doi:10.11959/j.issn.1000-436x.2017240

支持安全共享的云存储系统研究

宋衍[1,2]，韩臻[1]，李建军[2]，韩磊[1,2]
(1. 北京交通大学智能交通数据安全与隐私保护技术北京市重点实验室，北京 100044；2. 信息保障技术重点实验室，北京 100072)

摘 要：针对云存储的数据安全共享及效率问题，提出一种实用的云存储系统方案。方案使用对称加密和属性加密，融入代理加密功能，实现加密保护、访问控制和高效检索的无缝结合。基于双线性映射和多项式方程技术，实现连接关键词非域子集搜索。基于外包解密技术，将原属于用户完成的部分解密工作转移到云端服务器完成，降低解密带来的计算开销。对方案的性能进行了分析，并通过实验验证。

关键词：云存储；属性加密；可搜索加密；解密外包
中图分类号：TP309.7 **文献标识码**：A

Research on cloud storage systems supporting secure sharing

SONG Yan[1,2], HAN Zhen[1], LI Jian-jun[2], HAN Lei[1,2]
(1. Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing 100044, China;
2. Science and Technology on Information Assurance Laboratory, Beijing 100072, China)

Abstract: A practical scheme for the cloud storage system was proposed to ensure security and efficiency during data sharing. The scheme which combine attribute-based encryption, proxy encryption with symmetric encryption, could integrate access control, efficient search with encryption well together. The scheme archived conjunctive-keyword non-field subset search by using bilinear mapping and polynomial equation. The scheme reduced the cost of decryption by outsourcing most of the decryption operations from the terminal to the cloud. At last, the performance was analyzed and an experiment was made for verification.

Key words: cloud storage, attribute-base encryption, searchable encryption, decryption outsourcing

1 引言

云存储的数据异地性、环境开放性使云存储系统并不可信。通过客户端加密保证数据的机密性后，解决数据密文的访问控制和高效检索，成为云存储实现安全共享的关键问题。现有的成熟云存储服务大都采用传统密码体制来加密数据，如 Amazon S3、Windows Azure Storage Service 和阿里云 OSS 等都采用 AES 算法，用户可以选择服务器端加密还是客户端加密。Fu 等[1]基于客户端加密并保存根密钥的思想，使用传统密码体制设计了一种云存储的安全网盘系统，通过三层密钥管理来提高数据存储与共享过程中的安全性与高效性，通过目录树来支持文件夹级的数据共享。Xue 等[2]基于传统密码体制和三层密钥结构设计了一种安全云存储系统，通过访问控制块和可信验证服务器实现授权访问。上述云存储系统都不支持文件内容的搜索，同时，使用传统加密方式虽然加解密速率较高，但在实现多用户细粒度访问控制时，密钥管理较为复杂。

为了解决多用户细粒度访问控制问题，Sahai 和 Waters 于 2005 年欧密会上首次提出了属性加密[3]的概念，通过为身份配置细粒度的属性来实现更加精确的访问控制，通过双线性对技术实现更加灵活的

收稿日期：2017-09-08
基金项目：国家自然科学基金资助项目（No.61502030, No.61672092）；信息保障技术重点实验室创新基金资助项目（No.61421120401）
Foundation Items: The National Natural Science Foundation of China (No.61502030, No.61672092); Innovation Foundation of Science and Technology on Information Assurance Laboratory (No.61421120401)

密码学运算。最初的 ABE 机制虽然很好地改善了细粒度控制和工作效率，但是仅支持门限访问控制操作。因此，学者进一步提出了密钥策略（key-policy）[4]和密文策略[5]（ciphertext-policy）的 ABE 机制，实现了属性的与、或、非和门限操作，从而支持灵活的访问控制策略。针对算法开销的问题，Attrapadung 等[6]设计了 2 种方案：1) CP-ABE 方案在门限访问控制结构上具有固定的密文长度，而私钥长度未增加；2) KP-ABE 方案在非单调的访问策略结构中具有较短的密文长度。Green 等[7]提出了一种支持外包解密的 ABE 方案，先由服务器利用代理密钥对密文完成绝大部分的解密运算，再由终端完成密文的最终解密，从而降低本地的解密运算量和通信量。

为了解决不解密而进行搜索的难题，Song 等[8]提出了实用的对称加密方案，每项检索耗费的工作时间是线性的。Boneh 等[9]利用公钥密码算法构造了一个可搜索加密方案（PEKS），解决了对其他用户的加密数据进行检索的困难问题。鉴于一次搜索多个关键词能够缩小搜索范围改进搜索性能[10]，Golle 等[11]提出了关键词连接搜索的概念和安全模型，给出了 2 种连接关键词搜索方案；但是，方案基于对称密钥加密和搜索关键词，应用环境受限。Park 等[12]基于双线性映射提出了一种公钥密码系统连接关键词搜索方案（PKCKS）。Hwang 和 Lee[13]也基于双线性映射设计了 PKCKS 方案，并且将方案扩展到多用户环境；但是，采用多对公私钥的形式，给密钥管理带来较大负担。Zhang 等[14]提出支持连接关键词子集搜索的 PKCKS 方案，但是用户计算量较大。

总地来说，将属性加密和可搜索加密运用于云存储系统，已经成为云存储安全的重要发展方向。但是，目前学术界在属性加密和可搜索加密方面提出的算法方案，都只是解决了密文访问控制或密文数据搜索方面的某些问题，无法同时满足云存储对安全、高效共享的需要。本文基于属性加密和可搜索加密在双线性对技术使用上的相通性，提出一种同时实现访问控制、解密外包和关键词搜索的云存储系统算法方案。相比于同时使用属性加密和可搜索加密来实现安全共享，本文方案在安全上实现不同机制的无缝结合，在效率上有较大提高，能够满足云存储环境对多用户细粒度访问控制、关键词非域子集搜索、客户端运算量和通信量较小的要求。

