云计算中基于认证数据结构的数据外包认证模型

徐剑[1,2]，周福才[1]，陈旭[1]，朱志良[2]

（1. 东北大学 软件学院，辽宁 沈阳 110004；2. 东北大学 信息科学与工程学院，辽宁 沈阳 110004）

摘 要：利用认证数据结构(ADS, authenticated data structures)的安全特性，分析并设计了面向云计算的基于 ADS 的数据外包认证模型，给出了模型的形式化定义、数据查询认证协议与数据更新认证协议；对 ADS 在模型实际应用时遇到的关键问题进行分析，设计了扩展数据一致性证据生成算法和扩展验证算法，从而实现了 ADS 在模型中的有效融入。最后从安全性和效率两方面对模型的性能进行分析比较，结果表明模型以较高效率实现了数据的正确性与一致性认证。

关键词：云计算；认证数据结构；数据外包；认证跳表

中图分类号：TP309.2 **文献标识码**：A **文章编号**：1000-436X(2011)07-0153-08

Data outsourcing authentication model based on authenticated data structures for cloud computing

XU Jian[1, 2], ZHOU Fu-cai[1], CHEN Xu[1], ZHU Zhi-liang[2]

(1. Software College, Northeastern University, Shenyang 110004, China;

2. College of Information Science and Engineering, Northeastern University, Shenyang 110004, China)

Abstract: The outsourcing authentication model based on authenticated data structures was proposed. The formal definition, data query authentication protocol and data updating authentication protocol of the model was presented. The crucial problems when the authenticated data structures are being used were analyzed. With designing the new extended coherence proof generation algorithm and the new extended verifying algorithm, authenticated data structures in the model very well was used. Finally the model with others through the performance analysis including the security and efficiency, the results of which show that: the model is more efficient than others as well as the consistency and correctness requirements are also guaranteed.

Key words: cloud computing; authenticated data structures; data outsourcing; authenticated skip list

1 引言

云计算[1~3]本质上是一种新的提供资源按需租用的服务模式，是一种新型的互联网数据中心业务[4]。数据外包[5]是云服务计算模式中的一种，是指用户将己方数据信息的产生、管理和维护的工作外包给专业的云计算服务提供商。用户可以通过云计算服务商所提供的服务接口来进行远程的数据管理，从而达到优化本地资源配置，降低存储资源投资成本的目的。但事实表明，由于存在内部人员失职、敌手攻击及系统故障等多种安全风险，云计算服务商无法提供充足的证据让用户确信其外包的数据没

收稿日期：2011-02-28；**修回日期**：2011-06-08

基金项目：国家高技术研究发展计划（“863”计划）基金资助项目（2009AA01Z122）；国家自然科学基金资助项目（60872040）；沈阳市自然科学基金资助项目（F10-205-1-12）

Foundation Items: The National High Technology Research and Development Program of China (863 Program) (2009AA01Z122); The National Natural Science Foundation of China (60872040); The Natural Science Foundation of Shenyang (F10-205-1-12)

有被删除或恶意篡改，即其无法向用户提供安全的数据外包认证服务。由于上述原因，云数据管理、云存储等云计算服务模式的推广和使用受到安全问题的极大制约[6]。

在云计算环境中，由于大规模数据所导致的巨大通信代价，用户不可能将所有数据下载到本地后再进行真实性验证。因此，云用户需要通过下载较少的数据，并依赖某种计算方法或知识证明协议来实现高置信概率的数据完整性判断。目前，典型的研究工作包括: Juels 等学者提出的面向用户单独验证的数据可检索性证明（POR）方法[7]；Ateniese 等人提出的面向非可信存储的可证明数据拥有方法（PDP 方法）[8]；为提高已有方法的性能，Shachamd 等学者提出了基于 Merkel 散列树的 POR 方法的改进方法[9]；但文献[8,9]提出的方法都不能支持数据的动态变化，因此 Ateniese 等人又提出新的支持数据动态变化的改进 PDP 方法[10]。以上方法都在一定程度上解决了云计算环境中的数据外包认证问题，但却存在不能很好地支持动态数据变化（文献[8]和文献[9]仅支持追加操作，文献[10]支持追加、删除和修改操作，但是不支持对象插入操作）的问题。

认证数据结构[11](ADS, authenticated data structures)是在认证字典[12](authenticated dictionary)基础上发展起来的一种新型的分布式计算模型，能够很好地解决分布式环境下的数据认证问题[13]，即可以在响应者（数据发布代理）不可信的情况下，保证响应者发布的数据和数据源是一致的，从而使得用户不必和数据源建立任何通信，仅和响应者建立通信联系就能检验出响应者是否进行了数据篡改。

