支持结果排序的安全密文检索方法研究

姚寒冰[1,2] 邢娜娜[1] 周俊伟[1,2] 李勇华[1,2]
(武汉理工大学计算机科学与技术学院 武汉 430063)[1]
(交通物联网技术湖北省重点实验室(武汉理工大学) 武汉 430070)[2]

摘 要 越来越多的企业和个人用户将数据部署到低成本、高质量的云存储中。为了保护敏感数据,用户在部署前会对其进行加密处理,但海量的加密数据给检索工作带来很大挑战。文中将传统的倒排索引结构改造成密文倒排索引,并在密文倒排索引上构建计数布隆过滤器,进而提出了基于计数布隆过滤器的密文安全索引(SICBF),其在保证隐私安全的前提下实现了对密文的快速检索。为减少SICBF索引中的数据冗余,设计了计数布隆过滤器的剪枝算法。为保护密文倒排索引中相关分的隐私安全,采用一对多保序加密机制(OPME)对相关分进行加密,并在密文相关分上对检索结果直接进行排序,将最相关检索结果top-k返回给授权用户。安全分析表明,不同于原始数据分布,OPME算法加密后的相关分分布隐藏了数据的峰值,能防止针对相关分的统计攻击。实验结果表明,SICBF的检索效率高,计算量小,适用于海量加密数据文件的快速安全检索。

关键词 倒排索引,相关分,计数布隆过滤器,数据隐私,排序搜索
中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.05.021

Study on Secure Retrieval Scheme over Encrypted Data Supporting Result Ranking

YAO Han-bing[1,2] XING Na-na[1] ZHOU Jun-wei[1,2] LI Yong-hua[1,2]
(School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430063, China)[1]
(Hubei Key Laboratory of Transportation Internet of Things (Wuhan University of Technology), Wuhan 430070, China)[2]

Abstract More and more organizations and users outsource their data into the low-cost and high-quality cloud storage. In order to protect the sensitive data, users will encrypt them before deployment, but this will bring big challenge to retrieval. This paper modified the traditional inverted index into the encrypted inverted index, and then built counting Bloom filter(CBF) on the encrypted inverted index. It proposed secure index based on counting Bloom filter(SICBF) to search encrypted data, which meet strict keyword and index privacy requirements. It also designed the CBF pruning algorithm to reduce redundancy of SICBF index. In order to protect the privacy of the relevance score(RSC) in SICBF, it employed one-to-many order-preserving mapping encryption(OPME) to encrypt RSC. The search results are directly sorted on the encrypted RSC, and only the most relevant top-k documents are returned to the authorized users. Security analysis illustrates that the encrypted RSC distribution is different from the original RSC, which hides the peak value of the RSC to prevent statistical attacks. The experiments show that the SICBF has high retrieval efficiency and low computational cost, which is suitable for searching massive encrypted data.

Keywords Inverted index, Relevance score, Counting bloom filter, Data privacy, Ranked search

1 引言

随着云计算技术的日益成熟,大量的隐私数据被存储在云服务器上。云服务器为用户提供了廉价的存储空间,用户将数据部署到云服务器后,无须关心硬件的维护问题。然而,在云存储给用户带来方便的同时,其安全性和可用性受到人们越来越多的关注[1]。由于云服务器是"诚实但好奇"的,为保护数据隐私安全,用户上传文件之前需要对文件加密,但加密后的数据又给信息检索带来了很大困难[2]。近年来,国内外对密文全文检索问题的研究取得了一定进展。Song等[3]于2002年首次提出了对称可搜索加密方案,该方案基于流密码、伪随机函数和伪随机数生成器,能实现范围可控的安全检索,但存在可搜索攻击问题。针对此问题,Goh[4]提出了安全索引的概念,通过使用布隆过滤器来实现对密文的全文检索。

到稿日期:2017-02-23 返修日期:2017-05-05 本文受国家自然科学基金项目(61601337),湖北省自然科学基金重点项目(ZRZ2015000393),交通物联网技术湖北省重点实验室基金项目(2017III028-002),内河航运技术湖北省重点实验室基金项目(NHHY2017003)资助。
姚寒冰(1976-),男,博士,副教授,主要研究方向为网络与信息安全、数据挖掘,E-mail:yaohb@whut.edu.cn(通信作者);**邢娜娜**(1991-),女,硕士生,主要研究方向为网络与信息安全、信息检索;**周俊伟**(1986-),男,博士,副教授,主要研究方向为信息安全、低复杂度密码算法与分布式信源编码;**李勇华**(1977-),男,副教授,主要研究方向为软件需求工程、软件测试。

Dan 等[5]于 2004 年提出了带关键词搜索的公钥加密方案(PEKS),以解决安全邮件服务器的邮件搜索问题。Wang[6]于 2010 年提出了支持结果排序的单关键字密文检索方案,该方案利用安全索引来实现检索功能,以各个检索词的散列值为索引项,索引项对应的倒排链表内容均被加密保存,排序功能借助于保序加密技术实现,服务器直接对密文相关分进行大小比较,从而实现对结果文档的排序。2014 年,Cao 等提出了支持结果排序的多关键词密文检索方案(MRSE)[7],该方案采用向量空间模型,利用内积相关度方法计算用户输入的检索语句与检索结果文档之间的相关度,解决了多关键词对应文档的排序问题。针对 MRSE 存在的失序问题,Xu 等人提出了多关键词排序搜索方案(MKQE)[8],该方案同样采用向量空间模型表示文档和查询语句,但对查询语句的向量结构做了改进,提高了检索结果的排序特性。为提高检索效率,Sun 等[9]提出了一种改进方案,该方案为所有文档建立一棵索引树,树节点为权重向量,树叶子节点为文档编号,检索时以向量夹角的大小为排序依据。Fu 等[10]于 2013 年提出了为所有文档建立索引树的方案,树节点为 0-1 二元向量,检索时同样根据向量余弦值的计算结果进行排序。Chen 等[11]于 2016 年提出了在大数据量环境下支持更多搜索语义并满足快速密文搜索需求的层次聚类方法,其根据最小相关阈值聚类文档,将生成的簇划分成子簇,直到达到最大簇约束;但该方法只达到了线性复杂度,针对海量密文文档的检索效率仍然不高。