2　系统模型

系统模型如图 1 所示，包括 4 个参与角色，记为{UM, DSP, DO, DReq}。其中，UM 是统一身份和密码管理机构，负责系统的初始化，并管理用户的属性，根据用户属性为用户生成私钥。DSP 是云端的服务提供者，提供文档存储、关键词搜索和外包解密服务。DO 是数据拥有者，将自己的文档加密后交由 DSP 存储和管理。DReq 是数据请求者，向 DSP 提交自己的数据意向。

本文将 DReq 发送给 DSP 用于搜索和外包解密的数据称为凭证（token），包括 3 个部分：1) 用于验证 DReq 是否满足密文访问控制策略的部分信息，称为凭证的权限验证部分；2) 用于对称密钥密文外包解密的部分信息，称为凭证的外包解密部分；3) 用于关键词搜索的部分信息，称为凭证的搜索部分或陷门。

DO 上传到 DSP 处的密文包括 4 个部分：1) 文档的密文，称为文档密文部分；2) 加密文档的对称密钥的密文，称为对称密钥密文部分；3) 代表该密文访问控制策略的部分信息，称为权限验证部分；4) 文档的关键词集合的密文，称为关键词密文部分。

DReq 的解密密钥分为 3 种：1) UM 基于用户属性集合制定的用户属性私钥；2) 对称密钥，用于解密文档；3) 在 DSP 外包解密对称密钥密文形成半明文后，DReq 在终端进行解密时所用的用户终端解密密钥。

一个支持安全共享的云存储系统方案由以下几种多项式时间算法组成。

1) Setup(1^{λ})→(pm,mk): UM 执行该算法以初始化系统；输入一个安全参数 1^{λ}，输出一个公开参数 pm 和一个主私钥 mk。

2) KeyGen($Atts$,mk,pm)→sk: UM 根据 DReq 提供的属性集为其生成相应的用户私钥；输入公开参数 pm、主私钥 mk 和 DReq 的属性集 $Atts$，输出对应的私钥 sk。

3) DocED(yk,doc)→cph_{doc}: 有云存储需求的 DO 用户使用对称算法对文档进行加解密；输入文档明文 doc 和对称密钥 yk，输出文档密文 cph_{doc}；输入文档密文 cph_{doc} 和对称密钥 yk，输出文档明文 doc。

4) Share(pm,q,T)→cph_{pri}：DO 根据访问控制策略形成访问控制树，并实施秘密共享；输入随机数

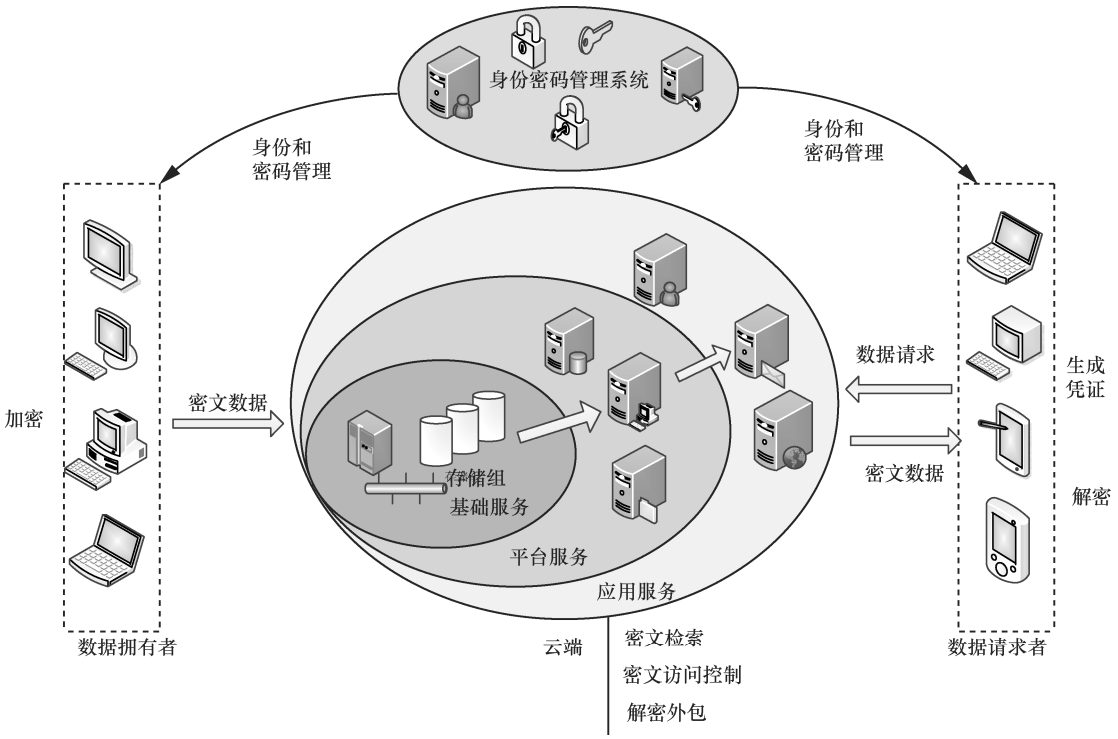


图 1 支持安全共享的云存储系统

q 作为待共享的值，以及访问控制树 T，计算 T 中每个叶子节点的共享值，输出密文的权限部分 cph_{pri}。

5) KeyEnc(pm,q,yk)→cph_{yk}：DO 使用属性加密算法加密对称密钥；输入公开参数 pm、共享值 q 和对称密钥 yk，输出对称密钥密文部分 cph_{yk}。

6) WordEnc(pm,q,W)→cph_{word}：DO 使用可搜索算法加密关键词集合；输入公开参数 pm、共享值 q 和关键词集合 W，输出关键词密文部分 cph_{word}。