利用 ADS 的安全特性，并结合云环境中数据外包实际安全需求，本文设计了基于 ADS 的数据外包认证模型。该模型采用了 ADS 的核心设计思想——“签名摊销”[13]，并对 ADS 的相关认证方法和关键算法进行改进，使其适应云计算中的数据外包环境。认证跳表[14]（ASL, authenticated skip list）是 ADS 的典型的高效实现方案（计算和通信复杂度为 $O(\log n)$），因此本文仍采用 ASL 作为模型的实现方案，但为满足云环境中数据外包的实际需求，本文对 ASL 中的数据一致性证据生成算法和验证算法进行改进，提出了满足数据外包认证需求的新的扩展数据一致性证据生成算法和扩展验证算法。最后，通过模拟敌手攻击手段，指出本文所提出的认证模型可以满足云环境中数据外包服务的正确性和一致性需求。

2 认证数据结构

下面首先给出认证数据结构模型的定义描述。

定义 1 认证数据结构模型（ADSM, authenticated data structures model）（如图 1 所示）包括以下实体：可信数据源 S、不可信响应者（数据发布代理）R 以及用户 U，并且包括定义在结构化数据集上的查询、更新和验证等操作。因此，ADSM 可以由以下元组表示：

ADSM{S, R, U, KeyGen, BasisGen, ProofGen, Verify}

其中，KeyGen 是密钥对（公钥/私钥）产生算法；BasisGen 是数据正确性根据产生算法；ProofGen 是数据一致性证据产生算法，Verify 是数据真实性验证算法。

“签名摊销”是认证数据结构模型的核心技

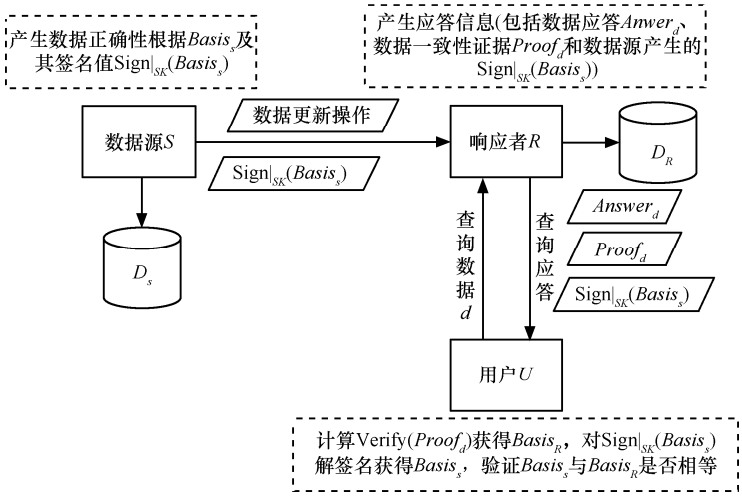


图 1 认证数据结构模型

术，所谓签名摊销是指一个结构化数据集上的所有数据项分摊一个签名，即通过密码学手段在一个数据集上提取出整个结构的摘要，对这个摘要进行签名以保证整个集合的完整性。

基于 ADSM 的分布式数据查询认证过程如下。

1) 数据源 S 是动态结构化数据集 D_S 的拥有者，利用基于签名摊销的数据正确性根据产生算法 BasisGen 计算 D_S 的一个整体性的结构特征值 $Basis_S$ = BasisGen (D_S)（$Basis_S$ 是数据正确性根据，满足抗碰撞性和不可逆性，并且它的产生与 D_S 中的每个元素都密切相关，D_S 中某一数据或结构发生 1bit 的变化，都将引起 $Basis_S$ 值的巨大变化）。

S 调用算法 KeyGen，输入为安全参数 1^k，输出为公私钥对(PK,SK)。S 用自己的私钥 SK 签名 $Basis_S$，之后将签名值 $\mathrm{Sign}|_{SK}(Basis_S)$发送给响应者 R。$Basis_S$ 和 $\mathrm{Sign}|_{SK}(Basis_S)$随着 D_S 的动态改变而改变，每次改变时 S 都要向响应者 R 发送 D_S 的对应更新以及新的 $\mathrm{Sign}|_{SK}(Basis_S)$，以确保 R 所保存的数据副本 D_R 与 D_S 相一致。