上述方案中,单关键字检索不能满足用户的检索需求,实用性不强;多关键字检索方案为支持检索结果排序,采用向量空间模型计算文档相关分,利用向量内积计算检索语句与结果文档之间的相似程度。随着云服务器文档量的急剧增加,向量的数量及维度都会膨胀,导致相关分计算量大,计算复杂度与文档数量呈线性关系,检索效率低,不适合海量密文文档的检索。

密文的全文检索面临三大安全问题:1)数据机密性;2)索引安全;3)关键词隐私。对用户来说,采用云服务器来存储数据时,数据不再处于自己可控信任域之内。如果不采取有效的加密机制,数据的机密性将受到威胁。此外,在密文检索过程中,如果将各种检索隐私如相关分分布、关键词频率等信息暴露给云服务器,服务器将会根据这些信息发起统计攻击,从而猜测用户检索文档的内容信息[12]。因此,如何既保证云存储数据的安全性,又能快速高效地检索加密数据,是目前密文检索领域面临的挑战。

通过对现有全文检索技术的研究,本文提出了一种基于计数布隆过滤器(Counting Bloom Filter,CBF)的安全密文索引 SICBF。该方法通过 AES 加密算法加密文档和 SICBF 索引中的关键词;通过一对多保序加密算法(One-to-Many Order-Preserving Mapping Encryption,OPME)对相关分进行加密。检索时,根据用户的检索请求生成关键词陷门,通过陷门查找 SICBF 索引,从而快速定位 SICBF 中的倒排表,并根据倒排表中的密文相关分在服务器端完成对检索结果的排序。整个检索过程能保障密文文档、密文索引的机密性,并通过关键字陷门屏蔽检索关键字出现的频率,以保护关键词的隐私安全。

2 密文倒排索引

索引文件是全文检索技术的核心,应用最广泛的全文索引是倒排索引[13-14],它对基于关键词的检索非常有效,被广泛应用于各种信息检索系统。直接将倒排索引应用到密文的全文检索中存在安全问题[15]:倒排索引中包含关键字字典、关键字对应的倒排表,倒排表中包含了词频(TF)、逆文档频率(IDF)和位置(Loc)等计算相关分的信息,容易暴露被索引文档的信息。根据倒排索引的构建过程,可以从两个方面对其进行改进:1)倒排索引直接存储相关分并加密;2)加密明文倒排索引中的索引关键字。

2.1 相关分

倒排索引根据词频和逆文档频率计算查询语句与文档之间的相关分,进而根据相关分对检索结果进行排序。本文在倒排索引中直接存储计算好的相关分,如图 1 所示。

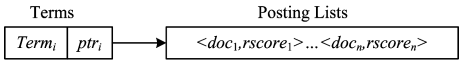

图 1 改进的倒排索引结构

Fig. 1 Structure of improved inverted index

文档相关性的计算方式主要有两种:1)向量空间模型;2)$TF*IDF$ 准则。向量空间模型以向量的形式来表示文档和查询语句[16],将检索转化为向量空间的向量匹配问题。因为每个文档都有一个向量表示,当文档数增加时,由文档产生的关键词集合会变大,向量的维度也会变大。检索时,每个文档向量都需要和查询向量相乘来计算相关度,计算量非常大,检索时间与文档数量呈线性关系,因此该方式不适用于海量密文检索。

本文采用 $TF*IDF$ 准则来计算相关分[17]。其中,TF 指词频,即关键词在文档中出现的次数,代表关键词在匹配文档中的相关度;IDF 指逆文档频率,用于衡量文档中某一关键词在该文档库中的重要程度。相关分的计算公式如下:

$$rscore(Q,F_d)=\sum_{q\in Q}\left(\frac{TF_q}{|F_d|}*IDF_q\right)=\sum_{q\in Q}\left(\frac{TF_q}{|F_d|}*\log\frac{N}{N_q}\right) \tag{1}$$

其中,Q 表示检索关键词集合,TF_q 表示关键词 q 在文档 F_d 中的词频 TF 值,N_q 表示包含关键词 q 的文档数,N 表示全部文档数,$|F_d|$ 表示文档 F_d 的长度。

2.2 密文相关分

为保证相关分的安全性,需要对相关分进行加密处理。本文采用一对多保序加密算法 OPME 对相关分进行加密[18]。在该算法中,明文上有序的数据经过加密后,在密文上仍然保持有序。OPME 采用 Order-Preserving Symmetric Encryption 随机的 plaintext-to-bucket 映射机制[19],在随机种子中加入了文档标识符 ID,使得相同的相关分明文在值域 R 中的映射值不相同,密文相关分与明文相关分呈现不同的数据分布。因此,该方法能隐藏明文相关分的数据分布,有效抵挡对相关分的统计攻击,如算法 1 所示。OPM 的参数可表示为:$OPM:\{0,1\}^{l}\times\{0,1\}^{\log|D|}\rightarrow\{0,1\}^{\log|R|}$,其中,$k,l,l',p$ 为安全参数,D 为保序函数 $f(x)$ 的定义域,R 为其值域。数据

拥有者调用密钥生成算法 $Keygen=(1^k,1^l,1^{l'},1^p,|D|,|R|)$ 生成随机密钥 $x,y,z\xleftarrow{R}\{0,1\}^k$，然后输出密钥 $K'=\{x,y,z,1^l,1^{l'},1^p,|D|,|R|\}$。

算法 1　一对多保序映射加密算法 OPME

输入：定义域 D，值域 R，明文 m，文档标识符 ID；

输出：密文 c

1. $OPME_{k'}$(D,R,m,ID)；
2. while |D|!=1 do
3. {D,R}←BinarySearch(K′,D,R,m)；/ * 二分查找，确定明文 m 的范围 * /
4. coin $\xleftarrow{R}$TapeGen(K′,(D,R,1‖m,ID)),c $\xleftarrow{R}$R
5. return c.