7) PriGen(sk)→tk_{pri},dk：DReq 调用该算法计算凭证中的权限部分；输入 DReq 的私钥 sk，输出用户解密密钥 dk 和凭证的权限部分 tk_{pri}。

8) DecGen(sk,s)→tk_{dec},dk：DReq 有搜索需求时，调用该算法计算凭证中的外包解密密钥；输入 DReq 的私钥 sk 和随机数 s，输出凭证的外包解密部分 tk_{dec}。

9) TrapGen($Atts$,dk,pm,W',s)→tk_{sch}：DReq 有搜索需求时，构造关键词集合，计算关键词集合对应的陷门。输入公开参数 pm、用户属性集合 $Atts$、用户解密密钥 dk、关键词集合 W'，输出搜索部分的组件 H_i，构造关键词陷门 tk_{sch}。

10) Combine(cph_{pri},tk_{pri})→E_{root}：DSP 验证用户 DReq 是否满足密文的访问控制策略；输入密文的权限部分 cph_{pri} 和凭证的权限部分 tk_{pri}，如果满足访问控制策略，返回正确的秘密共享值 E_{root}。

11) Search(cph_{word},tk_{sch},E_{root})→1：DSP 验证密文是否满足陷门的搜索策略；输入秘密共享值 E_{root}、凭证的搜索部分 tk_{sch} 和密文的陷门部分 cph_{word}，如果密文满足陷门的搜索策略，则返回 1，否则，返回 0。

12) DSPDec(cph_{yk},tk_{dec},E_{root})→M_{part}：DSP 对对称密钥的密文进行外包解密；输入密文的对称密钥密文部分 cph_{yk}、凭证的外包解密部分 tk_{dec} 和秘密共享值 E_{root}，输出对称密钥的半明文。

13) DReqDec (dk,C_1,YK_{part})→yk：DReq 对对称密钥的半明文进行用户终端解密。输入密文组件 C_1、半明文 YK_{part} 及其相应的用户解密密钥 dk，输出最终的对称密钥明文 yk。

其中，文档加解密算法 DocED 为公开的标准算法，如 3DES 或 AES 等。

3 算法方案

3.1 访问控制

访问树的内部节点代表门（关系），包括与（and）、或（or）和门限（threshold）；叶子节点代表属性。假设 num_v 为节点 v 下子节点的个数，k_v

为节点 v 的门限值，则有 $1\leqslant k_v\leqslant num_v$。2 种特别的情况是，当 $k_v=1$ 时，节点 v 代表或门；当 $k_v=num_v$ 时，节点 v 代表与门。使用 $Parent(v)$表示节点 v 的父节点，$ind(v)$表示节点 v 在父节点下的索引号，$lvs(T)$表示访问树 T 中所有叶子节点组成的集合，$att(v)$表示叶子节点 v 所代表的属性，T_v 表示 T 中根为 v 的子树。

给定一个属性表达式集合 $Atts$，$F(Atts,T_v)=1$ 表示 $Atts$ 满足树 T_v 代表的访问控制策略，$F(Atts,T_v)=0$ 表示 $Atts$ 不满足树 T_v 代表的访问控制策略。那么 $F(Atts,T_v)$的取值可以通过以下方式递归确定。

1) 当 v 为叶子节点时，如果 $att(v)\in Atts$，则设置 $F(Atts,T_v)=1$；否则，$F(Atts,T_v)=0$。

2) 当 v 为内部节点时，假设 $v_1,v_2,\cdots,v_{num}$ 为 v 的子节点，如果存在一个子集 $I\subseteq\{1,\cdots,num_v\}$ 使 $|I|\geqslant k_v$，并且对于$\forall j\in I$，有 $F(Atts,T_{v_j})=1$，那么设置 $F(Atts,T_v)=1$；否则，设置 $F(Atts,T_v)=0$。

对于访问树 T，本文将 T 的秘密共享算法表示为

$$\{q_v(0)\mid v\in lvs(T)\}\leftarrow Share(T,q) \tag{1}$$

算法从上至下为 T 中的每个内部节点 v 构造一个 k_v-1 次一元多项式 q_v，并为每个内部节点和叶子节点赋值。

1) 如果 v 是 T 的根节点，设置 $q_v(0)=q$，并为多项式 q_v 随机选取 k_v-1 个系数。

2) 如果 v 是 T 中除根节点外的其他内部节点，设置 $q_v(0)=q_{parent(v)}(ind(v))$，并为多项式 q_v 随机选取 k_v-1 个系数。

3) 如果 v 是 T 的叶子节点，设置 $q_v(0)=q_{parent(v)}(ind(v))$。

给定访问树 T 和一个秘密值集合 $\{E_{u_1},E_{u_2},\cdots,E_{u_m}\}$，其中，$u_1,u_2,\cdots,u_m$ 为 T 的叶子节点，则满足 $F(att(u_1),att(u_2),\cdots,att(u_m),T)=1$。对于$\forall j\in\{1,2,\cdots,m\}$，本文定义 $E_{u_j}=e(g_1,g_2)^{q_{u_j}(0)}$，并且将 $e(g_1,g_2)^q$ 的重构算法表示为

$$e(g_1,g_2)^q\leftarrow Combine\left(T,\left\{E_{u_1},E_{u_2},\cdots,E_{u_m}\right\}\right) \tag{2}$$

