doi:10.11959/j.issn.1000-436x.2016203

支持身份认证的数据持有性证明方案

李昊星[1]，李凤华[1,2]，宋承根[3]，阎亚龙[3]

(1. 西安电子科技大学综合业务网理论与关键技术国家重点实验室，陕西 西安 710071；
2. 中国科学院信息工程研究所信息安全国家重点实验室，北京 100093；3. 北京电子科技学院信息安全研究所，北京 100070)

摘 要：针对云应用场景中身份认证和数据持有性证明的双重需求，提出一种支持身份认证的数据持有性证明方案。基于数据标签签名和随机数复用，新方案通过 3 次交互即可实现用户对云持有数据的完整性验证、用户与云服务器之间的双向身份认证以及会话密钥协商与确认。与使用认证密钥协商和数据持有性证明的组合方案相比，新方案具有较少的运算量和交互轮次以及可证明的安全性。在随机预言机模型下，基于计算性 Diffie-Hellman 问题假设，给出方案的安全性证明。

关键词：认证；数据持有；云计算；可证明安全；随机预言机

中图分类号：TP393 **文献标识码**：A

Provable data possession scheme with authentication

LI Hao-xing[1], LI Feng-hua[1,2], SONG Cheng-gen[3], YAN Ya-long[3]

(1. State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China;
2. State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China;
3. Information Security Institute, Beijing Electronic Science and Technology Institute, Beijing 100070, China)

Abstract: To satisfy the requirements of identity authentication and data possession proven in the cloud application scenarios, a provable data possession scheme with authentication was proposed. Based on data tag signature and randomness reusing, the proposed scheme could accomplish several issues with three interactions, including the possession proof of cloud data, the mutual authentication between user and cloud computing server, the session key agreement and confirmation. Compared to the simple combination of authentication key agreement and provable data possession schemes, the proposed scheme has less computation and interactions, and better provable securities. In the random oracle model, the security proof of the proposed scheme is given under the computational Diffie-Hellman assumption.

Key words: authentication, data possession, cloud computing, provable security, random oracle

1 引言

随着云计算的快速发展，基于云构建的各种应用逐渐融入了人们的工作和生活。云计算利用虚拟化技术实现了多租户共享云资源，提高了硬件资源的利用率，降低了用户的使用成本[1]。云计算为用户提供了高性能、高稳定性、低费用的服务，同时也带来了一些新的安全问题，如云环境中的身份认证、访问控制、数据搜索、数据完整性验证和数据安全删除等。近期接连发生的云计算相关安全事故使云安全问题成为产业界的热门话题和学术界的研究热点[2]。

基于云构建的存储系统中，敏感数据被加密保护后再上传到云服务器，在使用时被取回并解密。

收稿日期：2016-06-20；**修回日期**：2016-08-03
通信作者：李凤华，lfh@iie.ac.cn
基金项目：国家自然科学面上基金资助项目（No.61170251）；国家高技术研究发展计划（“863”计划）基金资助项目（No.2015AA016007）；国家自然科学基金—广东联合基金资助项目（No.U1401251）

Foundation Items: The National Natural Science Foundation of China General Project(No.61170251), The National High Technology Research and Development Program of China (863 Program)(No.2015AA016007), The National Natural Science Foundation of China-Guangdong Provincial People's Government of the Joint Natural Science Fund Projects (No.U1401251)