其中，$TapeGen(\cdot)$是一个随机硬币生成函数，$BinarySearch(\cdot)$函数用于确定 m 的映射范围。

云服务器根据密文相关分对检索结果进行归并排序。归并排序是效率较高且稳定的排序算法，其平均时间复杂度为 $O(n\log n)$。

2.3　密文倒排索引

本文采用的倒排索引构建技术与传统信息检索技术类似[20]：给定文档集合 D，首先对其进行分词，提取关键词得到关键词集合 $W=\{w_1,w_2,\cdots,w_l\}$；然后使用 AES 对称密钥 KP_D 对倒排索引中的关键词进行加密，采用 OPME 算法对相关分进行加密。索引构建算法的思想如下：

(1)给定文档集合 D，提取关键词集合 $W=\{w_1,w_2,\cdots,w_l\}$；同时构建文档标识符 ID_{w_i}，$\forall w_i\in W$，ID_{w_i} 是指包含关键词 w_i 的文件标识符集合，包含关键词 w_i 的文档数 $N_i=|ID_{w_i}|$。

(2)对于 $\forall w_i\in W(1\leqslant i\leqslant l)$

1)根据式(1)计算第 $j(1\leqslant j\leqslant N_i)$个包含关键词 w_i 的文档的相关分 RSC_{ij}；

2)采用 $OPME_{f_z(w_i)}$ 加密相关分 RSC_{ij}，其与第 j 个加密文档 C_j 的 ID 组成一个索引项：$\langle w_i\|ID(C_j)\|OPME_{f_z(w_i)}(RSC_{ij})\rangle$；

3) $I(w_i)=I(w_i)\cup\langle w_i\|ID(C_j)\|OPME_{f_z(w_i)}(RSC_{ij})\rangle$，$I(w_i)$为包含关键词 w_i 的索引项。

(3)用 AES 对称密钥 KP_D 加密关键词，$I'(w_i)=I(AESEncrypt(KP_D,w_i))$；

(4)输出安全索引 I'。

索引构建算法如算法 2 所示。

算法 2　密文倒排索引构建算法 Indexbuild

输入：文档集合 D，对称密钥 KPD，OPME 初始参数

输出：安全索引 I′

1. Indexbuild(D,KP_D,OPME)；
2. extract W=$\{w_1,w_2,\cdots,w_l\}$ from D；
3. for each w_i in W do
4. for each d_j in D do
5. if w_i in d_j then RSC_{ij}=rscore_compute(w_i,d_j)；/ * 计算相关分 * /
6. $I(w_i)=I(w_i)\cup\langle w_i\|ID(C_j)\|OPME_{f_z(w_i)}(RSC_{ij})\rangle$；
7. $I'(w_i)=I(AESEncrypt(KP_D,w_i))$；
8. return I′.

3　SICBF 密文索引

倒排索引加密后，需要在不暴露索引关键词及检索关键词隐私的情况下实现索引的快速检索。本文建立了计数布隆过滤器(CBF)来实现密文倒排索引的快速检索，称其为 SICBF(Security Index Based on Counting Bloom Filter)索引。CBF 是布隆过滤器的改进，它将布隆过滤器数组的每一位扩展为一个小的计数器[21-22]，使其不仅支持元素的插入和查找操作，还支持存储在过滤器中的元素的删除操作。

3.1　CBF 生成算法

密文索引字典中加密后的关键词失去了语义性，给检索带来了一定的难度。在密文索引上建立 CBF，可以在不暴露密文索引字典的前提下实现对密文索引的安全和快速检索。CBF 生成与查找算法的思想如下：

(1)$CBF=\langle C,H\rangle$为计数布隆过滤器。其中，C 表示长度为 m 的数组，C_j 为 C 中第 j 个位置上的计数器；$H=\{h_1,h_2,\cdots,h_k\}$为 k 个散列函数的集合，其值域均为$\{1,2,\cdots,m\}$，且 $\forall x\in W=\{w_1,w_2,\cdots,w_n\}$，$h_i(x)\in\{1,2,\cdots,m\}(1\leqslant i\leqslant k)$。

(2)元素 x 的插入操作。使用 k 个散列函数对 x 进行散列求值，其结果为$\{h_1(x),h_2(x),\cdots,h_k(x)\}$，其中，$h_i(x)(1\leqslant i\leqslant k)$为元素 x 在数组 C 中的下标索引。取最小的 $C_{h_i(x)}(1\leqslant i\leqslant k)$所在的索引项，将元素 x 加入链表 A^I 中。

(3)元素 x 的查找操作。取$\{C_{h_1(x)},C_{h_2(x)},\cdots,C_{h_k(x)}\}$的最小值并将其记为 $C_{\min}$，查找对应链表 $A^{C\min}$。查找算法见 4.2 节算法 7。

CBF 生成算法如算法 3 所示。

算法 3　CBF 生成算法 CBFBuild

输入：安全索引 I′

输出：CBF

1. CBFBuild(I′)；
2. for each w_i′ in I′ do
3. x=$\langle w_i',p\rangle$；/ * w_i'为加密关键词，p 为指向包含加密关键词 w_i'的倒排表指针 * /
4. CBFInsertPruned(x). / * 见算法 5 * /

图 2 给出了根据倒排索引建立的 CBF。其中，哈希表使用 3 个哈希函数 h_1,h_2,h_3，若对关键字 T_1 有 $h_1(T_1)=1$，$h_2(T_1)=3$，$h_3(T_1)=5$，则 CBF 中的计数器 C_1，C_3 和 C_5 分别加 1；同时，将 T_1 插入到链表 A^1，A^3 和 A^5 中。T_2，T_3，T_4 的插入操作同理。