根据访问树 T 的结构，算法从底向上执行如下步骤。

对于节点 v，如果 $F(att(u_1),att(u_2),\cdots,att(u_m),T_v)=0$，则继续；否则，$F(att(u_1),att(u_2),\cdots,att(u_m),T_v)=1$，如果 v 是叶子节点，则存在一个 u_j，使 $u_j=v$；设置 $E_v=E_{u_j}(0)=e(g_1,g_2)^{q_{u_j}(0)}$；否则，$v$ 是内部节点；那么，对于 v 的子节点集合$\left\{v_1,v_2,\cdots,E_{num_v}\right\}$，存在一个子节点索引的集合 S，使$|S|=k_v$，并且对于$\forall j\in S$，有 $F(att(u_1),att(u_2),\cdots,att(u_m),T_{v_j})=1$；使用 $\triangle_{u_j}=\prod_{l\in s,l\neq j}\frac{-j}{l-j}$ 表示系数，则 $E_v=\prod_{l\in S}E_{v_j}^{\triangle_{v_j}}=\prod_{l\in S}\left(e(g_1,g_2)^{q_{v_j}(0)}\right)^{\triangle_{v_j}(0)}=e(g_1,g_2)^{q_v(0)}$。

当算法结束时，得到 T 的根节点的秘密值 $E_{root}=e(g_1,g_2)^{q_{root}(0)}=e(g_1,g_2)^q$。

3.2　算法设计

DO 可以任意设置文档的关键词集合，没有域的划分，没有关键词个数、位置的限制。假设 $H:\{0,1\}^*\rightarrow Z_p$ 是一个安全的单向散列函数，将位串随机的映射到群 Z_p 中。方案算法的详细描述如下。

1) Setup(1^λ)。运行算法 $\mathcal{G}$ (1^λ)获得(p,G,G_T,e)。随机选取 $b,c\in Z_p$，发布公开参数为

$$pm=(p,G_1,G_2,G_T,e,g_1,g_2,g_1^{\beta},e(g_1,g_2)^{\alpha},g_1^{b},g_1^{c},g_1^{d},g_2^{c},g_2^{d}) \tag{3}$$

保存主私钥为

$$mk=(\alpha,\beta,b,c,d) \tag{4}$$

2) KeyGen($Atts,mk,pm$)。构造属性集 $Atts$ 对应的私钥。随机选择 $r\in Z_p^*$，计算 $K_0=g_2^{\frac{\alpha+r}{\beta}}$，$K_1=g_2^{\frac{bc-r}{d}}$。对于每个 $att_j\in Atts$，随机选择 $r_j\in Z_p^*$，计算 $A_j=g_2^{r}g_2^{H(att_j)r_j}$，$B_j=g_2^{r_j}$。构造私钥

$$sk=(Atts,K_0,K_1,\{(A_j,B_j)\mid att_j\in Atts\}) \tag{5}$$

3) DocED(yk,doc)。使用对称算法对文档明文 doc（文档密文）进行加（解）密，得到文档密文 cph_{doc}（文档明文）。

4) Share(pm,q,T)。对访问控制树实施秘密共享。随机选取 $q\in Z_p^*$ 作为访问树 T 的共享值，执行$\{q_v(0)\mid v\in lvs(T)\}\leftarrow Share(T,q)$，使 T 中的每个叶子节点得到一个关于 q 的共享值，并为每个叶子节点计算 $C_v=g_1^{q_v(0)}$ 和 $D_v=g_1^{H(att(v))q_v(0)}$。构造密文的权限验证部分为

$$chp_{pri}=\left(T,\left\{\left(C_v,D_v\right)\mid v\in lvs(T)\right\}\right) \tag{6}$$

5) KeyEnc(*pm*,*q*,*yk*)。使用属性加密算法加密对称密钥。计算 $C_0=g_1^{\beta q}$ 和 $C_1=yke\left(g_1,g_2\right)^{\alpha q}$，构造对称密钥密文部分为

$$cph_{yk}=(C_0,C_1) \tag{7}$$

6) WordEnc(*pm*,*q*,*W*)。使用可搜索算法加密关键词集合 $W=\{w_1,\cdots,w_m\}$。随机选择 $a,k\in Z_p$，构造一个 m 次多项式

$$\begin{aligned} f(x)&=a\left(x-H\left(w_1\right)\right)\left(x-H\left(w_2\right)\right)\cdots\left(x-H\left(w_m\right)\right)+k\\ &=a_m x^m+\cdots+a_1x+a_0 \end{aligned} \tag{8}$$

对于每个 $i\in\{0,1,\cdots,m\}$，计算 $F_i=g_1^{ca_i}$，并计算 $W_0=g_1^{bq}g_1^{dk}$ 和 $W_1=g_1^{dq}$。构造的关键词密文部分为

$$cph_{word}=\left(W_0,W_1,\left\{F_i\mid i\in\{0,1,\cdots,m\}\right\}\right) \tag{9}$$

形成密文为

$$chp=\left(cph_{doc},cph_{pri},cph_{yk},cph_{word}\right) \tag{10}$$

7) PriGen(*sk*)。计算凭证的权限验证部分 tk_{pri}。随机选取 $s\in Z_p^*$，设置用户解密密钥 $dk=s$。对于每个 $att_j\in Atts$，计算 $A_j'=A_j^s$，$B_j'=B_j^s$。构造凭证的权限部分为

$$tk_{pri}=\left(Atts,\left\{\left(A_j',B_j'\right)\mid att_j\in Atts\right\}\right) \tag{11}$$

8) DecGen(*sk*,*s*)。计算凭证的外包解密部分 tk_{dec}。根据随机数 s 计算 $T_0=K_0^s$，构造凭证的外包解密部分 $tk_{dec}=(T_0)$。

9) TrapGen(*Atts*,*pm*,*W*′,*s*)。构造凭证的搜索部分 tk_{sch}，即关键词集合 $W'=\{w_1',\cdots,w_t'\}$ 对应的陷门，其中，$t\leqslant m$。根据随机数 s，计算 $T_1=K_1^s$，$T_2=g^{cs}$。对于每个 $i\in\{0,1,\cdots,m\}$，计算 $H_i=g_2^{ds\frac{H(w_1')^i+\cdots+H(w_t')^i}{t}}$。构造关键词陷门为

$$tk_{sch}=\left(T_1,T_2,\left\{H_i\mid i\in\{0,1,\cdots,m\}\right\}\right) \tag{12}$$

