文的主要贡献如下。

1) DMKSE通过在服务器端使用动态布谷鸟过滤器实现了服务器端容量动态调整，满足动态可搜索加密中更新文档的实际需求。实验测试了C-DCF、L-DCF和J-DCF的更新和插入时间，C-DCF的更新速度最快，J-DCF搜索速度最快。

2) DMKSE设计了正向索引存储方式。首先使用结构体设计了文档−关键词对的存储方式，进而根据文档索引将结构体存储至动态布谷鸟过滤中。实验结果表明，DMKSE的空间利用率至少是已有方案的1.348 1倍。

3) DMKSE能够支持文档和关键词的增加和删除，同时在更新过程中能够抵抗文件注入攻击，具有前向安全。

1 相关工作

Song等[9]提出了第一个可搜索加密方案，该方案是基于对称加密原语实现的。Goh等[10]通过设计正向索引实现了可搜索加密，与Song等[9]方案遍历搜索相比，其搜索速度更快。为了进一步加快搜索效率，文献[11]提出了基于倒排索引的可搜索加密方案，能够根据关键词密文直接定位其对应的文档索引。

早期的可搜索加密方案[9-10,12]主要集中于静态可搜索加密，不支持数据更新。Kamara等[13]提出了动态可搜索加密方案，然而，更新操作可能会导致信息泄露。文献[14]正式定义了前向安全的概念，未来的更新与之前的搜索不相关，可以抵抗文件注入攻击，并提出了具有前向安全的单关键字可搜索加密方案。该方案基于非对称加密原语，非对称加密比对称加密计算和通信成本更高[15]。为降低计算和通信成本，Wei等[16]基于对称加密提出了单关键字可搜索加密方案——FSSE。FSSE具有前向安全，在令牌生成和搜索操作中具有较高的效率。

Hu等[1]通过在客户端添加BF将Σοψος[14]转换为多关键字可搜索加密，该方案增加了客户端存储开销。为进一步减少客户端存储开销，文献[2]通过在服务器端添加CF，将FSSE[16]转换为多关键字可搜索加密，该方案在使用CF存储文档−关键词对的过程中存在空间浪费问题且不能满足服务器端持续的扩容需求。文献[3]基于非对称加密原语CF实现了多关键字可搜索加密，与文献[2]相同，文献[3]中的CF也不能满足更新过程中扩容需求。非对称加密通信及计算开销高于对称加密，本文主要关注对称加密。

此外，布尔搜索、模糊关键字搜索、范围搜索和语义搜索等各种方案[17-20]被提出，用来增强可搜索加密的表达能力。

2 预备知识

2.1 动态布谷鸟过滤器

基于文献[21]，Fan等提出了CF[5]。CF可以快速判断一个元素是否属于一个集合，被广泛用于隐私保护中[22-23]。CF本质上是一个二维哈希表，其中第一维由β个桶组成，第二维由b个槽组成。CF容量是固定的，在初始化时确定。

为实现容量扩展，动态布谷鸟过滤器被提出。动态布谷鸟过滤器由CF构成，通过添加或删除CF的个数，实现容量的动态变化。不同的动态布谷鸟过滤采用不同的方法来组织这些CF。C-DCF[6]将多个CF组合成一个链表，通过在链表尾部附加一个空CF来扩展容量，从链表中删除一个CF来减少容量。为提高C-DCF的搜索效率，L-DCF[7]被提出。L-DCF将多个CF组合成二叉树，通过将空CF附加到树的叶子来扩展容量，从树中删除CF来减少容量。为进一步降低L-DCF的空间开销，符鹏涛等[8]提出了J-DCF。J-DCF维护一个由多个CF组成的动态列表，使用跳跃一致哈希算法[24]在元素和CF之间建立一对一的映射关系，J-DCF通过在列表末尾添加或删除CF实现容量动态变化。

2.2　索引结构

正向索引和倒排索引是信息检索领域常用的2种索引结构。如图1所示，正向索引是文档ind到该文档包含关键词的映射，可以快速确定某个关键词是否存在文档中。其中，ind表示文档索引，w表示关键词。

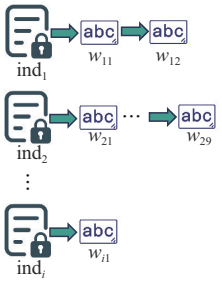

图1　正向索引

如图2所示，倒排索引是关键词到包含该关键词的文档ind的映射，可以快速搜索包含某个关键词的所有文档。其中，ind表示文档索引，w表示关键词。

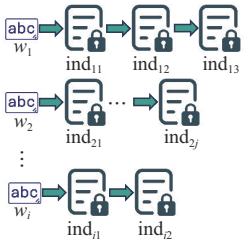

图2　倒排索引

2.3　安全性定义

现实世界的游戏相当于动态可搜索加密方案中的操作，理想世界的游戏反映了模拟器$\mathcal{S}$的操作。泄露函数$\mathcal{L}=(\mathcal{L}^{\text{Setup}},\mathcal{L}^{\text{Search}},\mathcal{L}^{\text{Update}})$定义了在初始化、搜索和更新过程中敌手$\mathcal{A}$获取的信息。