2) 响应者 R 维护 D_S 的副本 D_R 和经 S 签名后的 $Basis_S$ 的签名值 $Sign|_{SK}(Basis_S)$。R 与 S 周期性交互通信，从而同步获取 D_S 的更新信息以及新的 $\mathrm{Sign}|_{SK}(Basis_S)$。当用户 U 向 R 提交关于数据 d 的真实性查询认证请求时，R 在 D_R 上执行查询操作，并对数据 d 应用一致性证据产生算法 ProofGen(D_R, d)，将应答信息返回给 U，其中包括查询应答 $Answer_d$、数据的一致性证据 $Proof_d$ 和 $\mathrm{Sign}|_{SK}(Basis_S)$。

3) 用户 U 向不可信的响应者 R（不是可信的数据源 S）提出查询请求，因此，U 要确保能够验证从 R 处获得的数据 d 是否与 D_S 中的相一致，U 对应用验证算法 Verify($Proof_d$) 对 $Proof_d$ 进行验证，所得结果为 D_R 的数据正确性根据 $Basis_R$，同时利用 S 的公钥 SK 计算 $\mathrm{Sign}|_{PK}(Basis_S)$得 $Basis_S$，并通过比较 $Basis_S$ 和 $Basis_R$ 是否相等来判断所获取的查询结果是否真实可信。

3　基于 ADS 的数据外包认证模型

3.1　模型设计

下面首先给出云环境中基于 ADS 的数据外包认证模型的定义。

定义 2　基于 ADS 的数据外包认证模型(DOAM-ADS, data outsourcing authentication model based on authenticated data structures)（如图 2 所示）包括数据外包服务器(DOSer, data outsourcing server)、客户 C 和结构化数据集 D_C，以及定义在 D_C 上的查询、更新、验证等操作。DOAM-ADS 可以由以下元组表示：

DOAM-ADS
$\{\mathrm{DOSer}, C, D_C, \mathrm{ProofGen}, \mathrm{ProofEGen}, \mathrm{Verify}\}$

1) 客户 C 仅存储动态结构化数据集 D_C 唯一的密码学计算结果，即数据正确性根据 $Basis_{D_C}$，$Basis_{D_C}$ =BasisGen (D_C)为 D_C 的结构特征值，其大小为常数。

2) 客户 C 将数据集 D_C 外包给一个云计算环境下的数据外包服务器 DOSer。

3) 客户 C 可以通过 DOSer 提供的接口，对 DOSer 上的数据集合 D_C 进行操作，并通过相关的密码学操作来更新和维护自己的数据正确性根据 $Basis_{D_C}$。

4) 在任何时候，客户 C 通过 DOSer 发起对某一数据的查询认证，计算得到 DOSer 上保存的 D_C 的结构特征 $Basis_{D_C}$，这个 $Basis_{D_C}$ 和客户 C 先前保存在本地的 $Basis_{D_C}$ 记录应是一致的，本文将这种特性称为数据结构特征值的一致性，而这种特征值的一致性可以通过单向散列函数来实现。

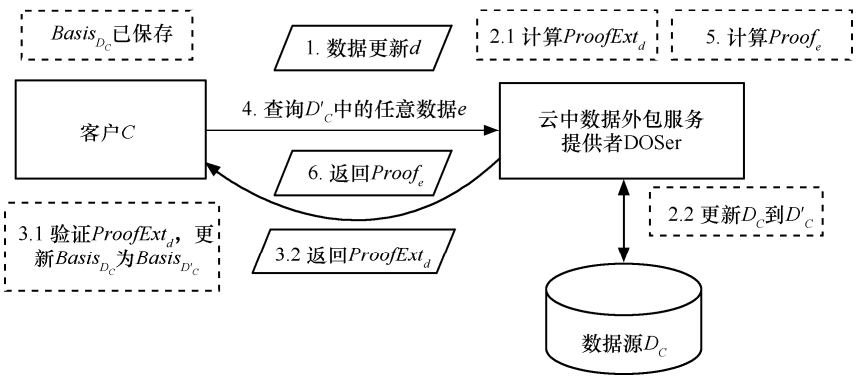


图 2　基于 ADS 的数据外包认证模型

3.2 DOAM-ADS 中的协议设计

根据 DOAM-ADS 模型架构，本文设计了满足模型认证需求的数据查询认证协议和数据更新认证协议。

3.2.1 数据查询认证协议

Step1 客户 C 保存 D_C 的 $Basis_{D_C}$，然后向 DOSer 提出关于 D_C 中某一数据 d 的查询认证请求。

Step2 DOSer 获得客户 C 提交的关于 d 的查询认证请求后，调用服务器端的 ProofGen(d, D_C)，产生数据的一致性证据 $Proof_d$，并将其返回给客户 C，即