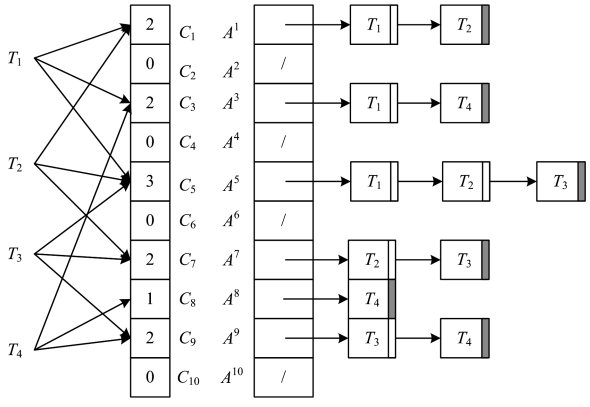

图 2　计数布隆过滤器

Fig. 2　Counting bloom filter

3.2 CBF 剪枝算法

相较于布隆过滤器，CBF 需要更多的存储空间。在图 2 中，当需要查找元素 T_1 时，计算 $h_1(T_1)=1$，$h_2(T_1)=3$，$h_3(T_1)=5$，然后在 CBF 中获取相应的计数器 C_1，C_3 和 C_5，其中 $C_1=2$，$C_3=2$，$C_5=3$，表示在哈希表的链表 A^1，A^3 和 A^5 中均包含元素 T_1，且其长度分别为 2，2 和 3。显然，有 $C_1=C_3<C_5$，因此，在链表 A^1 中查找元素 T_1。根据上述查找算法，将在链表 A^1 中查找元素 T_1，链表 A^3 和 A^5 中的拷贝将不会被查找，浪费了存储空间。因此，需要对 CBF 进行裁剪，将不被查找的拷贝从 CBF 中删除。CBF 剪枝算法的基本思想如下：

(1)对元素 x 求 k 个散列值，获取 x 的最小计数值 $C_{\min}=\min\{C_{h_1(x)},C_{h_2(x)},\cdots,C_{h_k(x)}\}$ 及对应的数组位置 I。

(2)根据元素 x 的 k 个散列值，获取 x 的 k 个拷贝所在链表 $A^{h_i(x)}(1\leqslant i\leqslant k)$；若 $h_i(x)\neq I$，删除 $A^{h_i(x)}$ 中的元素 x。

CBF 剪枝算法如算法 4 所示。

算法 4 CBF 剪枝算法 CBFPruned

输入：加密关键词集合 W

输出：剪枝后的 CBF

```
1. CBFPruned(W);
2. for each x in W do
3.    I=index of min{ ⋃_{i=1}^{k} C_{h_i(x)} };
4. for i=1 to k do
5.       if h_i(x) != I
6.        then A^{h_i(x)} = A^{h_i(x)} − x;
```

图 2 的 CBF 经过裁剪后的结果如图 3 所示。

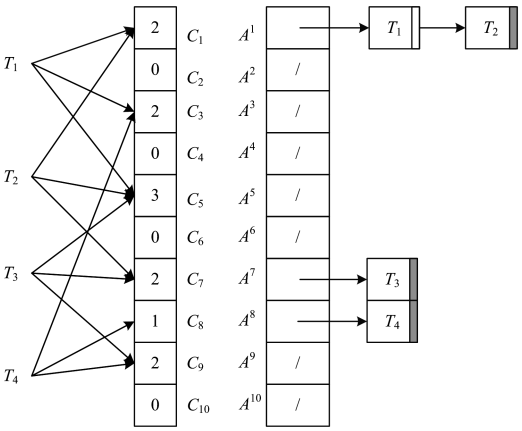

图 3 裁剪后的计数布隆过滤器

Fig. 3 Pruned counting bloom filter

3.3 CBF 插入算法

CBF 剪枝后，须对元素插入算法做特殊处理，否则会导致不正确的查找，因为 CBF 剪枝后，计数器的值不再反映链表中元素的个数。例如，图 3 中的元素 T_1 被映射到链表 $\{A^1,A^3,A^5\}$ 中，其对应计数器的值为 $\{2,2,3\}$。如果此时插入一个新元素 x，且与元素 T_1 在链表 A_1 发生冲突，则 $C_1=3$。在查找 T_1 时，由于 $\min\{C_1,C_3,C_5\}=\min\{3,2,3\}=C_3=2$，因此选择链表 A^3 进行查找。但是由于 CBF 剪枝后，A^3 中没有元素 T_1，导致查找失败。改进算法的思想如下：

(1)给定待插入元素 x，将其加入插入元素列表 *insertlist*。

(2)求 x 的 k 个散列值，结果为 $\{h_1(x),h_2(x),\cdots,h_k(x)\}$，将对应的 $C_{h_i(x)}$ $(1\leqslant i\leqslant k)$ 加 1，并将相应链表 $A^{h_i(x)}$ $(1\leqslant i\leqslant k)$ 中的元素全部加入 *insertlist* 列表中，然后将 $A^{h_i(x)}$ 置空。

(3)对于插入元素列表 *insertlist* 中的每个元素 x，求 $I=\min\{\bigcup_{i=1}^{k}C_{h_i(x)}\}$，并将其重新插入链表 A^I 中，插入时计数器不再加 1。

改进的元素插入算法如算法 5 所示。

算法 5 CBF 插入算法 CBFInsertPruned

输入：待插入元素 x

输出：CBF

```
1. CBFInsertPruned(x);
2. insertlist=insertlist∪x;
3.    for i=1 to k do
4.       insertlist=insertlist∪A^{h_i(x)};
5.       A^{h_i(x)}=∅;C_{h_i(x)}++;
6.    for each x in insertlist do
7.       I=index of min{ ⋃_{i=1}^{k} C_{h_i(x)} };
8.         A^I=A^I∪x.
```

4 SICBF 的工作机制

基于 SICBF 的关键词排序检索系统主要涉及 3 个实体：数据拥有者、授权用户和云服务器。数据拥有者将数据文件和索引加密处理后外包存储在云服务器中，以保护数据的隐私安全[23-25]。授权用户发送检索请求至云服务器，请求需要的文档集合。云服务器根据接收到的检索请求，查询安全索引，将排序好的检索结果 top-k 返回给授权用户。基于 SICBF 的关键词排序检索系统如图 4 所示，其工作过程主要包含初始化阶段和检索阶段。