此过程中，应避免取回大量无效数据造成通信资源和下载时间的浪费。而基于云构建的复杂计算系统中，程序运算量大，单次运行周期长，应在程序执行前确认相关算法、策略或参数等关键内容的完整性，避免无效运行的情况发生。在诸如此类的云应用系统中，需要考虑外源数据的完整性问题。但云服务器并不完全可信，用户无法依赖云自身的安全防护手段保证数据的完整性，而且云服务器存在设备损坏、遭受自然灾害和黑客攻击的可能性。所以，用户需要自主地验证云数据的完整性。如果用户将云数据下载到本地以验证其完整性，在数据量较大的情况下是低效的。为此，研究者们提出了数据持有性证明（PDP, provable data possession）[3]和数据可恢复证明（POR, proofs of retrievability）[4]。PDP 方案使用户即使删除了本地数据，仍然可在无需下载数据的条件下，利用数据标签验证云服务器是否持有完整的用户数据，且在此过程中不会泄露数据内容。PDP 方案注重快速验证云端数据是否损坏，而 POR 方案除此功能之外还能恢复已损坏数据，二者对应的应用需求不同。另外，在云计算环境中，用户身份认证是防止数据非法访问的前提条件，也是计费、日志记录和审计的基础[5]。即使云存储系统中的敏感数据已被加密保护，仍有必要对访问者进行身份认证，以便降低密钥泄露对敏感数据的安全威胁，也能有效减少攻击者发动侧信道攻击的可能性。通过上述分析可以看出，云应用场景中存在同时需要数据持有性证明和身份认证的情况。

针对云数据持有性证明问题，研究者首先提出了公开验证的 PDP 方案[3,6~8]。在公开验证的 PDP 方案中，任何人都可以向云提出数据持有性验证请求并得到正确的应答结果。但某些具有较高安全需求的用户将云服务器的数据持有情况视为隐私，不希望这些信息被公开获得。而攻击者可以利用公开验证的 PDP 方案评估攻击效果，进而找出针对云中敏感数据的有效攻击方法。所以，如何限定验证者身份成为 PDP 方案需要解决的问题。为此，Shen 和 Zeng[9]提出了代理验证的 PDP 方案，该方案中数据拥有者为验证代理生成代理密钥，云服务器利用该代理密钥对数据拥有者生成的标签进行转换，使代理可以验证云中数据的完整性。Wang[10]给出了一个基于授权的 PDP 方案，该方案的基本思想是数据拥有者利用签名给验证者授权，拥有授权才能验证云服务器持有的数据是否完整。Ruan 和 Lei[11]提出了一种具有细粒度权限控制能力的代理验证方案，该方案中验证者可在不提供私钥的情况下向他人证明数据丢失。Xu 等[12]提出了一种可保护验证者隐私的代理验证方案。

针对云环境中的身份认证，Nimmy 和 Sethumadhavan[13]提出了一种利用秘密共享实现的双向认证方案。该方案中，服务器将用户的凭证分为 2 份，一份存储在智能卡中，另一份存储在服务器中。Hao 等[14]提出了一种基于时间有效期票据的云环境双向认证方案。该方案的优点是服务器向用户发出一定数量的数字票据，用户可以一次数据验证使用一张票据，所以它可以节省验证的时间。Huang 等[15]提出了一种具有顽健性和隐私保护的云认证方案。Nagaraju 和 Parthiban[16]提出了可证明安全的多因子云认证方案。

虽然数据持有性证明方案和云认证协议分别解决了云环境中数据持有性证明及用户身份认证问题，但在一个方案中既完成用户认证又实现数据持有性证明，可提供更好的安全性、运算效率和环境适应能力。所以，研究支持身份认证功能的数据持有证明具有理论和应用价值。2015 年，Hahn 等[17]首次提出了基于数据块持有证明的认证方案。但该方案仅实现了认证，不支持密钥协商，且用户需要存储由云服务器生成的、完整的 Merkle 树，要求用户具有较大的存储空间。

为此，本文提出一种支持身份认证的数据持有性证明方案，简称 A-PDP 方案。本方案利用预生成的数据标签和数据持有性证明过程的中间数据实现用户与云服务器之间的身份认证与密钥协商，且不泄露数据内容和用户身份。本文贡献如下：提出一个方案，用于既要验证云持有数据完整性，又要验证用户身份的应用场景，仅当用户拥有授权且云持有完整数据的条件下，才能完成相互认证和会话密钥协商；方案具有密钥确认功能，且交互轮次比组合方案减少一半；方案的用户端运算量小、存储内容少；在随机预言机模型下证明了方案的安全性。