10) Combine(cph_{pri},tk_{pri})。验证用户 DReq 是否满足密文的访问控制策略。根据密文中的访问树 T 和陷门 tk 中的属性集 *Atts*，服务器从 *Atts* 中选择一个满足 T 的属性子集 S。如果 S 不存在，则返回 0；否则，对于每个 $att_j\in S$，找到与之相应的 $att(v)=att_j$，计算 T 中叶子节点的秘密值为

$$E_v=\frac{e\left(A_j',C_v\right)}{e\left(B_j',D_v\right)}=e\left(g_1,g_2\right)^{rsq_v(0)} \tag{13}$$

然后计算根节点的秘密值为

$$\begin{aligned} E_{root}&=\mathrm{Combine}(T,\{E_v\mid att_v\in S\})\\ &=e(g_1,g_2)^{rsq_{root}(0)}=e\left(g_1,g_2\right)^{rqs} \end{aligned} \tag{14}$$

11) Search(cph_{word},tk_{sch},E_{root})。当 DReq 的属性集合满足密文的访问控制策略时，通过运算验证密文是否满足陷门的搜索策略。根据计算得到的秘密共享值 E_{root}，输入陷门的搜索部分和密文的关键词集合部分，如果 $e\left(W_0,T_2\right)=e\left(W_1,T_1\right)E_{root}\prod_{i=0}^{m}e\left(F_i,H_i\right)$，则返回 1，否则，返回 0。

12) DSPDec(cph_{yk},tk_{dec},E_{root})。当 DReq 满足密文的访问控制策略，密文满足 DReq 的搜索策略时，对对称密钥的密文进行外包解密。根据计算得到的秘密共享值 E_{root}，输入陷门的搜索部分和密文的关键词集合部分，计算对称密钥的半明文为

$$YK_{part}=E_{root}/_e(T_0,C_0)=e(g_1,g_2)^{-\alpha qs} \tag{15}$$

DSP 将 C_1 和 YK_{part} 这 2 个部分，连同文档密文 cph_{doc} 一起返回给 DReq 进行解密。

13) DReqDec (dk,C_1,YK_{part})。当 DSP 返回密文时，对对称密钥的半明文进行用户终端解密。DReq 根据密文组件 C_1、半明文 M_{part} 及其相应的用户解密密钥 dk 计算最终的对称密钥明文为

$$yk=C_1\left(YK_{part}\right)^{\frac{1}{dk}} \tag{16}$$

3.3 运算过程

根据第 3.2 节权限验证步骤 10)，权限验证阶段的计算过程分为 2 步，第一步计算叶子节点的秘密值，第二步根据秘密值重构算法计算访问树根节点的秘密值。在第一步中，如果 DReq 满足叶子节点 v 代表的属性权限，则可计算叶子节点的秘密值为

$$\begin{aligned} E_v&=\frac{e\left(A_j',C_v\right)}{e\left(B_j',D_v\right)}\\ &=\frac{e\left(g_2^{rs}g_2^{H\left(att_j\right)r_js},g_1^{q_v(0)}\right)}{e\left(g_2^{r_js},g_1^{H\left(att(v)\right)q_v(0)}\right)}\\ &=\frac{e\left(g_1,g_2\right)^{rsq_v(0)}e\left(g_1,g_2\right)^{H\left(att_j\right)r_jsq_v(0)}}{e\left(g_1,g_2\right)^{H\left(att_j\right)r_jsq_v(0)}}\\ &=e\left(g_1,g_2\right)^{rsq_v(0)} \end{aligned} \tag{17}$$

在第二步中，当用户的属性集合满足密文的访问控制策略，则可根据秘密值重构算法，利用叶子

节点的秘密值重构得到 T 的根节点的秘密值 $E_{root}=e(g_1,g_2)^{rsq}$。

完成权限验证后，DSP 根据 DReq 用户的关键词陷门对满足权限的密文进行搜索，利用陷门组件与密文组件做如下的双线性映射运算。

$$e(W_0,tok_0)=e\left(g_1^{bq}g_1^{dk},g_2^{cs}\right)=e(g_1,g_2)^{bqcs+cdks} \tag{18}$$

$$e(W_1,tok_1)=e\left(g_1^{dq},g_2^{(bc-r)s/d}\right)=e(g_1,g_2)^{bqcs-rqs} \tag{19}$$

$$\prod_{i=0}^{m}e(F_i,H_i)=\prod_{i=0}^{m}e\left(g_1^{ca_i},g_2^{ds\frac{H(w_1')^i+\cdots+H(w_t')^i}{t}}\right)$$

$$=\prod_{i=1}^{t}e(g_1,g_2)^{cds\frac{a_mH(w_i')^m+\cdots+a_1H(w_i')+a_0}{t}} \tag{20}$$

根据第 3.3 节可搜索算法步骤 6)中构造的 m 次多项式 $f(x)=a_mx^m+\cdots+a_1x+a_0$ 可知，当 DReq 搜索的关键词 w_i' 属于密文的关键词集合，即 $w_i'\in W$ 时，多项式计算 $a_mH(w_i')^m+\cdots+a_1H(w_i')+a_0=k$。因此，当陷门的关键词集合包含于文档的关键词集合，即 $W'\subseteq W$ 时，有 $\sum_{i=1}^{t}\frac{a_mH(w_i')^m+\cdots+a_1H(w_i')+a_0}{t}=k$，则能够运算得到 $\prod_{i=0}^{m}e(F_i,H_i)=e(g_1,g_2)^{cdsk}$。因此有

$$\frac{e(W_0,tok_0)}{e(W_1,tok_1)E_{root}\prod_{i=0}^{m}e(F_i,H_i)}$$

$$=\frac{e(g_1,g_2)^{bqcs+cdks}}{e(g_1,g_2)^{bqcs-rqs}e(g_1,g_2)^{rqs}e(g_1,g_2)^{cdsk}}$$

$$=1 \tag{21}$$

从而说明，当 DReq 满足密文的访问控制策略，并且密文满足 DReq 的搜索策略时，有 $(W_0,tok_0)=e(W_1,tok_1)E_{root}\prod_{i=0}^{m}e(F_i,H_i)$。