$$Proof_d \leftarrow \text{ProofGen}(d, D_C)$$

Step3 客户 C 执行 Verify 验证算法，该算法的输入为 C 存储在本地的当前 $Basis_{D_C}$，数据 d 和对应的数据一致性证据 $Proof_d$，Verify 算法向客户 C 提供数据 d 以及整个数据集 D_C 的真实性认证结果。若验证通过，即 Verify($d, Proof_d, Basis_{D_C}$) 的结果为真，此时客户 C 可以认为 D_C 是真实的，否则表明 D_C 已遭非法篡改。即

$$\{yes, no\} \leftarrow \text{Verify}(d, Proof_d, Basis_{D_C})$$

3.2.2 数据更新认证协议

Step1 客户 C 存储 D_C 的 $Basis_{D_C}$，向数据外包服务器 DOSer 提交关于数据 d 的更新请求。

Step2 DOSer 执行客户 C 所提交的更新（插入/删除）请求，即 DOSer 将 D_C 更新为 D_C'，并调用服务器端算法 ProofEGen(d, D_C, D_C') 产生 $ProofExt_d$ 并将其发送给客户 C，即

$$ProofExt_d \leftarrow ProofEGen(d, D_C, D_C')$$

其中，$ProofExt_d$ 称为扩展的数据一致性证据，ProofEGen 称为扩展的数据一致性证据生成算法，本文将在 3.2 节和 3.3 节中给出 $ProofExt_d$ 和 ProofGen 算法的详细分析与设计过程。

Step3 客户 C 执行 Verify 验证算法，该算法的输入为 C 存储在本地的当前 $Basis_{D_C}$，数据 d 和相应的扩展数据一致性证据 $ProofExt_d$，同时，生成新的 $Basis_{D_C'}$。Verify 算法向客户 C 提供关于数据集 D_C 的真实性认证结果，同时帮助客户 C 决定是否接受或拒绝更新原有的 $Basis_{D_C}$。若 $Basis_{D_C}$ 经验证通过，即表示 Verify($d, ProofExt_d, Basis_{D_C}$) 的结果为真，说明 D_C 是真实而未遭篡改的。此时客户 C 更新本地 $Basis_{D_C}$ 为新的 $Basis_{D_C'}$。否则客户 C 拒绝更新 $Basis_{D_C}$。即

$$\{(yes, Basis_{D_C'}), (no, \perp)\} \leftarrow \text{Verify}(d, ProofExt_d, Basis_{D_C})$$

Step4 Step3 中验证的实际是数据 d 更新前的数据集合，即数据集 D_C 的真实性，即使 Verify($d, ProofExt_d, Basis_{D_C}$) 的验证结果为真，仅能表明 D_C 未遭到篡改，而仍无法确定出 D_C' 的真实性。为验证 D_C' 的真实性，客户 C 还需在验证算法 Verify 执行完成后，对 D_C' 进行数据查询认证。若在数据查询认证过程中对 D_C' 中的任意元素 e 的查询认证结果为真，则说明 D_C' 的真实性得到认证，关于数据 d 的更新才正式完成；否则表明关于数据 d 的更新已遭非法篡改，更新操作将回滚，D_C' 被恢复为 D_C，客户端 $Basis_{D_C'}$ 回滚到 $Basis_{D_C}$，协议结束。

3.3 DOAM-ADS 中关键问题分析

下面对 ADS 在 DOAM-ADS 应用时需要考虑的关键问题进行分析。

1) 在模型初始化阶段，要保证客户 C 所保存的 $Basis_{D_C}$ 是正确的，因为以后的每次认证计算都要依赖于上一次的 $Basis_{D_C}$。

2) 对于 Verify($d, Proof_d, Basis_{D_C}$) 和 Verify(d, $ProofExt_d, Basis_{D_C}$) 来说，数据 d 均是必须参与运算的要素，并且数据 d 必须是客户端 C 所需的关键数据的值。数据及数据一致性证据（扩展数据一致性证据）共同参与验证计算可以保证 d 和 $Proof_d$ 或 $ProofExt_d$ 始终密切关联，这样可以防止服务器端造假。若数据 d 被篡改为 d'，即使 $Proof_d$ 或 $ProofExt_d$ 是真实的，客户 C 仍然可以验证数据 d' 已遭篡改。反之，若 Verify()的参数不包含数据 d，而只验证 $Proof_d$ 或 $ProofExt_d$ 是否与 $Basis_{D_C}$ 相匹配，那么服务器端将可以利用真实的 d 生成真实的 $Proof_d$ 或 $ProofExt_d$ 让客户验证通过，而同时又把 d 篡改为 d'。

3) Verify($d, ProofExt_d, Basis_{D_C}$) 验证机制

① $Basis_{D_C'}$(for verify) $\leftarrow$ Verify()

② if ($Basis_{D_C'}$(for verify) == $Basis_{D_C}$)