2 A-PDP 模型

本节首先给出 A-PDP 方案定义，然后描述方案的安全目标和相应的攻击模型。

2.1 方案定义

A-PDP 方案架构如图 1 所示，方案中存在 3 种实体：云服务器 S、数据拥有者 H 和用户 U。

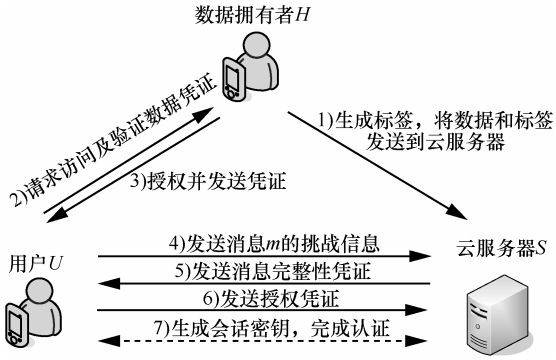


图 1　A-PDP 方案架构

数据拥有者 H 首先对将要上传到云服务器 S 的数据进行分组，为每组数据生成标签，然后将数据分组和标签存储到云服务器 S 中。用户 U 在访问及验证数据拥有者 H 上传到云中的数据前，首先要通过安全信道获得数据拥有者 H 的授权。用户 U 得到授权之后，可以生成数据持有性验证的挑战并发送给云服务器 S。云服务器 S 收到挑战后，利用数据分组和对应的标签，生成数据持有凭证并返回给用户 U。用户 U 根据持有凭证判断云服务器 S 是否持有完整的数据，然后生成授权凭证并发送给云服务器 S。云服务器 S 根据授权凭证判断用户拥有的授权是否有效。最后，双方根据需要协商出会话密钥。上述过程可简述为：用户 U 判断云服务器 S 中数据拥有者 H 的数据是否完整；而云服务器 S 判断用户 U 是否拥有授权。只有当云服务器 S 持有完整的数据且用户 U 拥有授权的情况下，二者才能协商出相同的会话密钥。在 A-PDP 方案中，假设云服务器 S 是半可信的，会诚实地执行方案，不会主动泄露私钥和秘密数据。

一个 A-PDP 方案包括 7 个多项式时间算法：Setup、TagGen、Authorize、Challenge、GenProof、CheckProof 和 CheckAuthority，算法具体定义如下。

Setup(1^k)：系统输入一个安全参数 k，为数据拥有者 H 和云服务器 S 生成公私钥对。其中，数据拥有者 H 的公私钥对为 (pk_H, sk_H)，云服务器 S 的公私钥对为 (pk_S, sk_S)。

TagGen$(pk_H, sk_H, m) \to Tag_m$：数据拥有者 H 根据自身的公私钥对 (pk_H, sk_H)，为数据分组 m 生成标签 Tag_m，然后将数据分组 m 和 Tag_m 一起发送给云服务器 S。

Authorize$(sk_H, ID_U) \to cert$：数据拥有者 H 根据自身的私钥 sk_H 和用户 U 的身份标识 ID_U，生成授权 $cert$ 并安全递送给用户。

Challenge$(pk_S, cert, Tag_m) \to Chal$：用户 U 根据云服务器 S 的公钥 pk_S、授权 $cert$ 和数据标签 Tag_m，生成挑战 $Chal$ 并发送给云服务器 S。

GenProof$(pk_H, sk_S, Tag_m, m, Chal) \to V_m$：云服务器 S 根据数据拥有者 H 的公钥 pk_H、自身的私钥 sk_S、数据标签 Tag_m、数据分组 m 以及用户 U 生成的挑战 $Chal$，生成持有数据 m 的凭证 V_m 并发送给用户 U。

CheckProof$(pk_H, pk_S, cert, Chal, V_m) \to \{$success, failure$, V_U\}$：用户 U 根据数据拥有者 H 的公钥 pk_H、云服务器 S 的公钥 pk_S、授权数据 $cert$、挑战 $Chal$ 和云服务器 S 生成的凭证 V_m，判断云服务器 S 是否持有完整的数据分组 m，然后生成拥有授权 $cert$ 的凭证 V_U，将凭证 V_U 发送给云服务器 S。