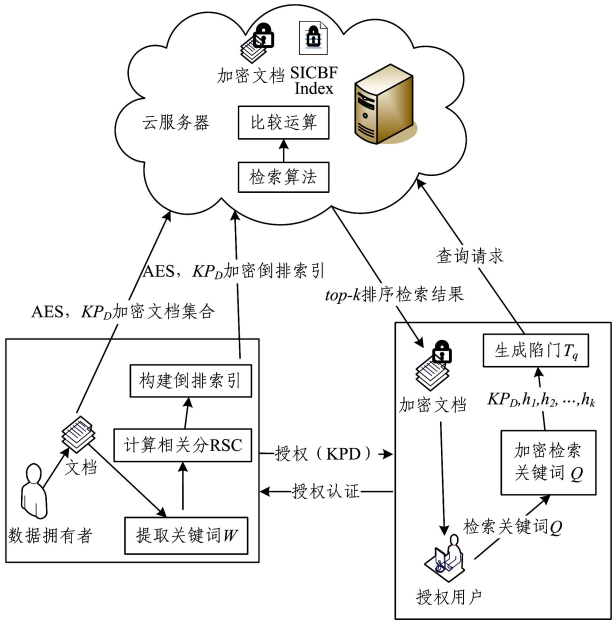

图 4 基于 SICBF 的多关键词排序检索系统

Fig. 4 Multi-keyword ranked retrieval system based on SCIBF

4.1 初始化阶段

对于待存储数据文档集 $D=\{D_1,D_2,\cdots,D_n\}$，在初始化阶段，数据拥有者建立明文倒排索引，生成加密密钥，加密明文倒排索引，主要步骤如下。

(1)从文档中提取关键词集合，建立倒排索引，根据式(1)计算倒排表中文档的相关分。

(2)生成 OPME 保序加密秘钥 K'，对相关分进行加密处理。

(3)生成 AES 加密密钥 KP_D，对明文文档、明文倒排索引关键字进行加密，并根据密文倒排索引建立 CBF。

(4)数据拥有者将加密文档集合、密文倒排索引以及 CBF 部署到云服务器中。

4.2 检索阶段

在密文检索阶段，为防止攻击者根据检索关键词推测检索结果中密文的内容，须根据检索关键词生成陷门，服务器通过陷门检索 SICBF 索引，主要步骤如下。

(1)用户随机化检索关键词 Q，生成陷门 T_Q，并向服务器提交 T_Q 和希望得到的最相关文档数 k。陷门生成算法如算法 6 所示。

算法 6　陷门生成算法

输入：检索语句 Q，秘钥 KP_D

输出：陷门 T_Q

1. Trapdoor(KP_D,Q)；
2. Q′=RK(Q)；
3. for each q_i in Q′ do
4. q_i'=AESEncrypt(KP_D,q_i)；
5. $h=\bigcup_{j=1}^{k} h_j(q_i')$；$T_Q=T_Q\cup h$；
6. return T_Q.

(2)服务器接收陷门 T_Q，根据 T_Q 查找 CBF，定位 CBF 链表位置。计算 CBF 链表中每个关键词的散列值，并将其与接收到的陷门 T_Q 中的散列值进行对比，如果两者相同则检索到关键词，从而得到该检索词的倒排表。索引查找算法如算法 7 所示。

算法 7　索引查找算法

输入：陷门 T_Q，最相关文档数 k，CBF 和安全索引 I′

输出：top-k 结果文档

1. SearchIndex(T_Q,k,CBF,I′)；
2. for each h in T_Q do
3. for each t in $A^{I'}$(h)
4. if $\bigcup_{j=1}^{k} h_j(t)$ == h
5. then ps=ps∪CBFSearchItem(t).p；
6. return ps.
7. MergeSort(ps,0,n)；/* 排序文档集合 */
8. return top-k. /* 返回最相关 top-k 文档集合 */

(3)根据密文相关分对查询得到的文档进行排序，将 top-k 检索文档返回给授权用户。

陷门生成算法的思想如下：

(1)给定检索语句 Q，通过算法 RK(·)将随机关键词添加至 Q，得到 Q'。RK(·)算法详见 5.2 节。

(2)对于 $\forall q_i\in Q'$，使用对称密钥 KP_D 对其进行加密，$q_i'=AESEncrypt(KP_D,q_i)$。

(3)求 q_i'的散列值 $h=\bigcup_{j=1}^{k} h_j(q_i')$。

(4)合并所有 q_i'的散列值，即 $T_Q=T_Q\cup h$。

SICBF 索引查找算法的思想如下：

(1)给定陷门 T_Q，k，CBF 和安全索引 I'，对 $\forall h\in T_Q$ 查找 CBF，定位 CBF 链表 $A^{I'}(h)$。

(2)对于 $\forall t\in A^{I'}(h)$，计算散列值 $\bigcup_{j=1}^{k} h_j(t)$。

(3)若 $\bigcup_{j=1}^{k} h_j(t)==h$，则查找 CBF 以获取 t 的指针 p，p 指向包含检索关键词的倒排表。

(4)$ps=ps\cup p$，ps 指向待排序的全部文档集合。

(5)调用归并排序算法 MergeSort(ps,0,n)。

1)将 ps 指向的 n 个待排序文档分解成两个子序列，每个子序列包含 $n/2$ 个元素；

2)根据加密相关分对每个子序列进行归并排序 MergeSort；

3)合并 Merge 排序后的子序列，生成排序文档。

(6)根据 k，返回排序后的 top-k 结果文档。

5 安全性分析

密文文档及索引机密性可由 AES 加密算法保证。本节重点分析了 OPME 算法的安全性及如何保护检索关键词的隐私。

5.1 OPME 的安全性分析

一对多的保序加密算法 OPME 是通过改进 OPSE 得到的，它引入了文件标识符 ID 作为随机种子的一部分，使得多个相同相关分不再确定性地映射到同一密文相关分，而是映射到范围 R 中的多个随机值上。由此可知，R 的值越大，原始相关分数据分布的峰值就会越少地被保存下来。因此，可以通过增大 R 的值，来增加攻击者确定值域 R 的映射值是否属于同一分数的难度。但 R 的值不能无限增大，因为过大的范围会降低 HGD(Hyper Geometric probability Distribution)函数的效率。可以根据信息理论中的最小熵来确定 R 的大小，如式(2)所示：

$$\frac{max/(|R|\cdot\frac{1}{2}^{5\log M+12})}{\lambda}\leqslant 2^{-(\log k)^c} \tag{2}$$