$\text{Real}_{\mathcal{A}}(\lambda)$。敌手$\mathcal{A}$选取数据库DB，客户端和服务器端运行初始化算法$\text{Setup}(\lambda)$得到加密数据库EDB，$\lambda$为安全参数；敌手$\mathcal{A}$在EDB中进行搜索或更新操作获取信息，最终敌手$\mathcal{A}$根据所获取的信息输出$b$，$b\in\{0,1\}$。

$\text{Ideal}_{\mathcal{A},\mathcal{S}}(\lambda)$。敌手$\mathcal{A}$选取数据库DB，模拟器$\mathcal{S}$根据泄露函数$\mathcal{L}^{\text{Setup}}(\lambda)$生成加密数据库EDB′，$\lambda$为安全参数；敌手$\mathcal{A}$在EDB′中进行搜索或更新操作，模拟器$\mathcal{S}$将搜索结果和更新结果发送给敌手$\mathcal{A}$；敌手$\mathcal{A}$根据所获取的信息输出$b$，$b\in\{0,1\}$。

定义1　$\mathcal{L}$-自适应安全。若存在一个模拟器$\mathcal{S}$，对任意概率多项式时间敌手$\mathcal{A}$，都满足$P[\text{Real}_{\mathcal{A}}(\lambda)=1]-P[\text{Ideal}_{\mathcal{A},\mathcal{S}}(\lambda)=1]\leqslant\text{neg}(\lambda)$，其中$\text{neg}(\lambda)$为可忽略函数，则称该可搜索加密方案是$\mathcal{L}$-自适应安全的。

定义2　前向安全。如果具有$\mathcal{L}$-自适应安全的动态可搜索加密方案的更新泄露函数$\mathcal{L}^{\text{Update}}$可以表示为$\mathcal{L}^{\text{Update}}(i,\text{op}_i,\text{ind}_i,w)=(i,\text{op}_i,\text{ind}_i)$，则该方案具有前向安全。其中，$i$表示时间戳，$\text{op}_i$表示更新操作的类型，$\text{ind}_i$和$w$分别表示更新的文档和关键词。换句话说，若具有$\mathcal{L}$-自适应安全的动态可搜索加密方案在更新操作中只泄露操作类型op_i和被更新的文档ind_i，则该方案具有前向安全。

3　动态多关键词可搜索加密方案DMKSE

3.1　系统模型

如图3所示，DMKSE方案包括用户和服务器。用户将文档和关键字外包给服务器之前对其进行加密。服务器根据用户发送的加密关键字（搜索令牌）进行查询，然后将匹配到加密的文档索引发送给用户。DMKSE允许用户在服务器上添加或删除文档或关键字。

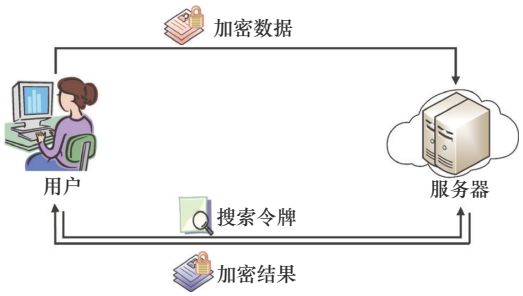

图3　系统模型

3.2　方案形式化定义

1) 初始化协议 Setup(1^{λ})。客户端输入安全参数 λ，输出密钥 uk、空映射 W；服务器端输出空树 T、空动态布谷鸟过滤器 DCF。

2) 更新协议 Update(op,ind,$\{w_1,w_2,\cdots,w_n\}$)。客户端输入更新操作 op、更新的文档索引 ind 及关键词 $\{w_1,w_2,\cdots,w_n\}$，服务器端输出更新后的树 T 及动态布谷鸟过滤器 DCF。

3) 搜索协议 Search($\{w_1,w_2,\cdots,w_m\}$)。客户端输入待搜索的多关键词 $\{w_1,w_2,\cdots,w_m\}$，服务器端将搜索结果 R_2 发送给客户端。

3.3　方案构造

首先，本文基于体构造新的文档−关键词对存储方式，进而设计了正向索引存储方法，最后结合方案 FSSE[17] 和所设计的正向索引存储结构来实现 DMKSE。

图 4 为本文所设计的文档−关键词对的存储方法，其中，ind 表示文档索引，K 表示加密的关键词集合。结构体中包含 2 个变量，文档索引和加密关键词集合。由于集合大小可变在存储文档−关键词对时不存在空间浪费问题。