CheckAuthority$(pk_H, sk_S, V_U) \to \{$success,failure$\}$：云服务器 S 根据收到的授权凭证 V_U，验证用户 U 是否具有数据拥有者 H 的授权。

最后，云服务器 S 和用户 U 可利用上述步骤的过程数据协商出相同的会话密钥。

2.2　安全模型

A-PDP 方案具有数据持有性证明和认证密钥协商功能，应同时满足二者的安全目标。A-PDP 方案的安全目标为：1) 在无数据 m 的情况下，云服务器不能正确地给出持有数据 m 的凭证；2) 无授权用户无法判断出云服务器中挑战对应数据的完整性；3) 无授权用户或者其他授权用户无法正确地猜测出某授权用户与云服务器之间协商的会话密钥。其中，目标 1)和目标 2)分别为数据持有性证明的可验证性和授权验证性，目标 3)为认证密钥协商安全性。

以下对上述安全目标对应的攻击者进行说明。目标 1)的攻击者是云服务器，云服务器希望在篡改或删除用户数据的情况下，仍然可以向用户证明存储在服务器上的数据是完整的，该情况是所有 PDP 方案都需要解决的问题。目标 2)的攻击者是没有授权的用户，称为“外部攻击者”，该攻击者希望得到云服务器数据持有的相关信息，如某份数据是否存储在云服务器中。目标 3)的攻击者有 2 类，一类是外部攻击者，另一类是有授权的恶意用户，称为“内部攻击者”。目标 3)的 2 类攻击者都希望获得其他授权用户与云服务器之间的会话密钥，而内部攻击者拥有外部攻击者的全部能力并且拥有合法的授权数据。为便于说明，本文将外部攻击者作为目标 2)的攻击者，将内部攻击者作为目标 3)的

攻击者。

根据安全目标和相应的攻击者，构造 3 种攻击模型。其中，$\mathcal{C}$ 表示挑战者，$\mathcal{A}$ 表示攻击者。

可验证性攻击模型：在该模型中，攻击者是云服务器，拥有云服务器的私钥以及若干询问。攻击模型如下。

Setup：$\mathcal{C}$ 运行 Setup(1^k) 为云服务器 S 和数据拥有者 H 生成公私钥对，得到 (pk_S, sk_S) 和 (pk_H, sk_H)，公开公钥并将云服务器私钥 sk_S 发送给 $\mathcal{A}$。$\mathcal{C}$ 运行 Authorize(sk_H, ID_U)，得到授权 $cert$。$\mathcal{C}$ 运行标签产生算法 TagGen(pk_H, sk_H, m_i)，计算出若干数据分组 m_i 的标签 Tag_{m_i}，将数据和标签发送给 $\mathcal{A}$。

TagQuery：$\mathcal{A}$ 可以选择任意数据块 m_i，将 m_i 发送给 $\mathcal{C}$。$\mathcal{C}$ 运行标签产生算法 TagGen(pk_H, sk_H, m_i)，生成标签 Tag_{m_i} 并返回给 $\mathcal{A}$。

HashQuery：$\mathcal{A}$ 可以将任意数据发送给 $\mathcal{C}$，$\mathcal{C}$ 将对应的散列结果返回给 $\mathcal{A}$。

Challenge：$\mathcal{C}$ 选择一个 $\mathcal{A}$ 没有的数据 m'，生成相应的挑战信息 $Chal_{m'}$ 以及 m' 的标签 $Tag_{m'}$，将 $Chal_{m'}$ 和 $Tag_{m'}$ 发送给 $\mathcal{A}$。

Forge：$\mathcal{A}$ 根据 $Chal_{m'}$ 生成挑战应答 $Ans_{m'}$（此时 $\mathcal{A}$ 不知道 m_i'），将 $Ans_{m'}$ 发送给 $\mathcal{C}$。

