雾计算中细粒度属性更新的外包计算访问控制方案

杜瑞忠[1,2]，闫沛文[1]，刘妍[1]

（1. 河北大学网络空间安全与计算机学院，河北 保定 071000；2. 河北省高可信信息系统重点实验室，河北 保定 071000）

摘 要：针对基于密文策略的属性加密（CP-ABE）在低时延需求较高的雾计算环境中，存在加解密开销大、属性更新效率低的问题，提出了一种雾计算中细粒度属性更新的外包计算访问控制方案，使用模加法一致性秘密（密钥）分享技术构建访问控制树，将加解密计算操作外包给雾节点，降低用户加解密开销；结合重加密机制，在雾节点建立组密钥二叉树对密文进行重加密，实现对用户属性的灵活更新。安全性分析表明，所提方案在决策双线性 Diffie-Hellman 假设下是安全的。仿真实验结果表明，所提方案中用户加解密时间开销相比其他方案更小，属性更新效率更高。

关键词：访问控制；雾计算；外包计算；属性更新；基于密文策略属性加密

中图分类号：TP309

文献标识码：A

DOI: 10.11959/j.issn.1000−436x.2021063

Fine-grained attribute update and outsourcing computing access control scheme in fog computing

DU Ruizhong[1,2], YAN Peiwen[1], LIU Yan[1]

1. School of Cyber Security and Computer, Hebei University, Baoding 071000, China
2. Key Lab on High Trusted Information System in Hebei Province, Baoding 071000, China

Abstract: To slove the problem that in the fog computing environment with comparatively high low latency demand, ciphertext policy attribute based encryption (CP-ABE) faced the problems of high encryption and decryption overhead and low efficiency of attribute update, an fine-grained attribute update and outsourcing computing access control scheme in fog computing was proposed. The unanimous consent control by modular addition technique was used to construct an access control tree, and the computing operations of ecryption and decryption were outsourced to fog nodes in order to reduce user encryption and decryption overhead. Combined with the re-encryption mechanism, a group key binary tree was established at the fog node to re-encrypt the ciphertext so that user attribute can be updated flexibly. The security analysis shows that the proposed scheme is safe under the decision bilinear Diffie-Hellman hypothesis. Compared with other schemes, the results of simulation experiment prove that the time cost of user encryption and decryption in this scheme is lower and the efficiency of attribute update is higher.

Keywords: access control, fog computing, outsourcing computing, attribute update, CP-ABE

1 引言

云计算通过基础设施向用户提供服务，为应用提供弹性资源，但云计算提供的资源主要集中在远离用户的核心网络中，这导致用户和云之间交互信息时会产生时延。随着用户对低时延的需求越来越高，雾计算作为云计算的延伸被提出[1]。雾计算将云服务与靠近网络边缘的分布式资源相结合，使存

收稿日期：2020−09−07；**修回日期**：2020−12−07

基金项目：国家自然科学基金资助项目（No.61572170）；河北省自然科学基金资助项目（No.F2018201153）；河北省自然科学基金重点资助项目（F2019201290）

Foundation Items: The National Natural Science Foundation of China (No.61572170), The Natural Science Foundation of Hebei Province (No.F2018201153), The Key Project of Natural Science Foundation of Hebei Province (No.F2019201290)

储和数据处理等贴近网络边缘设备，可以提供外包计算、资源分配和缓存等多种服务[2-4]，有效缓解了云计算中存在的时延问题，在智能医疗、智慧城市等对低时延需求较高的领域被广泛应用[5-6]。

雾计算更贴近终端用户，数据中包含大量用户隐私信息，由于存在虚假节点恶意获取数据的情况，因此需要一种雾计算环境下高效的访问控制方案[7-8]。在雾计算中，边缘设备基于密文策略的属性加密（CP-ABE, ciphertext policy attribute based encryption）是实现细粒度访问控制的技术之一，数据所有者根据属性对数据进行加密，而不需要知道用户的确切身份，只有满足属性的用户才能对数据进行访问。CP-ABE 多采用门限秘密共享方案，需用多项式插值法来重建秘密，其解密阶段配对和求幂开销较大的计算，给雾计算中资源受限的边缘设备带来计算瓶颈。另外，雾计算中边缘设备的属性会频繁变化，为防止用户使用更新前属性密钥进行访问，需要建立有效即时属性撤销机制以提高其灵活性。

为了解决上述问题，本文提出了一种雾计算中细粒度属性更新的外包计算访问控制方案；结合雾计算的特点，建立了云−雾−用户分层的访问控制架构，用户通过雾节点与云进行交互；采用 CP-ABE 技术实现数据机密性和细粒度访问控制，并设计安全高效的外包加解密方案，将加密和解密中繁重的计算外包给雾节点，使用模加法技术构建访问控制树，避免了多项式插值法产生的计算开销，使加密、密钥生成和解密阶段的计算量更少，减少数据所有者和数据用户的计算负担；在雾节点建立组密钥二叉树生成属性组密钥，用户提交访问请求时对密文进行重加密，实现了属性的细粒度更新与即时撤销。通过雾节点对用户进行管理，减小了组密钥二叉树规模与更新私钥的用户数量，并在更新时将属性组内用户标记为未更新用户，避免了频繁更新私钥的冗余操作，提升了属性更新效率；对方案进行仿真实验对比，实验结果表明，用户加解密时间消耗较小且稳定，外包解密阶段耗时更少，属性更新时的效率更高。

本文主要的研究工作如下。

1) 建立更适用于雾计算的云−雾−用户分层的访问控制架构，雾节点对用户进行管理，用户必须通过雾节点与云进行交互。

2) 改进模加法机制构建访问控制树，降低计算开销，利用外包加解密思想和重加密机制设计了一种雾计算环境下细粒度属性更新的外包计算访问控制方案。

3) 进行安全性分析，基于决策双线性 Diffie-Hellman 假设，证明了所提方案满足明文攻击安全。通过仿真实验分析与其他方案在各阶段运行时间进行对比，所提方案在满足安全性前提下，效率更高。

2　相关工作

2007 年，Bethencourt 等[9]首次提出 CP-ABE 方案，之后其被广泛应用于云存储环境中数据的细粒度访问控制[10-11]，但由于加解密的计算开销随着访问结构中属性的数量线性增加，该方案效率较低且不适用于资源受限的边缘设备。Ibraimi 等[12]提出了一种高效的 CP-ABE 算法，该算法使用模加法一致同意控制方案（UCCMA, unanimous consent control by modular addition）建立访问控制树，避免多项式插值法的计算开销。为了进一步提高效率并减小用户负担，将加解密阶段的复杂计算进行外包[13-14]。Jin 等[15]以文献[12]为基础，设计了安全轻量级的访问控制方案用于移动云计算，引入加密服务提供商（ESP, encryption service provider）和解密服务提供商（DSP, decryption service provider）降低加解密过程中的计算开销，在实现系统高效性的同时，将移动设备大部分计算外包给 ESP 和 DSP，使移动设备可以有效地对存储在云端的数据进行访问和管理。Green 等[16]改进了 ABE（attribute based encryption）外包系统中的密钥生成算法，使用盲密钥技术生成转换密钥并发送给第三方代理，将满足访问策略的密文转换为简单密文，使明文恢复所需开销较小。为解决外包计算的正确性问题，Mao 等[17]和 Kumar 等[18]提出了基于属性的可验证外包解密的访问控制方案，外包计算后验证了数据的正确性，用户解密密文时只需要进行恒定数量的计算。外包计算的访问控制方案得到了广泛应用，Kibiwott 等[19]提出了基于云的电子健康大数据系统访问控制方案，通过 cloudlet 服务器进行外包计算，实现了机密性和数据完整性。但以上外包计算方案没有考虑属性撤销问题，在动态性较强的雾计算中并不适用。