对于满足搜索条件的密文，DSP 对其对称密钥密文进行外包解密，输出半明文，运算过程为

$$M_{part}=\frac{E_{root}}{e(AK_0,C_0)}=\frac{e(g_1,g_2)^{rqs}}{e\left(g_2^{\frac{\alpha+r}{\beta}s},g_1^{\beta q}\right)}$$

$$=\frac{e(g_1,g_2)^{rqs}}{e(g_1,g_2)^{(\alpha+r)qs}}=e(g_1,g_2)^{-\alpha qs} \tag{22}$$

在对称密钥的用户终端解密阶段，运算过程为

$$C_1(M_{part})^{\frac{1}{dk}}=yke(g_1,g_2)^{\alpha q}\left(e(g_1,g_2)^{-\alpha qs}\right)^{\frac{1}{s}}$$

$$=\frac{yke(g_1,g_2)^{\alpha q}}{e(g_1,g_2)^{\alpha q}}$$

$$=yk \tag{23}$$

可得到正确的对称密钥 yk。最后，DReq 使用对称密钥解密出文档明文。

4　模块实现

关键词加密和对称密钥加密都采用 CP-ABE 机制。对称密钥加密算法不同于关键词加密算法之处在于，关键词加密算法需要同时具备密文访问控制和密文检索功能，而对称密钥加密算法需要具备密文访问控制和外包解密功能。关键词加密算法和对称密钥加密算法使用相同的访问控制策略，因而，2 个算法中的秘密共享过程和权限验证过程完全可以一次实现、多次使用，从而减少计算开销。不同于目前已有的属性加密算法，本文将秘密共享运算和权限验证运算从关键词加密算法和对称密钥加密算法中独立出来，形成单独的算法——Share 算法和 Combine 算法。同时，对外包解密密钥的权限部分和关键词陷门的权限部分进行了整合统一，通过 PriGen 算法来生成。使在一次搜索和外包解密过程中，不需要分别进行权限的计算验证。并且，通过代理加密技术，在保证安全性的同时，使最初计算的秘密共享值，通过简单的变换后，便能够为以后每次搜索所使用，避免了在每次搜索和外包解密时秘密共享值的重复计算。

4.1　用户权限匹配

DReq 用户接收到用户属性私钥后，首先在空闲时段调用 PriGen 算法生成凭证中的权限部分 tk_{pri}，发送给 DSP。DReq 保存 PriGen 算法中选择的随机数 s，s 将在每次 DReq 具有文档需求时使用。

DSP 接收到 DReq 用户发送的凭证权限部分 tk_{pri} 后，利用空闲时段，第一步将 tk_{pri} 中的用户属性取值集合 $Atts$ 与密文中的访问树 T 进行匹配；第二步，对于 $Atts$ 满足 T 的那些密文，DSP 调用 Combine 算法，输入密文的权限部分 chp_{pri} 和凭证的权限部分 tk_{pri}，生成秘密值 $E_{root}=e(g_1,g_2)^{rsq}$，供以后的每次搜索时使用，从而避免每次搜索时，DReq 重新生成凭证的权限部分，以及 DSP 重新计算秘密值。DSP 维护一个二维表 ERoot_DReq，保存 E_{root}。

4.2 外包解密部分生成

DReq 构造凭证的权限验证部分 tk_{pri} 后，在空闲时段选择随机数 $u \in Z_p^*$，作为用户终端解密密钥 dk，并构造凭证的外包解密部分 tk_{dec}。DReq 维护一个循环队列 DEC_DReq 来保存随机数 u 及相应的凭证外包解密部分 tk_{dec}，DEC_DReq 使 DReq 能够提前预备多个外包解密部分。DReq 保存 u 用于之后构造凭证的搜索部分 tk_{sch}，以及对称密钥的终端解密。为了防止攻击者利用 DReq 过往的搜索凭证生成新的搜索凭证，仿冒 DReq 用户申请搜索文档，DReq 在每次构造凭证的外包解密部分时，都随机选择一个随机数 u，从而，保证不同搜索凭证中的组件不能互用。

4.3 搜索陷门生成

由于凭证的搜索部分（即陷门）基于搜索关键词集合构造，因此只能在用户提出搜索需求时进行计算；而凭证的权限部分和外包解密部分与搜索关键词集合无关，可以在空闲时段提前构造，从而降低搜索时的在线运算量和耗时。

DReq 从队列 DEC_DReq 中取出 u，计算 $\frac{u}{s}$，用于通过代理加密的方式替换秘密值 E_{root} 中的随机参数；其中，s 是 DReq 生成凭证权限验证部分 tk_{pri} 时选择和保存的随机数。DReq 根据搜索策略制定搜索的关键词集合；然后根据关键词集合和随机数 u，调用 TrapGen 算法构造凭证的陷门部分 tk_{sch}。DReq 将 $\frac{u}{s}$，以及该 u 相应的凭证外包解密部分，连同陷门部分，一起发送给 DSP。

4.4 云端搜索和外包

DSP 接收到 DReq 用户发送的搜索申请消息后，通过本地维护的二维表 ERoot_DReq 找到该 DReq 具有访问控制权限的文档密文，第一步从消息中取出 $\frac{u}{s}$，计算本次搜索的秘密共享值 $E_{root}=\left(e\left(g_1,g_2\right)^{rqs}\right)^{\frac{u}{s}}=e\left(g_1,g_2\right)^{rqu}$；第二步调用 Search 算法验证密文是否满足 DReq 的搜索策略。