$\{(yes, Basis_{D_C'})\} \leftarrow$ Verify() *else* $\{(no, \perp)\} \leftarrow$ Verify()

对 $ProofExt_d$ 进行数据正确性根据 $Basis_{D_C}$ 的验证计算，即产生服务器端在更新操作之前的 D_C' 的 $Basis_{D_C'}$，用来验证是否和客户 C 上次保存在本地的 $Basis_{D_C}$ 相一致，若是一致的，则可以通过 Verify()

来计算 $Basis_{D_C'}$，即服务器端在更新操作之后产生 D_C"的特征值。

3.4 DOAM-ADS 中关键算法的分析与设计

DOAM-ADS 仍采用认证数据结构中最常用的认证跳表（ASL）作为模型的实现方案。这是由于 ASL 算法中的元素插入和删除、节点特征值计算、查询和验证都与本文提出的认证模型相匹配。因此，下面将从模型中的几个关键点入手，对 ASL 在 DOAM-ADS 中的应用加以分析与设计（关于 ASL 特征值计算、元素插入、删除和查询验证等算法可参见文献[14]）。

1) 针对 3.3 节中的关键问题 1，可以采用如下处理方法：在模型初始化阶段，外包服务器 DOSer 在客户 C 所存储的 D_C 为空，此时 DOSer 基于空 D_C 建立的跳表仅有 2 个元素，即左哨兵和右哨兵。令客户 C 所存储的 $Basis_{D_C}$ 内容是一个常数，该常数对所有的客户都是完全相同的，它的值为这个跳表（只有左右哨兵 2 个元素）的结构特征值。因此，客户 C 可以在第一次数据更新操作完成后，保证调用验证算法的第一步骤得到的 $Basis_{D_C}$ 均能够通过验证（参见 3.3 节中 Verify()的验证制）。

2) ASL 验证算法（参见文献[14]）可以满足 3.3 节中的关键问题 2 对于 Verify()所要求的条件。通过元素 x 的 $Proof_x$ 序列计算算法与 ASL 的验证算法对比可知，$Proof_x$ 序列$\{x_1,x_2,\cdots,x_{j-1},x_j\}$略去了元素 x 的节点值，就是为了达到在 Verify()中增加参数 x 的目的，以便在验证时将 $Proof_x$ 序列和元素 x 紧密地联系在一起，防止跳表结构为真而元素 x 造假的情况。

3) 已有认证跳表算法不能很好地处理扩展的数据一致性证据的问题，因此，为了将 ADS 更好地应用到数据外包认证模型中，针对 3.3 节中的关键问题 2，本文设计了新的 ProofEGen 算法来处理扩展的数据一致性证据的产生问题，同时对 ASL 原有的验证算法也进行了相应的改进和扩展。

3.4.1 ProofEGen 算法

本文所设计新的 ProofEGen 算法，其目的是，既能得出更新操作前的 D_C' 的特征值 $Basis_{D_C'}$，又可以获得 DOSer 在更新操作后所生成的 D_C"的特征值，因此算法需要包括更新操作之前与之后的 $Proof_d$ 和 $Proof_d'$ 序列中的所有散列值。

由 ASL 查询算法的实现过程可知，$Proof_d$ 和 $Proof_d'$ 序列中存在很多散列值是重复的。因此可以用一种简单方法来求解 ProofGen 算法的输出结果 $ProofExt_d$：将更新操作前后的 $Proof_d$ 和 $Proof_d'$ 序列进行归并运算，联合后的结果就可作为 $ProofExt_d$。

$ProofExt_d$ 由以下两部分构成。

1) $Proof_d$ 和 $Proof_d'$ 序列归并运算后的结果。

2) 向量组 $ProofExtflag \in \{0,1,2\}^n$，$ProofExtflag$ 向量用来表示归并结果，该向量长度是 $ProofExt_d$ 序列中的散列值数目，向量中的每一个元素在集合{0,1,2}中取值，0 表示该元素仅出现在 $Proof_d$ 序列中；1 表示仅出现在 $Proof_d'$ 序列中；2 表示同时出现在这 2 个序列之中。

算法 1　ProofEGen 算法

```
Sequence ProofEGen (Sequence origin,
Sequence newproof, Sequence& proofExtflag)
Sequence ProofExt=new Sequence ();
int pre, current=0;
for(int i=0;i<origin.size(); i++)
  int j=pre;
      for(;j<newproof.size(); j++)
if((origin.get(i)).equals(newproof.get(j)))
                current =j;
for(int k=pre;k<current;k++)
            push(Proofextend, newproof.get(k));
push(proofExtflag, '1');
push(Proofextend, newproof.get(current));
push(proofExtflag, '2');
pre=current+1;
break;
if(j==newproof.size())
push(Proofextend, origin.get(i));
push(proofExtflag, '0');
return Proofextend;
```