其中，max 表示索引 I 中相关分副本的可能最大值，λ 表示映射到每个索引列表 $I(w_i)$的相关分的平均值。假定定义域的取值为{1,2,…,M}，则 M 为定义域的长度。若 $|R|$ 以比特为单位进行表示，且 $k=\log|R|$，则整理式(2)得：

$$\frac{max\cdot 2^{5\log M+12}}{2^k\cdot\lambda}=\frac{max\cdot M^5}{2^{k-12}\cdot\lambda}\leqslant 2^{-(\log k)^c} \tag{3}$$

在初始化阶段建立索引的过程中，可以得到所有相关分明文中最大副本数的百分比 (max/λ)，因此可以根据式(3)确定合适的 $|R|$ 值。选择好合适的 $|R|$ 值后，经过 OPME 加密的相关分在索引中是一个有着低副本率的数值序列。攻击者可能通过密文副本推测部分明文副本存在的相关关系，但是相关分的随机化和高度平缓的一对多映射仍然使得攻击者很难预测原始明文相关分的分布情况。

通过以上分析可知，经过 OPME 加密后，相同的明文相关分会被映射到不同的密文相关分上，攻击者不能根据加密相关分预测明文相关分分布，更不可能根据相关分分布推测明文关键词，因此该方法保证了关键词的隐私安全。

5.2 关键词隐私

虽然本文对密文倒排索引字典中的关键词及查询关键词都进行了加密处理，但攻击者还是可以根据查询关键词出现的频率推测关键词的内容。因此，本文的密文检索要求根据用户查询关键词生成的陷门而具备不确定性，即使提交相同的检索关键词，生成的陷门也不一样，从而隐藏查询关键词出现的频率。

本文的陷门生成算法包括 3 部分：首先，向用户检索关键词集合中添加随机关键词，且要求添加的随机关键词不能出现在倒排索引中；然后，对得到的随机关键词集合进行加密处理；最后，对加密的关键词集合进行散列求值，从而得到最终的陷门。如图 5 所示，针对用户检索关键词集合 Q，通过算法 RK(·)向集合 Q 中添加随机关键词，得到的新关键词集合 Q'具有随机性，即两次通过算法 RK(·)向关键词集合 Q 中添加关键词后得到的关键词集合 Q_1'和 Q_2'不相同。算法 RK(·)取当前系统时间作为种子，生成随机关键词。其中，$\{t_1,t_2,\cdots,t_n\}$是算法 RK(·)生成的随机关键词，$RK:\{q_1,q_2,\cdots,q_k\}\xrightarrow{Time}\{q_1,q_2,\cdots,t_1,\cdots,t_2,q_i,\cdots,t_n,\cdots,q_k\}$。对于输入关键词集合 Q，$RK^{(n)}(Q)=RK(RK^{(n-1)}(Q))$，而 $Q'=RK(RK^{(n-1)}(Q))$。因而，经过 n 次迭代，生成的关键词集合 $Q'=\{q_1',q_2',\cdots,q_{n+k}'\}$是随机的。

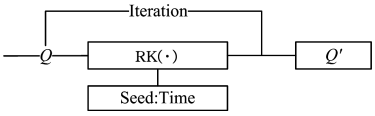

图 5 随机关键词的添加

Fig. 5 Adding of random keywords

对于关键词集合 Q'，对 $\forall q_i\in Q$ 进行加密，$q_i'=AESEncrypt(KP_D,q_i)$，求哈希值 $h=\bigcup_{j=1}^{k}h_j(q_i')$，$h_i(x)(1\leqslant i\leqslant k)$，其散列值具有不可逆性[26]。$h$ 即为密文关键词 q_i'对应的陷门。

综上，对于关键词集合 Q，因为算法 RK(·)随机添加关键词至关键词集合 Q 中，使得相同的查询关键词集合经过陷门生成算法也会得到不同的陷门 T_Q，从而使生成的陷门具有不确定性。因此，本文的陷门生成算法能够确保检索关键词的隐私安全性。

6 实验结果与分析

6.1 SICBF 索引的构建

为检验构建 SICBF 密文索引的时间消耗，以 RFC(Request For Comments)为数据集，选取 5000 个文本文件进行索引构建实验。实验环境为：Win7 旗舰版 64 位，Intel(R) Core(TM) i5-2410M 处理器(主频 2.3 GHz)，内存 4 GB。

相比于明文索引的构建，SICBF 密文索引构建的时间消耗还包括相关分加密的耗时和关键词加密的耗时。实验结果如表 1 所列，时间主要耗费在相关分加密和关键词加密上，且相关分加密的时间消耗较多。加密时间是一次性的，为保障相关分的隐私安全，并对密文查询结果进行排序，这种一次性加密的时间的消耗是可以接受的。

表 1 SICBF 索引构建的时间消耗

Table 1 Time comsumption of SICBF index

文档数量	构建明文索引/ms	RSC 加密/ms	关键词加密/ms	总时间/ms	相关分加密占总时间的比例/%
1000	8123	10289	114	18526	55.538
2000	16248	24632	132	41012	60.060
3000	21302	37678	143	59123	63.728
4000	30687	50247	156	81090	61.964
5000	42176	65396	164	107736	60.700

6.2 OPME 的加密相关分

为考查本文所采用的 OPME 算法的加密效果，通过实验将其与 OPSE 算法[27]进行对比。以 RFC 为实验数据集，选取 1000 个测试文档进行分词，提取到的关键词“Computer”的相关分分布如图 6 所示。

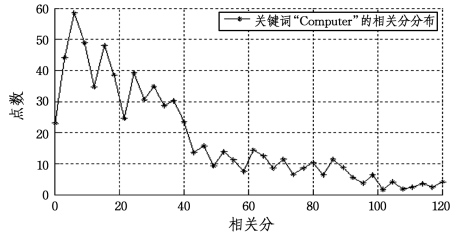

图 6 原始相关分分布

Fig. 6 Distribution of original relevance score

假设安全等级为 128，即 $M=128$，则 $D=\{1,2,\cdots,128\}$，$max/\lambda=0.06$，则根据式(3)可确定密文的映射范围 $|R|=2^{64}$。采用 OPSE 和 OPME 对相关分进行加密，则加密后的相关分分布如图 7 所示。