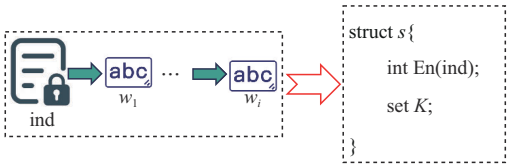

图 4　文档−关键词对的存储方法

以 L-DCF 为例，DMKSE 中正向索引的存储结构如图 5 所示。正向索引存储在动态布谷鸟过滤器中，动态布谷鸟过滤器的每个槽存储一个结构体，结构体的指纹及其所在的槽由加档索引确定，如式(1)所示，其中 En(·) 表示 AES 对称加密算法[25]（本文中 En(·) 均表示 AES 对称加密算法）。

$$\begin{aligned} f &= \text{fingerprint}(s.\text{En}(\text{ind})) \\ p_1 &= \text{Hash}(s.\text{En}(\text{ind})) \\ p_2 &= p_1 \oplus \text{Hash}(f) \end{aligned} \tag{1}$$

初始化　初始化过程如协议 1 所示。该协议为客户端构造密钥 uk 和一个空映射 W。为服务器建立一个空的二叉搜索树 T 和空的动态布谷鸟过滤器 DCF。映射 W 中包含密钥 key 和与每个关键词 w_i 对应的文档索引 id。

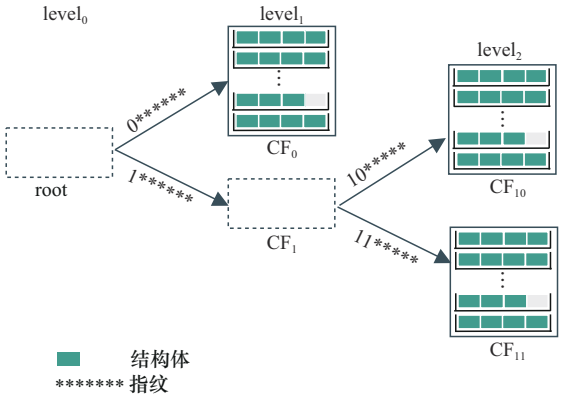

图 5　DMKSE 中正向索引的存储结构

协议 1　初始化协议 Setup(1^{λ})

客户端

1) uk ← $\{0,1\}^{\lambda}$

2) W ← 空映射

服务器端

1) T ← 空树

2) DCF← 空的动态布谷鸟过滤器

更新　文档索引 ind 及关键词 $\{w_1,w_2,\cdots,w_n\}$ 更新过程如协议 2 所示，其中，$F(\cdot)$ 是伪随机函数，$\text{Hash}_{\text{En}}(\cdot)$ 是加密哈希函数 SHA-256。更新操作包括增加和删除，用 op 表示，op = 1 表示增加，op = 0 表示删除。客户端首先为每个 (ind,w) 生成一个块，块包含两部分，即块索引 id^* 和数据部分 data。其中，数据部分包括文档索引 ind、op、下一个块索引 id、下一个块密钥 key。具有相同关键字的块链接在一起形成一条链，构建了倒排索引。进而客户端为一个文档索引 ind 及其对应关键词 $\{w_1,w_2,\cdots,w_n\}$ 构造一个结构体，客户端将块和结构体发送给服务器。服务器根据块索引将块插入二叉搜索树 T 中，也就是说，倒排索引存储在树 T 中。服务器根据结构体中加密文档索引将结构体插入动态分区过滤器中，构建了正向索引，即正向索引存储在动态分区过滤器中。

协议 2　更新协议 Update(op,ind,$\{w_1,w_2,\cdots,w_n\}$)

客户端

1) for i = 1; $i \leqslant n$; $i = i + 1$ do

2)　　(id,key) ← ($W[w_i]$.id,$W[w_i]$.key);

3)　　γ ← $\{0,1\}^{\lambda}$;

4)　　id^* ← $\{0,1\}^{\mu}$;

5)　$key^{*} \leftarrow F_{uk}(\gamma)$;
6)　$mask \leftarrow Hash_{En}(key^{*},id^{*})$;
7)　块 $b \leftarrow [id^{*},data=([En(ind)||op]||key||id)\oplus mask]$
8)　$W[w_i].id \leftarrow id^{*}$;
9)　$W[w_i].key \leftarrow key^{*}$;
10)　if $W[w_i].cnt = \perp$ then
11)　　$W[w_i].cnt = 0$;
12)　else
13)　　if op = 1 then
14)　　$W[w_i].cnt++$;
15)　　else
16)　　$W[w_i].cnt--$;
17)　　end if
18)　end if
19)　插入块 b 到集合 B 中;
20) end for
21) 集合 $K \leftarrow \{En(w_1),En(w_2),\cdots,En(w_n))\}$;
22) 结构体 $s \leftarrow [En(ind),K]$;
23) 发送 B 和 s 到服务器端;
服务器端
1) 基于块索引将集合 B 中的块插入树 T 中;
2) $f = fingerprint(s.En(ind))$;
3) $p_1 = Hash(s.En(ind))$;
4) $p_2 = p_1 \oplus Hash(f)$;
5) 根据 f、p_1 和 p_2 将 s 插入 DCF