在 CP-ABE 方案中，为防止用户使用过期的属性密钥，必须使用密钥撤销机制。Al-dahhan 等[20]

通过管理联合属性集存储数据，提供策略更新和密钥失效 2 种技术来消除共谋攻击，并解决密钥撤销问题，但密钥维护代价和计算复杂度高。Zhang 等[21]提出了一种雾环境下外包加密和属性更新方案，在属性更新时，所有拥有该属性的用户必须对属性密钥进行更新，对未撤销属性进行大量计算，这会大大降低属性更新的效率。为了避免撤销属性时密钥的大量更新，Chen 等[22]在方案中引入衰落函数，使属性密钥可以随时间进行动态更新，需要为每个属性设置生存周期，但选取合适的生存周期是一个难题，且衰落函数存在产生冲突的情况。Hur 等[23]提出了一种建立二叉树对密文双重加密的方法，庞大的设备数量会导致二叉树的维护开销较大。Liu 等[24]将用户身份信息分配给叶子节点信息以跟踪恶意用户，并通过将恶意用户添加到吊销列表和更新密文，实现对恶意用户的撤销操作。张凯等[25]提出了高效撤销的多机构访问控制方案，将撤销列表嵌入密文对用户的整体属性进行撤销，但无法对部分属性进行撤销操作，存在撤销粒度较粗的问题。以上方案虽然解决了属性撤销问题，但目前属性密钥撤销仍然给系统带来较大的开销。

综上所述，传统的访问控制方案效率较低，并缺乏高效的属性撤销方法。为此，本文提出了雾计算中细粒度属性更新的外包计算访问控制方案，在保障安全的前提下，降低用户开销，提升属性更新效率。

3 预备知识

3.1 双线性映射

G, G_T 是 2 个阶为 P 的循环群，g 是 G 的生成元，双线性映射 $e: G_0 \times G_0 \to G_T$ 满足以下性质。

1) 双线性：对任意 $u, v \in G_0$ 和 $a, b \in Z_{\mathrm{p}}$，都有 $e(u^a, v^b) = e(u, v)^{ab}$。

2) 非退化性：存在 $u, v \in G_0$ 使 $e(u^a, v^b) \neq 1$。

3) 可计算性：对于任意的 u，v，存在一个有效的算法计算 $e(u, v)$。

3.2 访问控制树

令根节点为 r 的访问控制树 T 描述访问控制策略，该策略指定了在解密密文时需要的属性组合。T 中的每一个内部节点都是逻辑运算符，例如 AND、OR。每个叶子节点都表示一种属性。属性可以是定义、分类或注释用户的任何描述性字符串。根据秘密共享的思想，访问树的每个节点表示一个秘密。在加密阶段需要自顶向下递归地为每一个节点分配一个秘密。而在解密阶段，需要自底向上回复根节点的秘密。

3.3 秘密分享

在秘密分享方案中，发放者将一个秘密 S 分割成 t 份，需要所有的分割部分才可以重新构造出 S。为了共享秘密 S（$0 \leqslant S \leqslant p-1$），发放者生成 $t-1$ 个随机数 S_i，即 $1 \leqslant S_i \leqslant p-1$，$1 \leqslant i \leqslant t-1$，同时 $S_t = S - \sum_{i=1}^{t-1} S_i \bmod p$。秘密 S 通过 $S = \sum_{i=1}^{t} S_i$ 进行恢复，将 S_i 分享给 P_i 个参与方，$1 \leqslant i \leqslant t$。对每一个参与者 P_i，得到的是 0～p−1 的随机数，因此，除了发放者之外没有任何一方知道关于秘密 S 的任何信息。

3.4 决策双线性

决策双线性 Diffie-Hellman（DBDH, decision bilinear Diffie-Hellman）问题的定义如下。挑战者根据安全参数选择一组阶为素数 p 的 G_0。让 g 作为生成元，选取随机数 $a, b, s \in Z_p$。如果挑战者给予对手 (g, g^a, g^b, g^s)，对于对手来说，将很难区分有效的元组 $e(\mathrm{g},\mathrm{g})^{\mathrm{abs}} \in G_T$ 与随机元素 $R \in G_T$。

一个概率性多项式时间算法 Q 能够以优势 ε 求解 DBDH 问题，当且仅当满足

$$|\Pr[Q(g, g^a, g^b, g^s, Z = e(g,g)^{abs}) = 0] - \Pr[Q(g, g^a, g^b, g^s, Z = R) = 0]| \geqslant \varepsilon$$

如果概率多项式在解决 DBDH 问题上的优势可以忽略不计，则 DBDH 假设成立。

4 系统模型

4.1 整体框架

本文为解决雾计算中用户加解密开销大、属性更新效率低的问题，设计了云−雾−用户分层的系统模型，主要由 5 类实体组成，系统结构如图 1 所示。

云服务提供商（CSP, cloud service provider）。CSP 提供数据存储等方面的服务，在本文方案中假设 CSP 可信。

雾节点。雾节点提供计算、存储等服务，负责对密文进行外包加解密计算，生成组密钥对密文进行重加密，并在未更新用户提交访问请求时，更新其私钥。

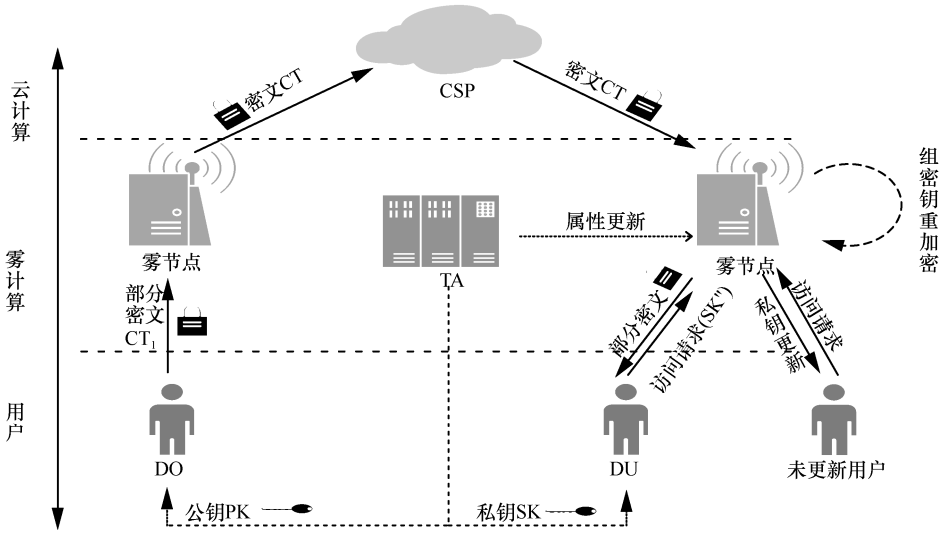

图 1　系统结构

数据所有者（DO, data owner）。DO 是有文件要上传到云端存储的用户，为文件定义访问结构，然后生成部分密文发送到雾节点。

数据用户（DU, data user）。DU 是访问存储在 CSP 中的文件的用户，当数据用户的属性集满足嵌入给定密文中的访问结构时，可以从密文中解密文件。若用户被标记为未更新用户，在请求访问时需要先更新私钥。