Check：$\mathcal{C}$ 运行 CheckProof($pk_H, pk_S, cert$, $Chal_{m'}, Ans_{m'}$)，若运行结果为 success，则表明 $\mathcal{A}$ 正确地生成了对于 m' 挑战应答，破坏了方案的可验证性。

定义 1 如果任何多项式时间的攻击者按照上述模型攻破某 A-PDP 方案可验证性的概率是可忽略的，则该 A-PDP 方案具有可验证性。

授权验证性攻击模型：在该模型中，$\mathcal{A}$ 是外部攻击者，不能获得云服务器的私钥和用户的授权数据。攻击模型如下。

Setup：$\mathcal{C}$ 运行 Setup(1^k) 为云服务器 S 和数据拥有者 H 生成公私钥对，得到 (pk_S, sk_S) 和 $(pk_H$, $sk_H)$。$\mathcal{C}$ 运行 Authorize(sk_H, ID_U)，得到授权 $cert_U$。$\mathcal{C}$ 运行标签产生算法 TagGen(pk_H, sk_H, m_i)，计算出若干数据分组 m_i 的标签 Tag_{m_i}，将数据分组和标签发送给 $\mathcal{A}$。

TagQuery：$\mathcal{A}$ 可以选择任意数据分组 m_i，然后发送 m_i 给 $\mathcal{C}$。$\mathcal{C}$ 运行标签产生算法 TagGen(pk_H, sk_H, m_i)，计算出数据分组 m_i 的标签 Tag_{m_i}，然后将得出的标签发送给 $\mathcal{A}$。

HashQuery：$\mathcal{A}$ 可以将任意数据发送给 $\mathcal{C}$，$\mathcal{C}$ 将对应的散列结果返回给 $\mathcal{A}$。

SendQuery：$\mathcal{A}$ 冒充当前用户，发送任意消息给 $\mathcal{C}$。$\mathcal{C}$ 根据协议运行，对消息进行处理后将对应的应答返回给 $\mathcal{A}$。

Challenge：$\mathcal{C}$ 选择一个 $\mathcal{A}$ 没有询问过的数据分组 m_j，发送数据分组 m_j 的标签 Tag_{m_j} 给 $\mathcal{A}$。需要 $\mathcal{A}$ 判断标签 Tag_{m_j} 对应的数据分组是否存储在云服务器 S 中。

Judge：如果 $\mathcal{A}$ 可以正确地计算出 CheckProof ($pk_H, pk_S, cert, Tag_{m_j}, Chal, V_{m_j}$) 的结果，则说明 $\mathcal{A}$ 破坏了方案的授权验证性。

定义 2 如果任何多项式时间的攻击者按照上述模型攻破某 A-PDP 方案授权验证性的概率是可忽略的，则该 A-PDP 方案具有授权验证性。

认证密钥协商安全性攻击模型，在该模型中，$\mathcal{A}$ 是内部攻击者。攻击模型如下。

Setup：$\mathcal{C}$ 运行 Setup(1^k) 为云服务器 S 和数据拥有者 H 生成公私钥对，得到 (pk_S, sk_S) 和 (pk_H, sk_H)。$\mathcal{C}$ 运行 Authorize($sk_H, ID_{U'}$) 为所有合法用户生成授权，$\mathcal{A}$ 可得到除当前用户 U 之外任何其他用户 U' 的授权 $cert_{U'}$。$\mathcal{C}$ 运行标签产生算法 TagGen(pk_H, sk_H, m_i)，计算出若干数据分组 m_i 的标签 Tag_{m_i}，将数据分组和标签发送给 $\mathcal{A}$。

TagQuery：$\mathcal{A}$ 可以选择任意数据分组 m_i，然后发送 m_i 给 $\mathcal{C}$。$\mathcal{C}$ 运行标签产生算法 TagGen(pk_H, sk_H, m_i)，计算出数据分组 m_i 的标签 Tag_{m_i}，然后将得出的标签发送给 $\mathcal{A}$。

HashQuery：$\mathcal{A}$ 可以将任意数据发送给 $\mathcal{C}$，$\mathcal{C}$ 将对应的散列结果返回给 $\mathcal{A}$。