当客户端只更新文档对应的关键词时，与上述过程的唯一不同之处在于动态布谷鸟过滤器中只需更新对应结构体中的关键词。

搜索　搜索过程如协议 3 所示。当客户端在 DMKSE 中搜索关键词 $\{w_1,w_2,\cdots,w_m\}$ 时，首先查询对应文档数量最少的关键字 w（为方便描述将其记作 w_1），查询结果保存到 R_1 中。然后检查 R_1 中的文档索引所在的结构体中是否包含剩余关键字。若包含，则将该文档索引添加到集合 R_2 中，集合 R_2 为最终的查询结果。搜索过程相当于先在倒排索引中查询关键词 w_1，查询结果为 $\{ind_1,\cdots,ind_j\}$，进而依次判断 ind 中是否包含剩余关键词，若包含，则 ind 中包含所有关键词。

协议 3　搜索协议 Search($\{w_1,w_2,\cdots,w_m\}$)
客户端
1) 根据映射 W 中的 cnt 找到对应文档数量最少的关键词 w_1;
2) $(id,key) \leftarrow (W[w_1].id,W[w_1].key)$;
3) $\{En(w_i)\} \leftarrow$ 加密关键词 $(w_2,\cdots,w_m)$;
4) 发送 (id,key) 和 $\{En(w_i)\}$ 到服务器端;
服务器端
1) $R_1,R_2 \leftarrow$ 空集合，$i \leftarrow 0$，$b.ptr \leftarrow id$;
2) while $b.ptr \neq \perp$ do
3)　　$mask \leftarrow Hash_{En}(key,id)$;
4)　　$b \leftarrow T[id].data \oplus mask$; //$T[id]$ 表示二叉搜索树 T 中索引为 id 的块
5)　　$b.ptr \leftarrow b.id$，$key \leftarrow b.key$;
6)　　$En(ind)||op \leftarrow b.[En(ind)||op]$;
7)　　$R_1[i++] \leftarrow En(ind)||op$;
8) end while
9) 将集合 R_1 中成对出现的“$En(ind)||0$”和“$En(ind)||1$”删除;
10) for $En(ind) \in R_1$ do
11)　　$f = fingerprint(En(ind))$;
12)　　$p_1 = Hash(s.En(ind))$;
13)　　$p_2 = p_1 \oplus Hash(f)$;
14)　　根据 f、p_1 和 p_2 在 DCF 中搜索 s;
15)　　if $s.K == En(w_2),\cdots,En(w_m)$ then
16)　　$R_2 \leftarrow En(ind)$;
17)　　end if
18) end for
19) 发送 R_2 到客户端;

4　理论分析

本节首先证明 DMKSE 具有前向安全性，然后分析分析服务器端空间复杂度和时间复杂度并与相关工作进行对比。

4.1　安全性分析

本节在随机预言机模型下证明了 DMKSE 方案是前向安全的。首先证明 DMKSE 方案是 $\mathcal{L}$-自适应安全的，进而证明是前向安全的。由于 DMKSE 是基于单关键词可搜索加密方案 FSSE[16] 实现的且 FSSE 具有前向安全，因此只需证明 DMKSE 中多关键词搜索部分是前向安全的即可。

定义更新泄露函数 $\mathcal{L}^{Update}(i,op_i,ind_i,w) = \perp$，其中 i 表示时间戳，op_i 表示更新操作的类型，ind_i 和

w分别表示更新的文档和关键词。

游戏G_0。G_0是现实世界中游戏，即$P[\mathrm{Real}_{\mathcal{A}}^{\mathrm{DMKSE}}(\lambda)=1]=P[G_0=1]$，其中$\lambda$为安全参数。

游戏G_1。更新文档和关键词时使用随机生成的密钥加密，并将该密钥存储于表KEY中。查询关键词w时，先检查表KEY中是否有该关键词的密钥。若有，直接使用表中的密钥进行加密；若没有，随机生成密钥进行加密，并将该密钥存储至表KEY中。使用随机生成密钥进行更新和搜索的过程如下。

更新过程

客户端

1) $\left\{\{0,1\}^{\lambda},\{0,1\}^{\lambda},\cdots,\{0,1\}^{\lambda}\right\}\leftarrow$随机生成密钥加密关键词$(w_1,\cdots,w_n)$，密钥存储至表KEY中；