当先后满足访问控制策略和搜索策略时，DSP 调用 DSPDec 算法对对称密钥密文进行外包解密，然后将生成的半明文组件 YK_{part}，与原密文组件 C_1 和文档密文 cph_{doc} 一起返回给 DReq。

4.5 终端解密

DReq 接收到 DSP 返回的密文后，在本地维护的队列 DEC_DReq 中找出相应的用户终端解密密钥 dk=u，调用 DReqDec 算法，对对称密钥的半明文进行用户终端解密，得到对称密钥明文 yk；然后调用文档解密算法 DocED，对消息中的文档密文 cph_{doc} 进行解密，得到最终的文档明文 doc。

5 性能优化分析

云存储系统在将传统密码与属性密码结合使用的基础上，采用以下方式进一步降低开销。

1) 通过外包解密降低终端的运算量和通信量。将对称密钥密文的解密分为外包解密和用户终端解密 2 个阶段，绝大部分运算量由 DSP 在外包解密阶段完成，使返回给终端的对称密钥密文长度以及对称密钥终端解密的运算量都降为常数级。虽然 DSP 的计算量相对增加，但是总计算量并没有明显增加。

2) 通过整合访问控制的权限验证和搜索的权限验证，降低运算量和通信量。将关键词搜索的权限验证和对称密钥解密的权限验证统一起来，规划单独的算法、密文权限验证部分和凭证权限验证部分。1) 权限验证部分同时供搜索运算和外包解密运算使用，降低了终端的运算量，减小了密文和凭证的长度；2) 权限验证结果同时供搜索运算和外包解密运算使用，降低了 DSP 的计算量。

3) 通过代理加密，降低权限验证给搜索和解密带来的运算量。传统的做法是，每次 DReq 具有文档访问需求时，都需要生成凭证的权限验证部分，而 DSP 需要根据密文和凭证的权限验证部分计算出秘密值 E_{root}。在本文方案中，DReq 只需在空闲时段一次性生成凭证的权限验证部分，由 DSP 计算出针对各个密文的 E_{root}；在每次该用户针对该密文的搜索中，DSP 只需使用代理密钥将 E_{root} 进行简单变换，即可以用于本次的搜索和外包解密，从而使 DReq 无需每次都生成凭证的权限验证部分，而 DSP 也无需每次都执行大量运算来计算 E_{root}，显著减少通信量、运算量和响应时间。

表 1 为使用上述 3 项措施前后性能对比，系统各阶段的运算量对比。假设 P 和 M 分别为循环群中的指数运算和乘法运算，E 为双线性映射运算，D 为对称加解密运算。m 为文档的关键词数量。v 表示文档访问控制策略中的属性个数，u 表示 DReq 的属性个数，i 表示参与权限验证的属性个数，有 $i \leqslant u$ 和 $i \leqslant v$。

表 1　　使用措施前后性能对比

工作项	使用前运算量	使用后运算量
构造密文的权限（部分）	$4vP$	$2vP$
构造对称密钥密文（部分）	$2P+M$	$2P+M$
构造关键词密文（部分）	$(m+4)P+M$	$(m+4)P+M$
文档加密（总共）	$(4v+m+6)P+2M+D$	$(2v+m+6)P+2M+D$
构造凭证的权限（部分）	$2uP$	$2uP$
构造凭证的外包解密（部分）	0	P
构造凭证的搜索（部分）	$(m+3)P$	$(m+3)P$
在线凭证构造（总共）	$(2u+m+3)P$	$(m+4)P$
权限验证	$2iE+iM$	P
搜索匹配	$(m+2)E+(m+2)M$	$(m+2)E+(m+2)M$
外包解密	0	$E+P$
云端搜索（总共）	$(2i+m+2)E+(i+m+2)M$	$(m+3)E+(m+4)M$
终端解密	$2iE+(i+1)M+D$	$P+M+D$

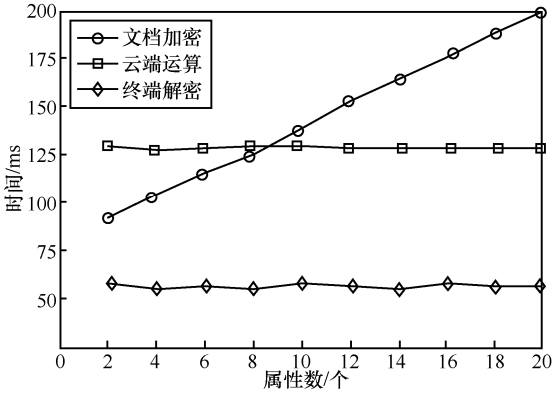
从表 1 可以看出，使用上述 3 项性能优化措施后，在线凭证构造运算量、云端搜索运算量和终端解密运算量都大幅降低。

6　性能测试

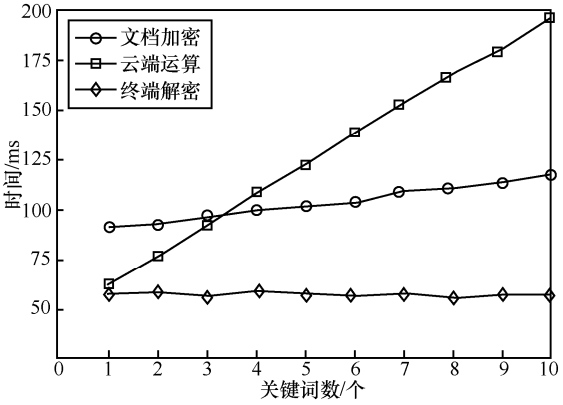
基于数学运算库 GMP-6.1.2 和映射运算库 PBC-0.5.14，对属性加密方案 cpable-0.11 的源代码进行了修改，实现了本云存储实验系统，并基于 cpable-0.11 中的参数进行了性能测试。实验环境中使用一台联想 M8500T 台式机，硬件配置为 i7-4790 3.6 GHz 处理器，4 GB 内存。在 Windows 7 操作系统，Oracle VirtualBox 虚拟机管理器上安装 Centos 6.3 虚拟机，分配内存 2 GB。