可信权威机构（TA, trusted authority）。TA 是完全受信任的一方，遵循协议规范执行分配任务，然后输出正确的结果，主要负责处理用户注册，为每个系统用户生成系统参数和密钥 SK，向雾节点发送属性更新信息。

4.2　算法定义

本文系统由系统初始化（Setup）、私钥生成（KeyGen）、数据加密（Encrypt）、组密钥生成（GKeyGen）、密文重加密（reEncrypt）、私钥更新（upGKeyGen）、密文解密（Decrypt）几个算法构成。

1) 系统初始化 Setup(1^k, L)$\rightarrow$(PK, MSK)。初始化阶段输入安全参数 k 和属性域 $L=\{a_1, a_2,\cdots,a_m\}$，进行运算获得系统的公钥 PK 和主密钥 MSK。

2) 私钥生成 KeyGen(MSK, S)$\rightarrow$SK。私钥生成算法通过输入主密钥 MSK，以及私钥所包含的属性集合 S，计算得到私钥 SK 并发送给用户。

3) 数据加密 Encrypt。为了获得较高的加密效率，DO 首先使用随机密钥 ck 对指定文件进行加密，然后使用对称加密算法对 ck 进行加密，该算法包含以下 2 个子算法：DO 在本地运行 DO.Encrypt，雾节点运行外包加密算法 Fog.Encrypt。

DO.Encrypt(PK, ck, A)$\rightarrow$$\mathrm{CT}_1$，算法输入 PK、ck 和访问结构 A，然后输出部分密文 CT_1，最后 DO 将 CT_1 发送给雾节点。

Fog.Encrypt(PK, CT_1)$\rightarrow$CT，该算法输入 PK 和 CT_1，然后输出密文 CT 并发送给 CSP。

4) 组密钥生成 GKeyGen($a_1, a_2,\cdots, a_m$)。该算法输入属性域 $L=\{a_1,a_2,\cdots,a_m\}$ 给雾节点，雾节点为用户生成二叉树 τ 并推算出组密钥 $E_j(1\leqslant j\leqslant m)$。

5) 密文重加密 reEncrypt(E_j, CT)$\rightarrow$CT′。雾节点收到数据用户访问请求后，从 CSP 获取密文 CT，利用生成的组密钥 $E_j(1\leqslant j\leqslant m)$ 与密文 CT 进行运算，得到重加密密文 CT′。

6) 私钥的更新 upGKeyGen(E_i, SK)$\rightarrow$SK′。用户通过计算得到组密钥 E_j，与 SK 中对应的属性私钥进行更新运算，得到新私钥 SK′。

7) 密文解密 Decrypt。雾节点接受用户发来的部分密钥 SK″，与重加密之后的 CT′ 进行计算得到 T，并发送给用户。如果用户的密钥满足访问结构 A，则运行以下算法得到 ck，该算法包括 2 个子算法：雾节点运行外包解密算法 Fog.Decrypt，用户执行本地子算法 User.Decrypt。

Fog.Decrypt(PK,CT′,SK″)$\rightarrow$$T$。输入 PK、CT′ 和 SK″，通过计算输出结果 T，并将其发送给 DO。

User.Decrypt(T,SK)$\rightarrow$ck。使用 T 和 SK 进行计算，生成密钥 ck。

因此，使用 ck 通过对称解密算法将得到文件。

5 方案描述

本节描述了雾计算中细粒度属性更新的外包计算访问控制方案的具体结构。雾计算应用对低时延有较高需求，因此降低用户计算复杂度和提高属性更新效率成为用户的重要需求。首先，本节提出一种安全外包加解密方法，以减少用户在访问控制过程中的计算负担。外包方法主要是将加密和解密阶段所有访问结构和属性相关的操作外包给雾节点，留给数据所有者和数据用户执行少量操作。然后，利用重加密机制在雾节点和用户之间构建一个组密钥二叉树 τ，随机生成树中的节点密钥，并生成组密钥，当用户的属性要增加或者撤销时，仅需改变属性的组密钥即可实现属性的细粒度更新。

5.1 具体方案

假设系统中拥有 m 个属性并表示为 $L=\{a_1,a_2,\cdots,a_m\}$，将用户所属雾节点设置为一种属性。$e: G_0\times G_0\to G_T$ 为双线性映射，G_0 为素数阶 p 以 g 为生成元的双线性群。哈希函数 $H: \{0,1\}^*\to G_0$，将任意属性映射到 G_0 的一个随机元素上，具体细节如下。

阶段 1 系统初始化 Setup

Setup(1^k, L)$\to$(PK, MSK)。系统定义阶为 p、以 g 为生成元的双线性映射群 $e: G_0\times G_0\to G_T$，随机选取参数 $\alpha\in Z_p$，对任意 $a_j\in L$ 选取一个随机数 v_j，计算 $\mathrm{PK}_j=g^{v_j}$，设置雾节点属性值为 $A_{\mathrm{fg}}\in Z_p$，最后得到公钥与主密钥。

$$\mathrm{PK}=\{G_0,g,e(g,g)^{\alpha},\{\mathrm{PK}_j=g^{v_j}|a_j\in L\}\}\ \ \mathrm{MSK}=\{\alpha,A_{\mathrm{fg}},\{v_j\,|\,a_j\in L\}\}$$

阶段 2 私钥生成 KeyGen

KeyGen(MSK, S)$\to$SK。该算法首先随机选择 r=Z_p，r 是分配给每个系统用户的唯一密钥；然后，选取随机数 $y\in Z_p$，将雾节点属性标记为 D_{fg}，计算私钥 $\mathrm{SK}=\{D=g^{\alpha+r},D'=g^{y},D''=g^{yr},\forall a_j\in S:\ D_j=g^{rv_j^{-1}},D_{\mathrm{fg}}=g^{rA_{\mathrm{fg}}^{-1}}\}$。

阶段 3 数据加密 Encrypt

在文件 M 上传到 CSP 之前需要进行的步骤如下。

1) DO.Encrypt(PK, ck, A)$\to\mathrm{CT}_1$。DO 定义一个访问结构 A，并随机选取密钥 ck，用对称加密算法对 M 进行加密，加密后的密文表示为 $E_{\mathrm{ck}}(M)$。DO 随机选择 $S'\in Z_p$，计算 $\tilde{C}=\mathrm{ck}e(g,g)^{\beta s'}$。输出密文 $\mathrm{CT}_1=\{A,E_{\mathrm{ck}}(M),\ \ \tilde{C}=\mathrm{ck}e(g,g)^{\beta s'}\ \ C=g^{s'}\}$。用户 DO 发送 CT_1 给雾节点。