3.4.2 扩展 Verify 算法

在本节中，将对原有的 ASL 验证算法 Verify 做相应的扩展，令其具有多态性，即可以验证扩展的数据一致性证据，并返回新 $Basis_{D_C'}$，算法输入为查询元素值 x、向量组 $ProofExtflag$ 和扩展的数据一致性证据 $ProofExt_d$。依据 $ProofExtflag$ 中值是 0、1 或 2 的情况，依次在新旧计算结果中累加 $ProofExt_d$ 中所对应的散列值，若通过计算获得的旧结果与 $Basis_{D_C}$ 相同，则返回新的计算结果 $Basis_{D_C'}$，否则返回空值。

算法 2　扩展 Verify 算法

```
String Verify(value x, Sequence ProofExtd, Se-
```

quence ProofExtflag, String Basis)

String result_old= result_new= contractDigest(*x*);

for(*int i*=0;*i*<*ProofExtd.Count*;*i*++)

if(*Proofextendflag*[*i*] == "0"||*Proofextendflag*[*i*] =="2")

result_old=contractDigest(*result_old,*(*String*) *ProofExtd*[*i*]);

if(*Proofextendflag*[*i*]=="1" || *Proofextendflag*[*i*] =="2")

result_new=

contractDigest(*result_new,*(*String*)*ProofExtd*[*i*]);

if (*Basis*== *result_old*)

return *result_new*;

else *return null*。

4 DOAM-ADS 性能分析

4.1 安全性分析

下面将从正确性和一致性 2 个方面分析 DOAM-ADS 的安全性。

定理 1 DOAM-ADS {DOSer, C, D_C, ProofGen, ProofEGen, Verify }表示云计算中安全性参数为 k 的基于 ADS 的数据外包认证模型。update 为更新操作，$Basis_{D_C}$ 表示初始是空，且经过某一系列数据更新操作 update($\{d_1,d_2,\cdots,d_n\}$, D_C)后的数据集 D_C 所对应的数据正确性根据。若 DOAM-ADS 满足以下属性，则表明其是安全的。

1) 正确性。在数据更新操作过程中，如果 $Basis_{D_C}$、数据一致性证据是正确的且和 D_C 状态相一致（可以通过上次数据查询或者更新操作使 $Basis_{D_C}$ 的正确性得到保证），那么就有：

$$ProofExt_d \leftarrow \mathrm{ProofEGen}(d, D_C, D_C') \Rightarrow$$
$$\{yes, Basis_{D_{C'}}\} \leftarrow \mathrm{Verify}(d, ProofExt_d, Basis_{D_C})$$

这里是指，如果 update(d)更新操作被正确执行，且扩展的数据一致性证据 $ProofExt_d$ 是使用算法 ProofEGen 产生的，那么验证算法 Verify 就必然会接受并计算出新的 $Basis_{D_{C'}}$（与新的数据集相一致）。

同理，在数据查询过程中，若 $Basis_{D_C}$ 正确无误且存在 $Proof_d \leftarrow \mathrm{ProofGen}(d, D_C)$，就一定有：

$$\{yes\} \leftarrow \mathrm{Verify}(d, Proof_d, Basis_{D_C})$$

令 $Basis_{D_C} \triangleq \mathrm{BasisG}(D_C)$ 代表 $Basis_{D_C}$ 为真，即：

$$Pr[Basis_{D_C} \triangleq BasisG(D_C); \sigma \leftarrow \mathrm{ProofEGen}(d, D_C, D_C'):$$
$$Basis_{D_C}' \triangleq BasisG(D_C') \wedge \neg \mathrm{Verify}(d, \sigma, Basis_{D_C})] = 0$$

2) 一致性。在数据更新时，假设在多项式时间内存在敌手 A，A 随机访问 ProofGen、ProoEfGen 和 Verify 算法，输入为数据集 D_C 和 update(d, D_C)操作生成 d 的扩展数据一致性证据 $ProofExt_d$，若存在 $\{yes, Basis_{D_{C'}}\} \leftarrow \mathrm{Verify}(d, ProofExt_d, Basis_{D_C})$，并且对于 D_C' 中的任何元素 e（并不与 d 严格区分，在 update 表示增加操作时 e 可以等价于 d），并同时满足 $\{yes\} \leftarrow \mathrm{Verify}(e, Proof_e, Basis_{D_{C'}})$，那么 $Basis_{D_{C'}}$ 与 D_C' 不一致的概率在 $v(k)$是可以忽略的。这表示，假定该多项式时间敌手 A 遵守多项式时间复杂度的协议调用，然后生成扩展的数据一致性证据 $ProofExt_d$，如果 $ProofExt_d$ 对于 update(d, D_C)操作被算法 Verify 所接受，且 $Proof_e$ 在执行 update(d, D_C) 操作后仍被算法 Verify 所接受，则这一操作已被正确执行，且新状态 $Basis_{D_{C'}}$ 与新数据集 D_C' 保持一致。类似可获得数据查询过程中的一致性定义。