2) 集合$K\leftarrow\left\{\{0,1\}^{\lambda},\{0,1\}^{\lambda},\cdots,\{0,1\}^{\lambda}\right\}$；

3) $\{0,1\}^{\lambda}\leftarrow$随机生成密钥加密文档索引ind，密钥存储至表KEY中；

4) $D\leftarrow\{0,1\}^{\lambda}$；

5) 结构体$s\leftarrow[D,K]$；

6) 发送s到服务器端；

服务器端

1) $f=\mathrm{fingerprint}(s.D)$；

2) $p_1=\mathrm{Hash}(s.\mathrm{En}(\mathrm{ind}))$；

3) $p_2=p_1\oplus\mathrm{Hash}(f)$；

4) 根据f、p_1和p_2，将s插入DCF；

搜索过程

客户端

1) $\left\{\{0,1\}^{\lambda}\right\}\leftarrow$表KEY中的密钥加密关键词$(w_2,\cdots,w_m)$；

2) 发送$\left\{\{0,1\}^{\lambda}\right\}$到服务器端；

服务器端

1) $R_1\leftarrow$单关键词搜索结果，$R_2\leftarrow$空集合；

2) for $\left\{\{0,1\}^{\lambda}\right\}\in R_1$ do

3) $f=\mathrm{fingerprint}(\{0,1\}^{\lambda})$；

4) $p_1=\mathrm{Hash}(s.\mathrm{En}(\mathrm{ind}))$；

5) $p_2=p_1\oplus\mathrm{Hash}(f)$；

6) 根据f、p_1和p_2在DCF搜索s；

7) if $s.K==\left\{\{0,1\}^{\lambda}\right\},\cdots,\left\{\{0,1\}^{\lambda}\right\}$then

8) $R_2\leftarrow\{0,1\}^{\lambda}$；

9) end if

10) end for

11) 发送R_2到客户端；

由于G_1使用随机生成密钥加密文档和关键词，与游戏G_0使用AES算法加密的文档和关键词是不可区分的，因此$P(G_0=1)-P(G_1=1)\leqslant\mathrm{neg}(\lambda)$。

模拟器$\mathcal{S}$。模拟器进行更新和搜索操作获得的数据本质都是随机字符串，与游戏G_1相同，因此$P[\mathrm{Ideal}_{\mathcal{A},\mathcal{S},\mathcal{L}_{\mathcal{S}}}^{\mathrm{DMKSE}}(\lambda)=1]-P[G_1=1]=0$。

根据G_0、G_1和$\mathcal{S}$可得，$P[\mathrm{Real}_{\mathcal{A}}^{\mathrm{DMKSE}}(\lambda)=1]-P[\mathrm{Ideal}_{\mathcal{A},\mathcal{S},\mathcal{L}_{\mathcal{S}}}^{\mathrm{DMKSE}}(\lambda)=1]\leqslant\mathrm{neg}(\lambda)$。所以，DMKSE方案是$\mathcal{L}$-自适应安全的。更新过程中文档索引和关键词都是经过AES算法加密的，服务器仅能得到被更新的文件索引及更新操作类型，所以更新过程中的泄露函数可以表示为$\mathcal{L}^{\mathrm{Update}}(i,\mathrm{op}_i,\mathrm{ind}_i,w)=(i,\mathrm{op}_i,\mathrm{ind}_i)$，DMKSE方案具有前向安全。

4.2 空间开销

DMKSE与文献[2]方案的区别在于服务器端存储正向索引方法不同。文献[2]方案在服务器上使用CF来存储正向索引，将单关键字可搜索加密方案FSSE转换为多关键字可搜索的加密。CF中的每个桶存储一个加密的文档索引和该文档对应的所有关键字。第一个槽存储加密的文档索引，其余槽每个存储一个加密的关键字，如图6所示。

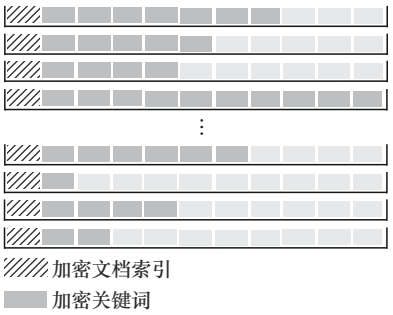

图6 文献[2]方案的正向引存储方式

由于客户端能够进行更新操作，服务器数据不断增加，然而CF容量固定，无法满足实际需求。此外，CF的桶大小取决于文档中关键字的个数，导致空间浪费。例如，有10个文档，每个文档的关键字数量从1到10，那么桶的大小为11（即桶中槽的个数为11）。