2) Fog.Encrypt(PK , CT_1)$\to$ CT。雾节点根据收到的访问结构 A 建立访问控制树 T，如图 2 所示，为 T 的根节点赋值为 S，随机选择 $S_{\mathrm{fg}}\in Z_p$ 将其作为根节点的右子树，其他子节点标记为未赋值，对于每个未赋值的非叶子节点递归地执行以下操作。

如果运算符是 AND 且它的子节点未赋值，使用模加法方案为每个子节点赋值，对于除了最后一个子节点之外的每一个节点赋值为随机数 S_i ($1\leqslant S_i\leqslant p-1$)，将最后一个子节点赋值为 $S_t=S-\sum_{i=1}^{t-1}S_i \bmod p$。

如果运算符是 OR，为每一个子节点赋值 S。T 树中叶子节点的值用来生成密文。为每一个叶子节点属性 $a_{j,i}\in T$，计算属性密钥 $C_{j,i}=g^{v_j s_i}$，i 为访问树的索引属性，对于给定的访问结构，索引值被唯一地以排序的方式分配给节点。

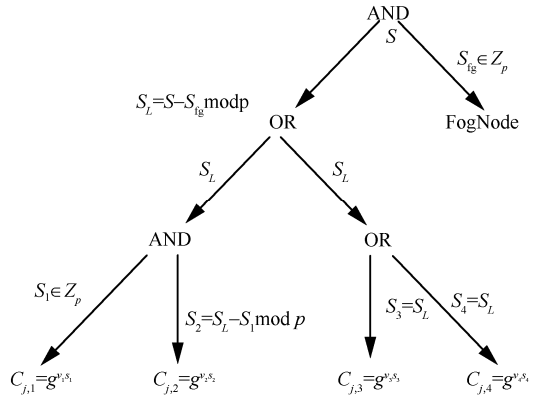

图 2 访问控制树

最后通过计算得到

$$\mathrm{CT}=\left\{E_{\mathrm{ck}}(M),\tilde{C}=\mathrm{ck}e(g,g)^{\alpha s'},C=g^{s'},C'=g^{s'}g^{s},\right.$$
$$\left.C_{\mathrm{fg}}=g^{S_{\mathrm{fg}}A_{\mathrm{fg}}},\{C_{j,i}=g^{v_j s_i},\forall a_{j,i}\in T:C_{j,i}\}\right\}$$

阶段 4 组密钥生成 GKeyGen

组密钥二叉树 τ 的构成原理如下。雾节点根据管理用户构建二叉树 τ，如图 3 所示，将所有用户分配到叶子节点，并为 τ 中的每一个节点生成一个节点密钥 $u_{n,m}\in Z_p$，用户叶子节点到根节点通过的

节点称为路径节点，用户存储路径节点的节点密钥。雾节点为属性定义一个公开参数 $\gamma_j \in Z_p$，根据属性所对应的用户组，找到包含用户的最大覆盖子树，建立组密钥 $E_j=\{V_{1,1},V_{2,1},\cdots,V_{n,m}\|\gamma_j\}$，通过门限法将 E_j 设置为(*S*,2)秘密共享，推算出 E_j。假设属性 ω_j 的用户组为 $\{u_1,u_2,u_3,u_4,u_7,u_8\}$，在组密钥二叉树中的最大覆盖节点为 $\{V_{2,1},V_{3,4}\}$，则属性 ω_j 的组密钥为 $E_j=\{V_{2,1},V_{3,4}\|\gamma_j\}$。

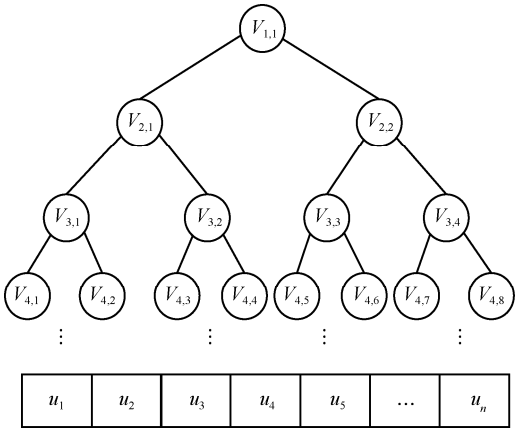

图 3　属性组密钥二叉树

阶段 5　密文重加密 reEncrypt

当用户向雾节点请求访问数据后，雾节点从 CSP 获取密文 CT，然后通过组密钥 E_j 对 CT 进行重加密，计算得到

$$\mathrm{CT}'=\left\{E_{\mathrm{ck}}(M),\tilde{C}=\mathrm{ck}e(g,g)^{\alpha s'},C=g^{s'},C'=g^{s'}g^{s},\right.$$
$$\left.C_{\mathrm{fg}}=g^{S_{\mathrm{fg}}A_{\mathrm{fg}}},\{C_{j,i}=g^{v_j s_{iE_j}},\forall a_{j,i}\in T:C_{j,i}\}\right\}$$

阶段 6　密钥更新 upGKeyGen

雾节点接收到 DO 的访问请求之后，如果数据用户被标记为未更新用户。雾节点将其需要更新属性路径节点所在二叉树层数发送给用户。门限方案中 E_i 设置为(*S*, 2)秘密共享，只要知道两部分就可以计算出组密钥 E_i，由于 γ_i 是公开的，因此只要属性覆盖用户组内包含该用户，便能够计算出 E_i，DO 利用 E_j 进行计算，$D_j'=D'^{1/E_j}$，则更新之后的私钥为 $\mathrm{SK}'=\{D=g^{\alpha+\mathrm{r}},D'=g^{\varepsilon},D''=g^{\varepsilon r},\forall a_j\in S:\ D_j=\ g^{rv_j^{-1}E_j^{-1}},$ $D_{\mathrm{fg}}=\mathrm{g}^{rA_{\mathrm{fg}}^{-1}}\}$。

阶段 7　密文解密 Decrypt

如果用户拥有的属性满足访问结构 *A*，则运行下面的解密算法解密 CT，获得内容密钥 ck，用户可以使用 ck 解密 $E_{\mathrm{ck}}(M)$ 并获得明文 *M*，由于密文和密钥进行双线性配对操作需要大量的计算开销，因此解密的计算开销非常大，为了减少用户的计算开销，将这些操作外包给雾节点。该解密算法分为外包解密算法 Fog.decrypt 和 User.Decrypt。

1) Fog.Decrypt(PK,CT′,SK″)$\rightarrow T$。雾节点从用户接受部分密钥 $\mathrm{SK}''=\{D'=g^{y},D''=g^{yr},\ \forall a_j\in S:$ $D_j=g^{rv_j^{-1}E_j^{-1}},D_{\mathrm{fg}}=g^{rA_{\mathrm{fg}}^{-1}}\}$ 对私钥中的 D_j 与重加密后的 CT′ 中的 $C_{j,i}$ 进行如下运算

$$F_R=\prod_{a_j\in s}e(C_{j,i},D_j)=\prod_{a_j\in s}e(g^{v_j s_i E_j},g^{rv_j^{-1}E_j^{-1}})=e(g,g)^{rs_L}$$