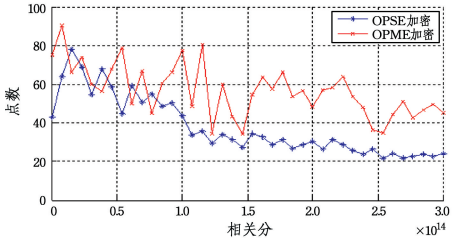

图 7 加密相关分分布

Fig. 7 Distribution of encrypted relevance score

6.3 SICBF 的检索效率

为验证 SICBF 索引的检索效率，以 RFC 为数据集，选取 5000 个文本文件，将其与 RSSE[6]和 MRSE[7]方案进行实验对比。RSSE 在相关分处理上与本文类似，对相关分进行保序加密，由服务器根据密文相关分完成检索结果的排序；MRSE 方案中采用向量表示每个文档，通过向量空间模型计算文档与检索关键词的匹配度。

RSSE 方案基于对称可搜索加密机制，对关键词陷门进行线性对比，因此需要对索引中大量的关键词进行顺序比较，从而使得索引的检索效率低。而对于基于 SICBF 索引的检索，云服务器根据陷门，通过哈希表直接查找 SICBF 索引，可

在固定常数时间内定位到 CBF 链表，通过散列值对比能快速查询到关键词相关的文档集合。实验结果如图 8 所示。

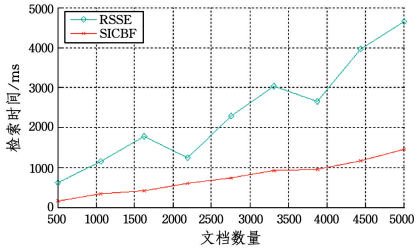

图 8　RSSE 和 SICBF 的检索效率

Fig. 8　Retrieval efficiency of RSSE and SICBF

MRSE 检索过程包括文档相似度计算、检索结果排序。每个文档由一个文档向量表示，查询关键词采用查询向量表示。随着文档和关键词数量的增加，文档向量的数量和维度都会增加，向量相似度的计算复杂度也将随之增加，时间消耗变大。SICBF 索引在构建时已经将相关分加密后保存在索引中，检索时只需通过陷门查找 SICBF 索引，可在常数时间内定位倒排表，快速查询到关键词相关的文档集合。实验结果如图 9 所示。

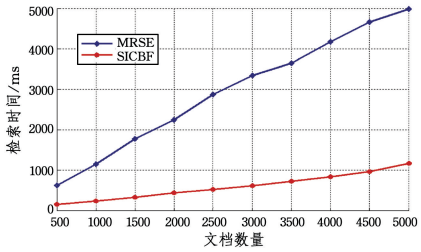

图 9　MRSE 和 SICBF 的检索效率

Fig. 9　Retrieval efficiency of MRSE and SICBF

因此，SICBF 对海量密文文档的检索效率高于 RSSE 和 MRSE，更适用于云存储环境下密文文档的快速安全检索。

结束语　本文对传统倒排索引结构进行了改造，提出了基于计数布隆过滤器的密文安全索引，以实现加密关键词的安全快速检索；同时，设计了计数布隆过滤器剪枝算法来减少索引数据的冗余。此外，本文还将一对多的保序映射加密算法 OPME 应用到相关分的加密技术中，通过 OPME 加密算法对相关分进行加密，使得加密后的相关分不仅隐藏了原始数据的分布，而且能够保存原始数据的有序性，使得云服务器可以在不暴露相关分信息的前提下，根据密文相关分对检索结果进行排序，并将排序后的文档发送给授权用户。本文提出的 SICBF 索引安全性高，检索速度快，适用于海量密文文档的快速检索。另外，SICBF 索引在检索时，计算工作在服务器端完成，客户端的计算量小，有利于手机、平板等资源受限的客户端快速检索密文文档。

参 考 文 献

[1] LI H, SUN W H, LI F H, 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] FENG D G, ZHANG M, ZHANG Y, et al. Study on Cloud Computing Security[J]. Journal of Software, 2011, 22(1): 71-83. ((in Chinese)
冯登国，张敏，张妍，等. 云计算安全研究[J]. 软件学报，2011，22(1)：71-83.

[3] SONG D X, WAGNER D, PERRIG A. Practical Techniques for Searches on Encrypted Data[C]// IEEE Symposium on Security & Privacy. 2002: 44-55.

[4] GOH E J. Secure Indexes[OL]. http://eprint.iacr.org/2003/216.

[5] DAN B, CRESCENZO G D, OSTROVSKY R, et al. Public Key Encryption with Keyword Search[M]// Advances in Cryptology-EUROCRYPT 2004. Springer Berlin Heidelberg, 2004: 506-522.

[6] WANG C, CAO N, LI J, et al. Secure Ranked Keyword Search over Encrypted Cloud Data[C]// IEEE International Conference on Distributed Computing Systems. IEEE, 2010: 253-262.

[7] CAO N, WANG C, LI M, et al. Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data[J]. IEEE Transactions on Parallel & Distributed Systems, 2014, 25(1): 222-233.

[8] XU Z, KANG W, LI R, et al. Efficient Multi-Keyword Ranked Query on Encrypted Data in the Cloud[C]// IEEE International Conference on Parallel and Distributed Systems. IEEE, 2012: 244-251.

[9] SUN W, WANG B, CAO N, et al. Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking[C]// ACM Symposium on Information, Computer and Communications Security. 2013: 71-82.

[10] FU Z, SUN X, XIA Z, et al. Multi-keyword ranked search supporting synonym query over encrypted data in cloud computing [C]// IEEE International PERFORMANCE Computing and Communications Conference. IEEE, 2013: 1-8.

[11] CHEN C, ZHU X, SHEN P, et al. An Efficient Privacy-Preserving Ranked Keyword Search Method[J]. IEEE Transactions on Parallel & Distributed Systems, 2016, 27(4): 951-963.

[12] ZERR S, OLMEDILLA D, NEJDL W, et al. Zerber +R: top-k retrieval from a confidential index[C]// Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology. ACM, 2009: 439-449.