为了解决这些问题，本文将文档-关键词对存

储在结构体中，进而插入服务器端的动态布谷鸟过滤器中。与文献[2]方案相比，DMKSE中过滤器容量可扩展且不存在空间浪费问题。

DMKSE在服务器端的空间复杂度为$O(N)+\sum_{i=1}^{i=D}|\omega|_i$，而文献[2]方案为$O(N)+D|\omega|_{\max}$，其中$N$表示关键字−文档对的个数，$|\omega|_i$表示第$i$个文档包含的关键字个数，$|\omega|_{\max}=\mathrm{MAX}(|\omega|_i)$且$i\in[1,D]$。因为$\sum_{i=1}^{i=D}|\omega|_i<D|\omega|_{\max}$，所以DMKSE的空间复杂度低于文献[2]方案。

4.3　时间成本

文献[2]方案和DMKSE的搜索和更新过程的时间复杂度如表1所示。

表1　　**时间复杂度**

过程	文献[2]方案	DMKSE
搜索	$O(t_s a_w W_{LQ})$	$O(t_s a_w W_{LQ})$
更新	$O(t_s)$	$O(t_s)$

表1中，t_s表示对称加密的时间，a_w表示向数据库添加/删除关键字的次数，W_{LQ}表示查询中关键字的数量。虽然DMKSE修改了文献[2]方案的服务器端存储结构，但没有产生任何额外的时间开销。

总之，DMKSE具有较低的空间复杂度，更符合实际需求，不会产生任何额外的时间开销。

5　实验结果

本文测试了DMKSE在服务器端的空间利用率以及使用不同动态布谷鸟过滤器进行更新和搜索的时间开销。所有实验均在分配了4 GB内存和2个核心的CPU的Ubuntu 20.04.3上运行。计算机的配置为Intel(R) Core(TM) i5-10500H 2.50 GHz CPU和16 GB RAM。实验使用安然电子邮件数据集作为测试数据集。

5.1　空间效率

由图6可知，文献[2]方案的空间效率（空间利用率）为$\delta\zeta$ ($\delta\leqslant 1,\zeta\leqslant 1$)，其中$\delta$为桶利用率（非空桶数除以总桶数），$\zeta$为非空桶中槽利用率。例如，CF中有8个桶，则$\delta=37.5\%$。DMKSE的空间效率是过滤器的负载因子，本文用$\alpha$表示。

本文测试了在不同的指纹长度$|f|$下测试δ和α的值，如表2所示。为了控制单一变量，DMKSE中包含一个CF。使用50万封电子邮件的索引作为测试数据集。

表2　　**空间效率**

$\lvert f\rvert$	δ	α
8	60.39%	93.94%
12	63.41%	94.70%
16	67.06%	95.37%
32	70.74%	95.37%

根据表2可以看出，无论指纹长度如何，α都大于δ，α是δ的1.348 1～1.555 5倍。由于$\zeta\leqslant 1$，因此可得

$$1.348\,1\leqslant\frac{\alpha}{\delta\zeta}\leqslant 1.555\,5$$

根据上述关系式可知，DMKSE的空间效率至少是文献[2]方案的1.348 1倍。

5.2　时间开销

本节测试了使用C-DCF、L-DCF和J-DCF进行更新和搜索操作的时间。C-DCF、L-DCF和J-DCF中的CF桶大小为2^{17}，每个桶中包含4个槽。图7显示了不同指纹长度和数据集大小（安然电子邮件的倍数）下过滤器进行更新时间和搜索所需时间。