雾节点接着计算

$$\psi=F_R e(C_{\mathrm{fg}},D_{\mathrm{fg}})=e(g,g)^{rs_L}e(g^{S_{\mathrm{fg}}A_{\mathrm{fg}}},g^{rA_{\mathrm{fg}}^{-1}})=e(g,g)^{rs}$$

$$\omega=\frac{\dfrac{e(D'',C')}{e(D',C')}}{\psi}=\frac{\dfrac{e(g^{\varepsilon r},g^{s'}g^{s})}{e(g^{\varepsilon},g^{s'}g^{s})}}{e(g,g)^{rs}}=e(g,g)^{rs'}$$

将得到的 $T=\{E_{\mathrm{ck}}(M),\tilde{C}=\mathrm{ck}e(g,g)^{\alpha s'},\ C=g^{s'},\omega\}$ 发送给用户。

2) User.Decrypt(*T*,SK')$\rightarrow$ck。用户收到雾节点发送的 *T*，与私钥 SK'计算得到 ck。

$$\frac{\tilde{C}\omega}{e(D,C)}=\frac{\mathrm{ck}e(g,g)^{\alpha s'}e(g,g)^{rs'}}{e(g^{\alpha+r},g^{s'})}=\mathrm{ck}$$

因此，使用 ck 通过对称解密算法将得到 *M*。

5.2　属性更新

1) 属性添加与撤销

当用户属性进行添加或撤销时，雾节点根据更新后用户叶子节点重新寻找其最大子树的根节点，运行 GKeyGen 算法完成组密钥的更新。用户添加或撤销新的属性 ω_j 时，TA 将属性的更新信息发送给雾节点，雾节点根据更新后属性 ω_j 包含的用户组，寻找能覆盖所有用户的最大子树节点，根据组密钥的构成方法计算得到新 E_j'。将其他拥有属性 ω_j 的用户标记为未更新用户，当未更新用户向雾节点发送访问请求时，进行私钥更新，避免在属性频繁更新时，给用户带来冗余操作。属性 ω_j 的组密钥更新不会影响其他雾节点域内属性的组密钥，仅需要该雾节点域内持有该属性的用户更新私钥，减小了更新所带来的操作开销。

2) 用户整体变更

对用户的变更分为用户加入与撤销，可以看作

向新用户分配属性和将老用户的所有属性进行撤销。因此，用户的整体变更可以归纳为对属性更新。

① 当用户节点加入系统时，将用户分配到空叶子节点，为用户生成组密钥，并对用户私钥进行更新。

② 当用户节点离开系统时，则修改用户从叶子节点到根节点的所有密钥链。首先雾节点生成随机数 $V_{\text{tmp}} \in Z_p$，V_{tmp} 与密钥链节点进行 XOR 操作生成新的节点密钥；完成更新的节点将其节点密钥通过广播加密的方式发送给其子树的叶子节点。

6 安全性分析

在所提雾计算中细粒度属性更新和外包计算访问控制方案中选择明文攻击安全模型，基于 DBDH 问题进行安全性分析。设 G, G_T 为 2 个阶为 P、生成元为 g 的双线性群，双线性映射 $e: G_0 \times G_0 \to G_T$。模型生成 (g, g^a, g^b, g^s, Z)，其中 $a, b, s, \theta \in Z_p^*$，设置参数 $\vartheta \in \{0,1\}$，一个随机元素 $R \in G_T$，当参数 $\vartheta = 0$ 时，$Z = e(g,g)^{abs}$；否则 $Z = R$。

定理 1 如果故手 $\mathcal{A}$ 能够在一个概率多项式时间内以不可忽略的优势 $\varepsilon > 0$ 求解，那么算法 φ 能够区分一个 DBDH 元组和一个随机元组，并且优势为 $\varepsilon / 2$。

证明 敌手 $\mathcal{A}$ 和挑战者 $\mathcal{B}$ 执行操作如下。

初始化 敌手 $\mathcal{A}$ 定义访问控制树 T*，并发送给挑战者 $\mathcal{B}$。

系统设置 挑战者 $\mathcal{B}$ 对系统进行初始化，$\mathcal{B}$ 选择一个随机数 $\chi' \in Z_p$，计算 $\alpha = \chi' + ab$。设置 $u = e(g,g)^{\alpha} = e(g,g)^{\chi'} e(g,g)^{ab}$。对每一个属性 $a_j \in L$，$\mathcal{B}$ 选择随机数 s_j，如果 $a_j \in T^*$，则 $v_j = as_j^{-1}$；如果 $a_j \notin T^*$，则 $v_j = s_j$。最后 $\mathcal{B}$ 将公钥 $\text{PK} = \{G_0, g, u, \{\text{PK}_j | a_j \in L\}\}$ 发送给敌手 $\mathcal{A}$。

阶段 1 $\mathcal{A}$ 选取任意属性集 $S \subseteq L$，向 $\mathcal{B}$ 发送私钥请求。$\mathcal{B}$ 选取随机数 $r' \in Z_p$，计算 $r = ar' - ab$，则 $D = g^{\alpha + r} = g^{(\chi' + ab) + (ar' - ab)} = g^{\chi' + ar'} = g^{\chi'} A^{r'}$，计算 $D' = g^{a\varepsilon} = A^{\varepsilon}$，$D'' = g^{\alpha r' \varepsilon} = A^{r' \varepsilon}$，$D_{\text{fg}} = g^{r' A_{\text{fg}}^{-1}}$。当 $a_j \in T^*$，$D_j = g^{(ar' - ab)\, as_j^{-1}} = g^{r' s_j - bs_j} = g^{r' s_j} \psi^{s_j^{-1}}$ 时运行 upGKeyGen 算法得到组密钥 E_s。最后将私钥与组密钥都发送给敌手 $\mathcal{A}$。

挑战阶段 敌手 $\mathcal{A}$ 向挑战者 $\mathcal{B}$ 发送长度相同的两段信息 m_0 和 m_1，$\mathcal{B}$ 选择一个值 $S' \in Z_p$，随机选取参数 $\vartheta \in \{0,1\}$，通过执行 Encrypt 和 reEncrypt 算法对信息 m_{ϑ} 进行加密。$\mathcal{B}$ 构造密文 CT_1 中的 $\tilde{C}$ 和 C。

$$\tilde{C} = m_{\vartheta} e(g,g)^{\alpha s'} = m_{\vartheta} e(g,g)^{(ab + \chi')s'} = m_{\vartheta} Z e(g,g)^{\chi' s'}$$

$$C = g^{s'} = S$$

然后，将 CT_1 发送到雾节点，雾节点设 S 为访问控制树 T^* 的根节点的值，根据访问控制树 T^* 的构造方法对每个叶子节点属性 $a_{j,i} \in T^*$，计算 $C_{j,i} = g^{v_j s_i}$，g^s，则 $C' = g^{s'} g^s = S g^s$，$C_{\text{fg}} = g^{A_{\text{fg}} S_{\text{fg}}}$ 得到密文 CT，雾节点运行重加密算法将 CT 更新为

$$\text{CT}' = \{T^*, \tilde{C}, C, C', C_{\text{fg}}, \{C_{j,i} = g^{v_j s_i E_j}, \forall a_{j,i} \in T^*\}\}$$