实验中设置文档访问策略中的属性个数、DReq 的属性个数、权限验证时使用的属性个数一致；设置文档的关键词个数与关键词最大个数一致。同时，为了减小计算机随机因素对实验的影响，未使用文档的对称加解密。

图 2 给出了加/解密时间与属性个数和关键词个数的关系，图 2 中的耗时都是取 10 次运行结果的平均值。其中，图 2(a)是关键词个数为 2 时，不同属性数量情况下，除文档对称加密外的加密总耗时，云端搜索和外包解密总耗时，以及终端解密耗时。图 2(b)给出了属性个数为 2 时，除文档对称加密外的加密总耗时，云端搜索和外包解密总耗时，以及终端解密耗时。



(a) 加/解密时间与属性数量的关系



(b) 加/解密时间与关键词数量的关系

图 2　加/解密耗时趋势

从仿真实验的结果可以看出：随着属性个数和关键词个数的变化，加/解密所用的时间与第 5 节中给出的性能分析结果基本一致。文档加密共所用时间与属性个数、关键词个数都分别呈线性关系，云端搜索和外包解密共所用时间与关键词个数呈线性关系，终端解密时间始终为常量。

7　结束语

本文融合密文访问控制、外包解密、关键词连接搜索和代理重加密技术，创新性地提出了一种支持安全共享的云存储系统方案。该方案结合传统密码体制和双线性映射的属性密码体制，即利用传统密码实现了较高的加/解密性能，又利用属性密码实现了多用户细粒度访问控制和关键词密文搜索。同时，通过外包解密、代理加密等技术手段，降低了系统和终端用户的开销。基于性能分析和实验验证说明了方案的有效性。

参考文献：

[1]　傅颖勋, 罗圣美, 舒继武. 一种云存储环境下的安全网盘系统[J].

软件学报, 2014, 25(8): 1831-1843.
FU Y X, LUO S M, SHU J W. Secure online storage system based on cloud storage environment[J]. Journal of Software, 2014, 25(8): 1831-1843.

[2] 薛矛, 薛巍, 舒继武, 等. 一种云存储环境下的安全存储系统[J]. 计算机学报, 2015, 38(5): 987-998.
XUE M, XUE W, SHU J W, et al. A secure storage system over cloud storage environment[J]. Chinese Journal of Computers, 2015, 38(5): 987-998.

[3] SAHAI A, WATERS B. Fuzzy identity-based encryption[C]// Advances in Cryptology (EUROCRYPT 2005), Aarhus, Denmark, 2005. Berlin Heidelberg: Springer, 2005: 457-473.

[4] GOYAL V, PANDEY O, SAHAI A, et al. Attribute-based encryption for fine-grained access control of encrypted data[C]//The 13th ACM Conference on Computer and Communications Security (CCS' 06). 2006: 89-98.

[5] BETHENCOURT J, SAHAI A, WATERS B. Ciphertext-policy attribute-based encryption[C]//The 2007 IEEE Symposium on Security and Privacy (SP' 07)2007. 2008: 321-334.

[6] ATTRUPADUNG N, HERRANZ J, LAGUILLAUME F, et al. Attribute-based encryption schemes with constant-size ciphertexts[J]. Theoretical Computer Science, 2012, 422: 15-38.

[7] GREEN M, HOHENBERGER S, WATERS B. Outsourcing the decryption of ABE ciphertexts[C]//The 20th USENIX conference on security (SEC 2011). 2011: 34-49.

[8] SONG D, WAGNER D, PERRIG A. Practical techniques for searches on encrypted data[C]//The IEEE Symposium on Security and Privacy(S&P'00).2000: 44-55.

[9] BONEH D, CRESCENZOM G D, OSTROVSKY R, et al. Public key encryption with keyword search[C]//The International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2004). 2004: 506-522.

[10] LEE C C, HSU S T, H M S. A study of conjunctive keyword searchable schemes[J]. International Journal of Network Security, 2013, 15(5): 321-330.

[11] GOLLE P, STADDON J, WATERS B. Secure conjunctive keyword search over encrypted data[C]//Applied Cryptography and Network Security Conference (ACNS 2004). 2004: 31-45.

[12] PARK D J, KIM K, and LEE P J. Public key encryption with conjunctive-field keyword search[C]//The 5th Information Security Applications International Workshop (WISA 2004). 2004: 73-86.

[13] HWANG Y H, LEE P J. Public key encryption with conjunctive keyword search and its extension to a multi-user system[C]//The First International Conference of Pairing-Based Cryptography (Pairing 2007). 2007: 2-22.

[14] ZHANG B, ZHANG F G. An efficient public key encryption with conjunctive-subset keywords search[J]. Journal of Network and Computer Applications,2011, 34(1):262-267.

作者简介：

宋衍（1982-），男，湖北老河口人，北京交通大学博士生，信息保障技术重点实验室工程师，主要研究方向为密态计算、云计算安全等。

韩臻（1962-），男，浙江宁波人，北京交通大学教授、博士生导师，主要研究方向为信息安全体系结构、可信计算等。

李建军（1980-），男，山东荣城人，信息保障技术重点实验室博士后，主要研究方向为态势感知、智能防御等。

韩磊（1983-），男，内蒙古海拉尔人，信息保障技术重点实验室博士后，主要研究方向为云计算、态势感知等。