[13] CUTTING D, PEDERSEN J. Optimization for dynamic inverted index maintenance[J]. International Acm Sigir Conference on Research & Development in Information Retrieval. 1989: 405-411.

[14] ZOBEL J, MOFFAT A, RAMAMOHANARAO K. Inverted files versus signature files for text indexing[J]. ACM Transactions on Database Systems, 1998, 23(4): 453-490.

[15] ZOBEL J, MOFFAT A. Inverted files for text search engines [J]. ACM Computing Surveys, 2006, 38(2): 1-56.

[16] MANWAR B, MAHALLE S, CHINCHKHEDE D, et al. A Vector Space Model for Information Retrieval: A Matlab Ap-

proach[J]. Indian Journal of Computer Science & Engineering, 2012,3(2):222-229.

[17] WANG C,CAO N,REN K,et al. Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data[J]. IEEE Transactions on Parallel & Distributed Systems,2012,23(23):1467-1479.

[18] AGRAWAL R,KIERNAN J,SRIKANT R,et al. Order preserving encryption for numeric data[C]// ACM SIGMOD International Conference on Management of Data. ACM, 2004: 563-574.

[19] BOLDYREVA A,CHENETTE N,LEE Y,et al. Order-Preserving Symmetric Encryption[M]// Advances in Cryptology — EUROCRYPT 2009. Cologne,Germany,2009:224-241.

[20] LESTER N,ZOBEL J,WILLIAMS H E. In-Place versus Re-Build versus Re-Merge: Index Maintenance Strategies for Text Retrieval Systems[C]// 27th Australasian Computer Science Conference. 2004:15-22.

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

[22] BONOMI F,MITZENMACHER M,PANIGRAHY R,et al. An Improved Construction for Counting Bloom Filters[C]// Conference on European Symposium. Zurich,Switzerland,2006:684-695.

[23] YONG H H,LEE P J. Public Key Encryption with Conjunctive Keyword Search and Its Extension to a Multi-user System[C]// International Conference on Pairing-Based Cryptography. Springer-Verlag,2007:2-22.

[24] LEWKO A,OKAMOTO T,SAHAI A,et al. Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption[M]// Advances in Cryptology — EUROCRYPT 2010. French Riviera,2010:62-91.

[25] BALLARD L,KAMARA S,MONROSE F. Achieving Efficient Conjunctive Keyword Searches over Encrypted Data[C]// International Conference on Information and Communications Security(ICICS 2005). Beijing,China,2005:414-426.

[26] WANG S,SHAN P. Security analysis of a one-way hash function based on spatio temporal chaos[J]. Chinese Physics B, 2011,20(9):79-85.

[27] BOLDYREVA A,CHENETTE N,LEE Y,et al. Order-Preserving Symmetric Encryption[M]// Advances in Cryptology — EUROCRYPT 2009. Cologne,Germany,2009:224-241.

(上接第 122 页)

[17] LIU D. Practical Fully Homomorphic Encryption without Noise Reduction [EB/OL]. [2016-12-15]. http://eprint. iacr. org/2015/468. pdf.

[18] NAEHRIG M,LAUTER K,VAIKUNTANATHAN V. Can homomorphic encryption be practical? [C]// Proc. of the 3rd ACM workshop on Cloud computing security workshop. New York:ACM,2011:113-124.

[19] GJØSTEEN K,STRAND M. Fully homomorphic encryption must be fat or ugly? [EB/OL]. [2016-12-15]. http://eprint. iacr. org/2016/105. pdf.

[20] LI M,CAO N,YU S,et al. Findu: Privacy-preserving personal profile matching in mobile social networks[C]// Proc. of INFOCOM 2011. Piscataway,NJ:IEEE,2011:2435-2443.

[21] RAHULAMATHAVAN Y,PHAN R C W,VELURU S,et al. Privacy-preserving multi-class support vector machine for outsourcing the data classification in cloud[J]. IEEE Transactions on Dependable and Secure Computing,2014,11(5):467-479.

[22] REWADKAR D N,GHATAGE S Y. Cloud storage system enabling secure privacy preserving third party audit[C]// Proc. of IntConf on Control,Instrumentation,Communication and Computational Technologies. Piscataway,NJ:IEEE,2014:695-699.

[23] LIU X,DENG R,CHOO K K R,et al. An Efficient Privacy-Preserving Outsourced Calculation Toolkits with Multiple Keys [J]. IEEE Transactions on Information Forensics & Security, 2016,11(11):2401-2414.

[24] LIU X,CHOO R,DENG R,et al. Efficient and Privacy-Preserving Outsourced Calculation of Rational Numbers[J]. IEEE Transactions on Dependable and Secure Computing, 2016, PP (99):1-1.

[25] CHEON J H,KIM A,KIM M,et al. Floating-Point Homomorphic Encryption [EB/OL]. [2016-12-15]. http://eprint. iacr. org/2016/421. pdf.

[26] ARITA S,NAKASATO S. Fully Homomorphic Encryption for Point Numbers [EB/OL]. [2016-12-15]. http://eprint. iacr. org/2016/402. pdf.

[27] COSTACHE A,SMART N P,VIVEK S,et al. Fixed point arithmetic in she schemes [EB/OL]. [2016-12-15]. http://eprint. iacr. org/2016/250. pdf.

[28] ARMKNECHT F,BOYD C,CARR C,et al. A guide to fully homomorphic encryption [EB/OL]. [2016-12-15]. http://eprint. iacr. org/2015/1192. pdf.

[29] HOWGRAVE-GRAHAM N. Approximate integer common divisors[C]// Proc. of the Int Conf on Cryptography and Lattices. Berlin:Springer,2001:51-66.

[30] ZAHARIA M,CHOWDHURY M,FRANKLIN M J,et al. Spark: cluster computing with working sets[C]// Proc of the 2nd USENIX Workshop on Hot Topics in Cloud Computing. Berkeley,CA:USENIX,2010:10.

[31] ZAHARIA M,CHOWDHURY M,DAS T,et al. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing[C]// Proc. of the 9th USENIX Symp. on Networked Systems Design and Implementation. Berkeley,CA: USENIX,2012:15-28.