挑战者 $\mathcal{B}$ 将密文返还给 $\mathcal{A}$。

阶段 2 同阶段 1，敌手 $\mathcal{A}$ 可以继续向挑战者 $\mathcal{B}$ 进行询问。

猜测 游戏中如果 $\vartheta' = \vartheta$，挑战者 $\mathcal{B}$ 输出 0，表明 $Z = e(g,g)^{abs}$；否则，$\mathcal{B}$ 输出 1，表明 $Z = R$。当 $Z = e(g,g)^{abs}$ 时，$\mathcal{A}$ 获得有效的密文，则 $\mathcal{A}$ 的优势为

$$|\Pr[Q(g, g^a, g^b, g^s, Z = e(g,g)^{abs}) = 0] = \frac{1}{2} + \varepsilon$$

当 $Z = R$ 时，$\mathcal{A}$ 无法获得任何明文信息，即

$$|\Pr[Q(g, g^a, g^b, g^s, Z = R) = 0] = \frac{1}{2}$$

因此，φ 的优势为

$$\frac{1}{2}|\Pr[B(g, g^a, g^b, g^s, Z = e(g,g)^{abs}) = 0] + \Pr[B(g, g^a, g^b, g^s, Z = R) = 0]| - \frac{1}{2} = \frac{\varepsilon}{2}$$

在以上挑战过程中，假设 DBDH 成立，如果没有敌手在多项式时间内完成，则本文方案是满足选择明文攻击安全的，且该方案具有前后向安全性、抗共谋安全性。

1) 在本文方案属性更新机制中，当添加或者撤销用户属性时，雾节点运行 GKeyGen 算法更新组密钥得到 E_j'，再运行 upGKeyGen 算法对用户私钥进行更新得到 SK'，因为在门限方案中将 E_j' 设置为(S, 2)秘密分享，在该属性用户组内的用户可以利用 Lagrange 插值得出组密钥 $E_j' = f(0) = \sum_{m=1}^{2} f(x_m) \prod_{l=1, l \neq m}^{2} -\frac{x_l}{x_m - x_l}$，所以已撤销属性的用户无法推出 E_j'；同理，添加属性的用户也无法通过 E_j'，在雾节点进行外包解密计算

F_R 时，$F_R=\prod_{a_j\in s}e(C_{j,i},D_j)=\prod_{a_j\in s}e(g^{v_j s_i E_j^{'}},g^{rv_j^{-1}E_j^{-1}})$。因为组密钥无法匹配问题，无法计算得到 $e(g,g)^{rs_L}$，便无法对密文进行解密，所以本文方案满足前向后向安全性。

2) DO 设置访问结构 A 对文件进行加密，只有 DU 的属性 $S\in A$ 时才能解密 $\mathrm{ck}e(g,g)^{\beta s'}$。假设多个未满足访问结构的用户相互勾结交换密钥，组成了满足访问结构中的属性集。根据密钥生成 KeyGen 算法，不同用户的身份标识 r 都是随机数，则产生不同的私钥 $D_j=g^{rv_j^{-1}}$，若互相勾结的用户身份表示分别为 r_1,r_2，通过计算得到 $F_R=e(g,g)^{r_1r_2s_L}$，$A=e(g,g)^{r_1r_2s}$，便无法推出 $e(g,g)^{rs'}$，所以仍然无法对密文进行解密。因此，方案具有抗串谋攻击性。

证毕。

7　性能分析

在性能分析中，对本文方案在理论上和实验上进行分析，证明其可行性。将本文方案与其他方案进行功能比较，结果如表 1 所示。

表 1　　**功能比较**

方案	外包加密	外包解密	属性撤销
文献[12]方案	No	No	No
文献[18]方案	Yes	Yes	No
文献[21]方案	Yes	Yes	Yes
文献[24]方案	No	No	Yes
本文方案	Yes	Yes	Yes

7.1　理论分析

在理论分析中，将本文方案与其他方案在性能和计算开销方面进行了比较。为了更好地理解分析，表 2 总结了所使用的符号，表 3 针对方案中加解密开销与属性更新开销进行了比较。

从表 3 可以看出，文献[12, 24]方案没有进行加解密的外包，用户的计算开销是巨大的，会随着访问结构的复杂化而加大计算量。但使用了模加一致性构建访问控制树的文献[12]方案在用户加密与解密时，计算开销小于使用了线性秘密分享方案（LSSS, linear secret sharing scheme）访问结构的文献[24]方案。其他方案对加解密计算进行了外包，因此用户的计算量是恒定的，与属性以及访问控制结构无关。文献[18]方案在用户解密时直接将对称密钥通过第三方审计平台发送给用户，使用户只需要进行一次对称解密就可以解密密文，虽然解密效率更高，但存在安全隐患，对称密钥如果泄露，数据将会被获取。在用户外包解密方案中，本文方案使用了模加法技术构建访问控制树，减小雾节点外包解密的开销，在外包解密中本文方案是最优的。在属性撤销方面，文献[21]方案为了实现用户属性的即时撤销，在属性发生撤销时，需要对该用户以及其他持有该属性的用户更新私钥，当用户数量庞大且属性需要频繁更新时，计算开销是巨大的。文献[24]方案为了实现撤销机制，将撤销方法嵌入加解密中，使用户所持有属性在加解密时都要进行是否撤销的判断操作。本文方案通过雾节点对密文进行重加密便可以实现对属性的即时撤销。综上所述，本文方案为实现外包加解密和属性更新，在保证安全性的前提下，计算开销与其他方案相比较低。

表 2　　**性能分析中使用的符号**

符号	含义
n_1	访问树中的属性个数或者矩阵中的行数
n_2	用户持有属性个数
n_3	与用户关联属性个数
n_4	满足访问树中的最小节点
t_e	双线性配对运算
$G_i(i=0,T)$	乘法循环群内的幂运算或乘法运算
H	哈希运算
δ	对称解密

7.2　实验分析

通过仿真实验将本文方案与文献[18, 21, 24]方案在外包加密、外包解密与属性更新等方面进行比较分析。本文在 charm-crypto 框架内进行仿真实验，使用英特尔 i5-4200U 2.3 GHz 处理器，8 GB 内存，搭建环境为 Ubuntu14.04 64 位系统与 Python2.7。实验中所有结果均为 10 次结果的平均值。

用户加解密时间的比较如图 4 所示。从图 4 可以看出，没有涉及外包加解密的文献[24]方案随着属性数量的增多，用户加密时间线性增长；本文方案与其他外包加解密方案在用户进行加解密操作时，将复杂计算外包出去计算，用户所消耗的时间是基本恒定的，因此不会随着属性数量的增加而增长。在图 4(a)中，在用户加密的效率方面使用了外包加密的方案所消耗时间较相近，可以看到本文