无论指纹长度和数据集大小如何，C-DCF在更新过程中都是最快的。具体如下。

1) 1倍安然邮件数据集：无论指纹长度如何，L-DCF的时间开销是C-DCF的1.81～1.98倍，J-DCF的时间开销是C-DCF的1.87～1.99倍。

2) 1.5倍安然邮件数据集：不管指纹长度如何，L-DCF的时间开销是C-DCF的1.86～2.08倍，J-DCF的时间开销是C-DCF的1.85～2.13倍。

3) 2倍安然邮件数据集：无论指纹长度如何，L-DCF的时间开销是C-DCF的2.61～2.77倍，J-DCF的时间开销是C-DCF的2.40～3.61倍。

这是因为在更新过程中，L-DCF和JDCF涉及大量的指纹移动，而C-DCF则没有。

同样，无论指纹长度和数据集大小如何，J-DCF在搜索过程中都是最快的。具体如下。

1) 1倍安然邮件数据集：无论指纹长度如何，C-DCF的时间开销是J-DCF的1.32～1.52倍，L-DCF的时间开销是J-DCF的1.05～1.21倍。

2) 1.5倍安然邮件数据集：无论指纹长度如

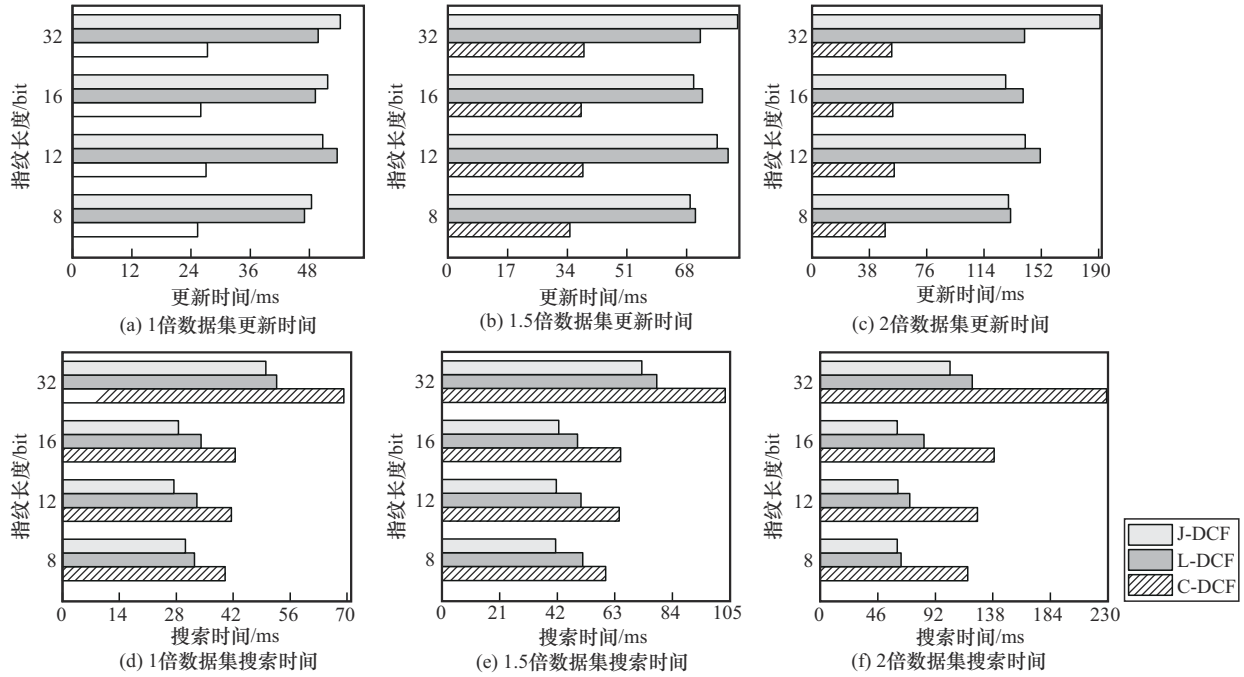

图7　C-DCF、L-DCF和J-DCF在不同配置下的时间开销

何，C-DCF的时间开销是J-DCF的1.42～1.55倍，L-DCF的时间开销是J-DCF的1.07～1.24倍。

3) 2倍安然邮件数据集：无论指纹长度如何，C-DCF的时间开销是J-DCF的1.92～2.25倍，L-DCF的时间开销是J-DCF的1.05～1.35倍。

这是因为J-DCF可以直接定位存储文档的CF，而不需要像C-DCF和L-DCF遍历寻找。

综上所述，C-DCF在更新过程中速度最快，而J-DCF在搜索过程中速度最快。根据具体的实际应用需求选择合适的动态布谷鸟过滤器非常重要。

6　结束语

本文提出的DMKSE方案设计了正向索引存储方法，在不增加时间开销的情况下提高了服务器端的空间利用率和实用性。DMKSE支持动态更新，本文证明了其具有前向安全。实验结果表明，与现有方案相比，空间效率提高了1.348 1倍。使用不同的动态布谷鸟过滤器测试更新和搜索操作所需的时间，实验结果表明更新过程中C-DCF速度最快，而在搜索过程中J-DCF速度最快。

参考文献：

[1] HU C Y, SONG X F, LIU P T, et al. Forward secure conjunctive-keyword searchable encryption[J]. IEEE Access, 2019, 7: 35035-35048.

[2] 汤永利, 李静然, 闫玺玺, 等. 支持联合搜索的动态前向安全可搜索加密方案[J]. 计算机研究与发展, 2022, 59(8): 1853-1866.
TANG Y L, LI J R, YAN X X, et al. A forward secure dynamic searchable encryption scheme supporting conjunctive search[J]. Journal of Computer Research and Development, 2022, 59(8): 1853-1866.

[3] 袁承昊, 李勇, 任爽. 多关键词动态可搜索加密方案[J]. 网络与信息安全学报, 2023, 9(2): 143-153.
YUAN C H, LI Y, REN S. Dynamic multi-keyword searchable encryption scheme[J]. Chinese Journal of Network and Information Security, 2023, 9(2): 143-153.