令 $Basis_{D_C} \triangleq \mathrm{BasisG}(D_C)$ 代表 $Basis_{D_C}$ 为真,即：

$$Pr[Basis_{D_C} \triangleq \mathrm{BasisG}(D_C); \sigma \leftarrow \mathrm{ProofEGen}(d, D_C, D_C');$$
$$\mathrm{Verify}(d, \sigma, Basis_{D_C}) = 1 \wedge \mathrm{Verify}(e, Proof_e, Basis_{D_{C'}}) = 1:$$
$$\neg Basis_{D_{C'}} \triangleq \mathrm{BasisG}(D_C')] = v(k)$$

由上述定义，若客户 C 的数据集 D_C 初始为空，并通过相关的更新操作，将数据集 D_C 依据数据外包模型中的协议外包给云环境中的数据外包服务器 DOSer，在每次的更新操作中，C 所保存的 $Basis_{D_C}$ 都会以与最终数据集 D_C 相一致的状态而结束。因此，数据集 D_C 与更新操作 update(d, D_C)的历史记录是一致的，并且以后所有的更新操作都将会得到验证。

由上述分析，DOAM-ADS 实现了对结构化数据集的数据正确性证据 *Basis* 的可追踪性，并可以依据 *Basis* 状态来判断整个数据集的真实性以及相应更新操作是否可被接受，从而实现了数据的真实性认证。

4.2 性能分析与比较

由于本认证模型采用了 ADS，因此支持数据集动态变化。同时，客户 C 为存储数据正确性根据所付出存储代价为 $O(1)$。由于采用了认证跳表来实现本模型，因此对大小为 n 的动态数据集、

外包服务器计算代价、客户方验证代价以及通信代价均为 $O(\log n)$。表 1 给出了本文所提出模型和目前典型的数据外包认证模型[8~10]的性能比较结果。

表 1　　性能比较

方案性能	文献[8]	文献[9]	文献[10]	本文方案
支持动态数据	否	否	否	是
服务器计算代价	$O(1)$	$O(1)$	$O(1)$	$O(\log n)$
通信代价	$O(1)$	$O(1)$	$O(1)$	$O(\log n)$
客户方计算代价	$O(1)$	$O(1)$	$O(1)$	$O(\log n)$
客户方存储代价	$O(1)$	$O(1)$	$O(1)$	$O(1)$
存储对象操作（支持追加）	是	是	是	是
存储对象操作（支持插入）	否	否	否	是
存储对象操作（支持删除）	否	否	是	是
存储对象操作（支持修改）	否	否	是	是

由上述比较结果可知，虽然在服务器计算代价、通信代价和客户计算代价方面，本文的方案高于目前的数据外包认证方案(文献[8~10])，但本文所提出的方案支持动态数据集变化，而文献[8~10]中的方案仅只支持静态数据集；对于外包存储的数据对象所能执行的操作，文献[8]和文献[9]的方案仅支持追加操作，文献[10]支持追加、删除和修改操作，但是不支持对象插入操作，而本文提出的方案支持对于外包数据存储对象的全部操作，包括追加、插入、删除和修改。由以上分析可知，文献[8~10]的方案虽然具有较低的计算和通信代价，但是其不能很好地支持动态数据集操作，不能提供对数据外包对象的追加、插入、删除和修改等操作的支持，因此其在云计算环境中难以得到很好的应用，而本文提出的方案虽然在计算和通信代价高于已有方案，但对数级的代价开销仍然是可以接受的，同时由于本文方案支持动态数据集操作，能够实现数据外包对象的追加、插入、删除和修改等操作，因此比以往的方案具有综合优势，更适用于云计算环境。

5　结束语

利用 ADS 的高效率、低代价的安全属性，并结合云环境中的数据外包安全需求，设计了基于 ADS 的数据外包认证模型。文中设计了扩展数据一致性证据生成算法和扩展验证算法，以实现 ADS 在数据外包模型中的应用。对模型的性能进行分析与比较，结果表明该数据外包认证模型不仅支持动态数据集操作，而且支持外包数据存储对象的全部操作，包括追加、插入、删除和修改，这都是目前云环境中其他典型方案所不具备的。虽然，本方案在代价方面略高于已有的方案，但是由于 ADS 的研究在不断深入，因此完全可以设计出更高效率的 ADS 方案已解决其效率较高的问题。同时，对数级的代价开销对于目前的计算机和网络环境来说也是可以接受的，因此综合考虑，本方案可以较好地用来解决云计算中的数据外包认证问题。

参考文献：

[1] 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.
FENG D G, ZHANG M, ZHANG Y, *et al*. Study on cloud computing Security[J]. Journal of Software,2011,22(1):71-83.