表 3 计算开销比较

方案	用户加密	外包加密	外包解密	用户解密	属性撤销
文献[12]方案	$(n_1+2)G_0$	×	×	$(n_3+1)t_e+n_3G_T$	×
文献[18]方案	$2G_0+G_T+H$	$(n_1+1)G_0$	$(n_3+3)t_e+n_4G_T$	δ	×
文献[21]方案	$4G_0+2G_T$	$(n_1+2)G_0$	$(n_3+2)t_e+(2n_4+2)G_T$	t_e+2G_T	n_3G_0
文献[24]方案	$(2n_1+3)G_0$	×	×	$(2n_4+3)t_e+(T+2)G_T$	n_2G_0
本文方案	$2G_0+2G_T$	$(n_1+2)G_0$	$(n_3+2)t_e$	t_e+G_T	G_0

注释：×表示方案中不涉及该项功能。

方案是较优的。在图 4(b)中，文献[18]方案在设计外包解密时，由第三方审计直接将对称加密密钥发送给用户，用户在解密时只需要进行一次对称解密，所以用户解密时间要低于本文方案，但是直接将对称加密的密钥进行传输，会带来一些安全性问题。在安全性的前提下，本文方案用户解密效率较高。

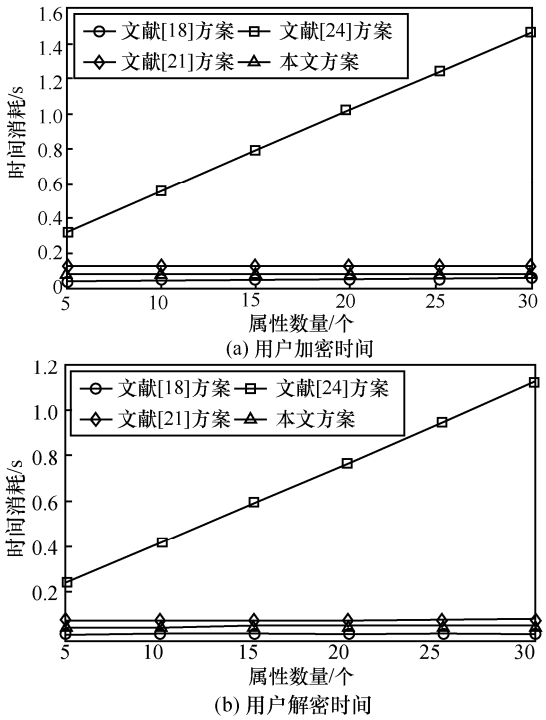

图 4 用户加解密时间的比较

外包加解密时间的比较如图 5 所示。从图 5 可以看出，外包加密时方案之间的差异较小，时间消耗随着属性数量的增加线性增长，所以访问控制策略越复杂加密时间也会越大。外包解密中，因为本文方案使用 UCCMA 技术建立访问控制树，避免了多项式插值法产生的巨大计算开销，计算时间上低于其他方案，随着属性数量的不断增多，这种优势将逐渐扩大。

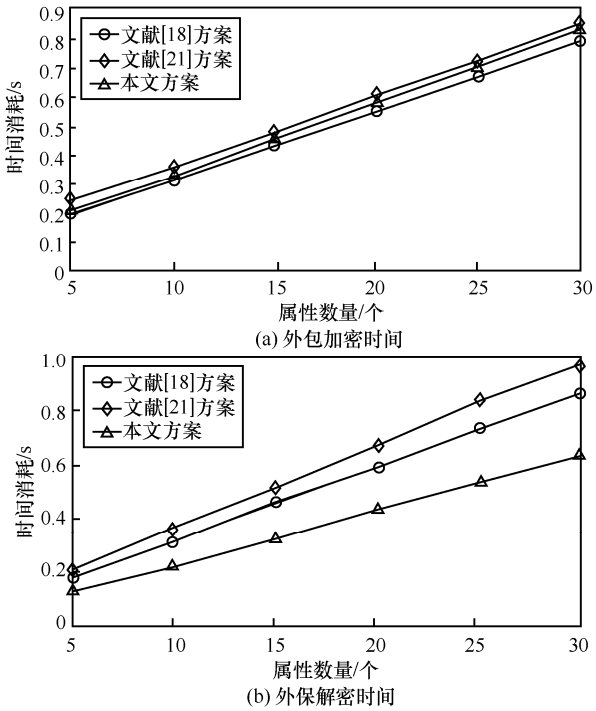

图 5 外包加解密时间的比较

属性更新时间的比较如图 6 所示。仿真实验中，假设所有的用户都拥有要更新的属性，图 6(a)中设定用户数量为 30，图 6(b)中设定更新次数为 2 次。文献[24]方案为了实现对属性的撤销，将属性撤销嵌入加解密过程中，导致用户在进行加解密操作时所持有属性都需要进行计算，使用户在加解密时开销太大，所以在此不做比较。文献[21]方案在属性更新时，为了实现属性的即时更新，需要立即对所有持有该属性的用户进行私钥的更新，随着用户数量的增多，以及属性更新次数的增加，该方案在属性更新时的开销会越来越大。本文方案实现对属性的即时更新，只需要改变属性对应组密钥并对密文进行重加密操作，因为用户由不同雾节点管理，更新用户私钥时只需要对当前雾节点内用户私钥进行更新，当短时间内属

性频繁更新时，未与雾节点进行交互的用户只需要更新一次私钥即可。图 6(a)中用户数量恒定，随着更新次数的增加本文方案时间消耗趋于稳定，文献[21]方案呈线性增加趋势。图 6(b)中更新次数为 2 次时，随着用户数量的增加，本文方案与文献[21]方案均呈线性增加趋势，但本文方案增加速率相对较慢。所以本文方案在属性更新方面效率较高。

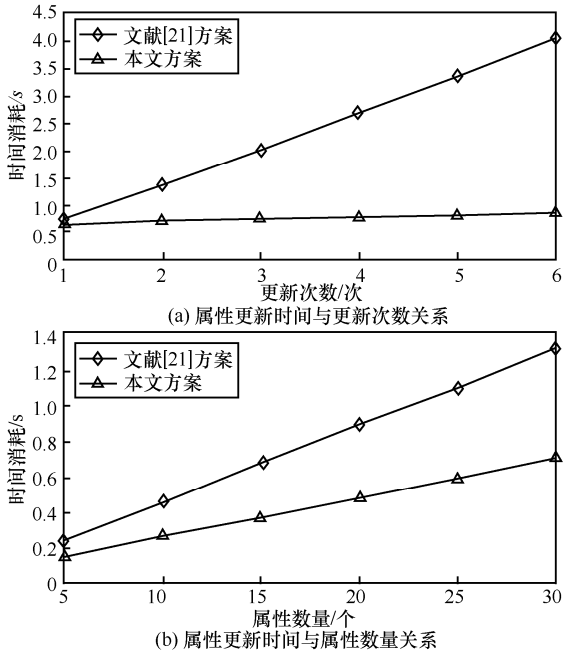

图 6　属性更新时间的比较

8　结束语

本文提出了一种雾计算中细粒度属性更新的外包计算访问控制方案，将加解密的复杂运算外包给雾节点，降低了用户的计算开销，并通过重加密的方法实现了属性的细粒度更新，能够根据需求完成对属性的添加与撤销。在安全性的前提下保证了其高效率，通过决策双线性 Diffie-Hellman 假设分析了安全性，通过仿真实验计算了在用户加解密、外包加解密和属性更新时的计算开销，得出本文方案中的用户终端设备开销较小，属性更新时效率较高，更适用于雾计算。

本文还存在一些不足，方案中通过雾节点对用户进行管理，没有涉及雾节点域的划分，并且随着万物互联的发展，海量边缘设备的增多，存在虚假节点或用户骗取数据的情况。在用户跨域访问时实现跨域的认证至关重要，因此雾环境下用户如何跨域认证实现动态高效且安全的访问控制是下一步需要研究的内容。

参考文献：

[1] HABIBI P, FARHOUDI M, KAZEMIAN S, et al. Fog computing: a comprehensive architectural survey[J]. IEEE Access, 2020, 8: 69105-69133.