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

[5] FAN B, ANDERSEN D G, KAMINSKY M, et al. Cuckoo filter: practically better than bloom[C]//Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies. New York: ACM Press, 2014: 75-88.

[6] CHEN H H, LIAO L Y, JIN H, et al. The dynamic cuckoo filter[C]//Proceedings of the 2017 IEEE 25th International Conference on Network Protocols (ICNP). Piscataway: IEEE Press, 2017: 1-10.

[7] ZHANG F, CHEN H H, JIN H, et al. The logarithmic dynamic cuckoo filter[C]//Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE). Piscataway: IEEE Press, 2021: 948-959.

[8] 符鹏涛, 罗来龙, 郭得科, 等. 跳跃滤波: 一种面向大数据治理的动态数据摘要设计[J]. 软件学报, 2023, 34(3): 1193-1212.
FU P T, LUO L L, GUO D K, et al. Jump filter: dynamic sketch design for big data governance[J]. Journal of Software, 2023, 34(3): 1193-1212.

[9] SONG D X, WAGNER D, PERRIG A. Practical techniques for searches on encrypted data[C]//Proceedings of the Proceeding 2000 IEEE Symposium on Security and Privacy. S&P. Piscataway: IEEE Press, 2000: 44-55.

[10] GOH E J. Secure indexes[J]. IACR Cryptology EPrint Archive, 2003, 2003: 216.

[11] 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.
[12] CHANG Y C, MITZENMACHER M. Privacy preserving keyword searches on remote encrypted data[C]//Proceedings of International Conference on Applied Cryptography and Network Security. Berlin: Springer, 2005: 442-455.
[13] KAMARA S, PAPAMANTHOU C, ROEDER T. Dynamic searchable symmetric encryption[C]//Proceedings of the 2012 ACM conference on Computer and communications security. New York: ACM Press, 2012: 965-976.
[14] BOST R. ∑οφος: forward secure searchable encryption[C]//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM Press, 2016: 1143-1154.
[15] BONEH D, CRESCENZO G D, OSTROVSKY R, et al. Public key encryption with keyword search[C]//Proceedings of Advances in Cryptology-EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2004: 506-522.
[16] WEI Y, LV S Y, GUO X J, et al. FSSE: forward secure searchable encryption with keyed-block chains[J]. Information Sciences, 2019, 500: 113-126.
[17] PATEL S, PERSIANO G, SEO J Y, et al. Efficient Boolean search over encrypted data with reduced leakage[C]//Proceedings of International Conference on the Theory and Application of Cryptology and Information Security. Cham: Springer International Publishing, 2021: 577-607.
[18] TONG Q Y, MIAO Y B, WENG J, et al. Verifiable fuzzy multi-keyword search over encrypted data with adaptive security[J]. IEEE Transactions on Knowledge and Data Engineering, 2023, 35(5): 5386-5399.
[19] WANG J F, CHOW S S M. Forward and backward-secure range-searchable symmetric encryption[J]. Proceedings on Privacy Enhancing Technologies, 2021, 2022: 28-48.
[20] YANG W Y, ZHU Y S. A verifiable semantic searching scheme by optimal matching over encrypted data in public cloud[J]. IEEE Transactions on Information Forensics and Security, 2021, 16: 100-115.
[21] PAGH R, RODLER F F. Cuckoo hashing[J]. Journal of Algorithms, 2004, 51(2): 122-144.
[22] DING X Q, ZHAO L S, LUO L L, et al. Gauze: enabling communication-friendly block synchronization with cuckoo filter[J]. Frontiers of Computer Science, 2022, 17(3): 173403.
[23] ZHUANG L Y, GUO N, CHEN Y F. TriNymAuth: triple pseudonym authentication scheme for VANETs based on cuckoo filter and paillier homomorphic encryption[J]. Sensors, 2023, 23(3): 1164.
[24] LAMPING J, VEACH E. A fast, minimal memory, consistent hash algorithm[J]. arXiv Preprint, arXiv:1406.2294, 2014.
[25] SELENT D. Advanced encryption standard[J]. Rivier Academic Journal, 2010, 6(2): 1-14.

[作者简介]

王谦（1999– ），女，山东泰安人，济南大学硕士生，主要研究方向为隐私保护。

赵圣楠（1994– ），男，山东泰安人，博士，泉城省实验室助理研究员，主要研究方向为安全多方计算和隐私集合求交。

赵川（1989– ），男，山东泰安人，博士，泉城省实验室副研究员、硕士生导师，主要研究方向为安全多方计算、隐私保护机器学习。

荆山（1979– ），男，山西阳泉人，济南大学副教授、硕士生导师，主要研究方向为计算机教育、网络管理、下一代互联网。

陈贞翔（1979– ），男，湖南桂东人，博士，济南大学教授、博士生导师，主要研究方向为流量监测和网络安全。