[2] 徐小龙, 程春玲, 熊婧夷. 基于 multi-agent 的云端计算融合模型的研究[J]. 通信学报,2010,31(10): 203-211.
XU X L, CHENG C L, XIONG J Y. Conjunct model of cloud & client computing based on multi-agent[J]. Journal on Communications, 2010, 31(10): 203-211.

[3] GOSCINSKI A, BROCK M. Toward dynamic and attribute based publication, discovery and selection for cloud computing[J]. Future Generation Computer Systems, 2010, 26(7): 947-970.

[4] 陈康, 郑纬民. 云计算: 系统实例与研究现状[J]. 软件学报, 2009, 20(5):1337-1348.
CHEN K, ZHENG W M. Cloud computing: system instances and current research[J]. Journal of Software, 2009, 20(5):1337-1348.

[5] 张敏, 洪澄, 陈驰. 一种服务器透明的外包数据库查询验证方法[J]. 计算机研究与发展,2010,47(1): 182-190.
ZHANG M, HONG C, CHEN C. Server transparent query authentication of outsourced database[J]. Journal of Computer Research and Development, 2010, 47(1): 182-190.

[6] KIM A, MCDERMOTT J, KANG M. Security and architectural issues for national security cloud computing[A]. Proceedings of 2010 IEEE 30th International Conference on Distributed Computing Systems Workshops[C]. Genoa, Italy, 2010.21-25.

[7] JUELS A, KALISKI B. Pors: proofs of retrievability for large files[A]. Proceedings of CCS 2007[C]. Alexandria, VA, USA, 2007. 584-597.

[8] ATENIESE G, BURNS R, CURTMOLA R, *et al*. Provable data possession at untrusted stores[A]. Proceedings of CCS 2007[C]. Alexandria, VA, USA, 2007.598-609.

[9] SHACHAM H, WATERS B. Compact proofs of retrievability[A]. Proceedings of ASIACRYPT 2008[C]. Melbourne, Australia, 2008. 90-107.

[10] ATENIESE G, PIETRO D, MANCINI R, *et al*. Scalable and efficient provable data possession[A]. Proceedings of SecureComm 2008[C]. Istanbul, Turkey, 2008. 1-10.

[11] PAPAMANTHOU C, TAMASSIA R, TRIANDOPOULOS N. Optimal authenticated data structures with multilinear forms[A]. Proceedings of Pairing 2010[C].Ishikawa, Japan, 2010.246-264.

[12] 卿斯汉, 周永彬, 张振峰等. 认证字典及其在PKI中的应用研究[J]. 电子学报, 2004, 32(8): 1356-1359.
QING S H, ZHOU Y B, ZHANG Z F, *et al*. Study on authenticated dictionary and its applications in PKI[J]. Acta Electronica Sinica, 2004, 32(8): 1356-1359.

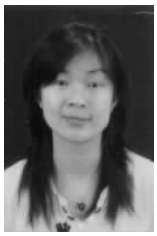
[13] MARTEL C, NUCKOLLS G, DEVANBU P, *et al*. A general model for authenticated data structures[J]. Algorithmica, 2004, 39(1):21-41.

[14] ANAGNOSTOPOULOS A, GOODRICH M T, TAMASSIA R. Persistent authenticated dictionaries and their applications[A]. Proceedings of ISC 2001[C]. Malaga, Spain, 2001.379-393.

作者简介：

徐剑（1978-），男，辽宁沈阳人，东北大学博士生、讲师，主要研究方向为密码学与网络安全、数据与身份认证技术、分布式计算等。

周福才（1964-），男，吉林长春人，博士，东北大学教授、博士生导师，主要研究方向为密码学与网络安全、可信计算、电子商务基础理论与关键技术。



陈旭（1984-），女，辽宁大连人，东北大学硕士生，主要研究方向为密码学与网络安全、数据认证技术等。

朱志良（1962-），男，辽宁沈阳人，博士，东北大学教授、博士生导师，主要研究方向为混沌信息处理与服务计算。