[2] GUO R, ZHUANG C, SHI H, et al. A lightweight verifiable outsourced decryption of attribute-based encryption scheme for blockchain-enabled wireless body area network in fog computing[J]. International Journal of Distributed Sensor Networks, 2020, DOI: 10.1177/1550147720906796.

[3] JIANG J F, TANG L Y, GU K, et al. Secure computing resource allocation framework for open fog computing[J]. The Computer Journal, 2020, 63(4): 567-592.

[4] SHAHID M H, HAMEED A R, ISLAM S U, et al. Energy and delay efficient fog computing using caching mechanism[J]. Computer Communications, 2020, 154: 534-541.

[5] DESIKAN K E S , KOTAGI V J, MURTHY C S R. Topology control in fog computing enabled IoT networks for smart cities[J]. Computer Networks, 2020, 167: 107270.

[6] VILELA P H, RODRIGUES J J P C, RIGHI R R, et al. Looking at fog computing for E-health through the lens of deployment challenges and applications[J]. Sensors, 2020, 20(9): 2553.

[6] 李琦, 朱洪波, 熊金波, 等. mHealth 中可追踪多授权机构基于属性的访问控制方案[J]. 通信学报, 2018, 39(6): 1-10.
LI Q, ZHONG H B, XIONG J B, et al. Multi-authority attribute-based access control system in mHealth with traceability[J]. Journal on Communications, 2018, 39(6): 1-10.

[7] ALEISA M A, ABUHUSSEIN A, SHELDON F T. Access control in fog computing: challenges and research agenda[J]. IEEE Access, 2020, 8: 83986-83999.

[8] ZHANG P Y, ZHOU M C, FORTINO G. Security and trust issues in fog computing: a survey[J]. Future Generation Computer Systems, 2018, 88: 16-27.

[9] BETHENCOURT J, AMIT S, WATERS B. Ciphertext-policy attribute based encryption[C]//2007 IEEE Symposium on Security & Privacy. Piscataway: IEEE Press, 2007: 321-334.

[10] WANG H, ZHENG Z H, WU L, et al. New large-universe multi-authority ciphertext-policy ABE scheme and its application in cloud storage systems[J]. Journal of High Speed Networks, 2016, 22(2): 153-167.

[11] LIANG K T, SUSILO W. Searchable attribute-based mechanism with efficient data sharing for secure cloud storage[J]. IEEE Transactions on Information Forensics and Security, 2015, 10(9): 1981-1992.

[12] IBRAIMI L, TANG Q, HARTEL P H. Efficient and provable secure ciphertext-policy attribute-based encryption schemes lecture notes in computer science[C]//2009 International Conference on Information Security Practice and Experience(ISPEC). Berlin: Springer, 2009: 1-12.

[13] WANG H Q, HE D B, HAN J G. VOD-ADAC: anonymous distributed fine-grained access control protocol with verifiable outsourced decryption in public cloud[J]. IEEE Transactions on Services Computing, 2017, 13(3): 572-583.

[14] WANG H Q, HE D B, YU J, et al. Incentive and unconditionally anonymous identity-based public provable data possession[J]. IEEE Transactions on Services Computing, 2016, 12(5): 824-835.

[15] JIN Y, TIAN C, HE H, et al. A secure and lightweight data access control scheme for mobile cloud computing[C]//2015 IEEE Fifth International Conference on Big Data and Cloud Computing. Piscataway: IEEE Press, 2015: 172-179.

[16] GREEN M, HOHENBERGER S, WATERS B. Outsourcing the decryption of ABE ciphertexts[C]//Proceedings of the 20th USENIX Conference on Security. Berkeley: USENIX Association, 2011: 523-538.

[17] MAO X P, LAI J Z, MEI Q X, et al. Generic and efficient constructions of attribute-based encryption with verifiable outsourced decryption[J]. IEEE Transactions on Dependable and Secure Computing, 2016, 13(5): 533-546.

[18] KUMAR P P, KUMAR P S, ALPHONSE P J A. A new verifiable outsourced ciphertext-policy attribute based encryption for big data privacy and access control in cloud[J]. Journal of Ambient Intelligence and Humanized Computing, 2019, 10: 2693-2707.

[19] KIBIWOTT K P, FENGLI Z, ANYEMBE O A, et al. Secure cloudlet-based ehealth big data system with fine-grained access control and outsourcing decryption from ABE[J]. International Journal of Network Security, 2018, 20(6): 1149-1162.

[20] AL-DAHHAN R R, SHI Q, LEE G M, et al. Revocable, decentralized multi-authority access control system[C]//2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion. Piscataway: IEEE Press, 2018: 220-225.

[21] ZHANG P, CHEN Z, LIU J K, et al. An efficient access control scheme with outsourcing capability and attribute update for fog computing[J]. Future Generation Computer Systems, 2018, 78(2): 753-762.

[22] CHEN N, GERLA M, HUANG D, et al. Secure, selective group broadcast in vehicular networks using dynamic attribute based encryption[C]//2010 The 9th IFIP Annual Mediterranean Ad Hoc Networking Workshop. Piscataway: IEEE Press, 2010, DOI: 10.1109/ MEDHOCNET.2010.5546877.

[23] HUR J, NOH D K. Attribute-based access control with efficient revocation in data outsourcing systems[J]. IEEE Transactions on Parallel and Distributed Systems, 2011, 22(7): 1214-1221.

[24] LIU Z H, DUAN S H, ZHOU P L, et al. Traceable-then-revocable ciphertext-policy attribute-based encryption scheme[J]. Future Generation Computer Systems, 2019, 93: 903-913.

[25] 张凯, 马建峰, 李辉, 等. 支持高效撤销的多机构属性加密方案[J]. 通信学报, 2017, 38(3): 83-91.
ZHANG K, MA J F, LI H, et al. Multi-authority attribute-based encryption with efficient revocation[J]. Journal on Communications, 2017, 38(3): 83-91.

[作者简介]

杜瑞忠（1975− ），男，河北献县人，博士，河北大学教授、博士生导师，主要研究方向为可信计算、信息安全等。

闫沛文（1994− ），男，河北张家口人，河北大学硕士生，主要研究方向为信息安全、访问控制、雾计算等。

刘妍（1994− ），女，河北保定人，河北大学硕士生，主要研究方向为网络安全、物联网安全、边缘计算等。