SendQuery：$\mathcal{A}$ 冒充当前用户，发送任意消息给 $\mathcal{C}$。$\mathcal{C}$ 根据协议运行，对消息进行处理后将对应的应答返回给 $\mathcal{A}$。

TestQuery：$\mathcal{A}$ 选择一个之前没有进行过任何询问的会话作为测试会话，对该会话进行 Test 询问。$\mathcal{C}$ 随机选择一个比特 b。若 b=1，则 $\mathcal{C}$ 发送正确的会话密钥给 $\mathcal{A}$；反之，b=0，$\mathcal{C}$ 选择一个随机数作为会话密钥返回给 $\mathcal{A}$。

Judge：如果 $\mathcal{A}$ 可以以不可忽略的优势猜测出 TestQuery 中 b 的值，即猜中 TestQuery 的输出是随机数还是真实的会话密钥，则说明攻击者破坏了方

案的认证密钥协商安全性。

定义 3　如果任何多项式时间的攻击者按照上述模型攻破某 A-PDP 方案认证密钥协商安全性的概率是可忽略的，则该 A-PDP 方案具有认证密钥协商安全性。

3　方案构造

本节给出 A-PDP 方案具体构造。A-PDP 方案利用指数运算的同态性，可同时对多个数据分组进行批量验证，即可实现数据分组的随机线性组合(random linear combination)验证。

A-PDP 方案具体构造如图 2 所示。

Setup(1^k)：系统选择 2 个安全的大素数 p 和 q。设 $N=pq$ 为 RSA 算法中的模，QR_N 是模 N 的二次剩余所组成的群，g 是 QR_N 的生成元。系统选择散列函数 $H(\cdot):(0,1)^* \to (0,1)^k$、一对安全的对称加解密算法和密钥派生函数 $KDF(i,len)$。其中，k 为安全参数，K 是对称密钥，i 是派生因子，len 是派生密钥的长度。N、g、$H(\cdot)$、$E_K(\cdot)$、$D_K(\cdot)$ 和 $KDF(\cdot,\cdot)$ 是系统公开参数。系统为数据拥有者 H 和云服务器 S 生成公私钥对，将公钥公开，将私钥安全递送给相应实体。其中，数据拥有者 H 的公私钥对为 $pk_H=g^{s_H}, sk_H=s_H$；云服务器 S 的公私钥对为 $pk_S=g^{s_S}, sk_S=s_S$。系统选择大素数 d、e，满足 $de\equiv 1 \bmod (p-1)(q-1)$，将$(d,e)$作为数据拥有者 H 的标签签名公私钥对。

TagGen($pk_H, d, pk_S, m_i(1\leqslant i\leqslant n)$)：数据拥有者 H 将数据 m（可以是加密生成的数据）分成 n 组，每个分组标记为 $m_i(1\leqslant i\leqslant n)$。数据拥有者 H 为每个分组 m_i 生成标签 $Tag_{m_i}=\left(g^{s_S\left(H\left(w_{m_i}\right)+m_i\right)}\right)^d$，然后将数据分组和标签一同发给云服务器 S。其中，w_{m_i} 是每个分组对应的特征值，云服务器 S 利用特征值查找数据分组，特征值不泄露数据内容，可包含文件名、按照一定规则定义的序号、关键词等。

Authorize(sk_H, ID_U)：用户 U 首先通过安全信道发送自己的身份标识 ID_U 给数据拥有者 H。数据拥有者 H 随机选择 $r\in Z_N^*$ 并计算 $R_U=g^r$ 和 $s_U=r+h_U s_H$。其中，$h_U=H(ID_U \| R_U)$。数据拥有者 H 通过安全信道将授权 $cert=(R_U, s_U)$ 发送给用户 U。用户 U 在收到授权 $cert$ 后，验证等式 $g^{s_U}=R_U pk_H^{H(ID_U\|R_U)}$ 是否成立。若等式成立则说明授权有效；否则说明授权无效。

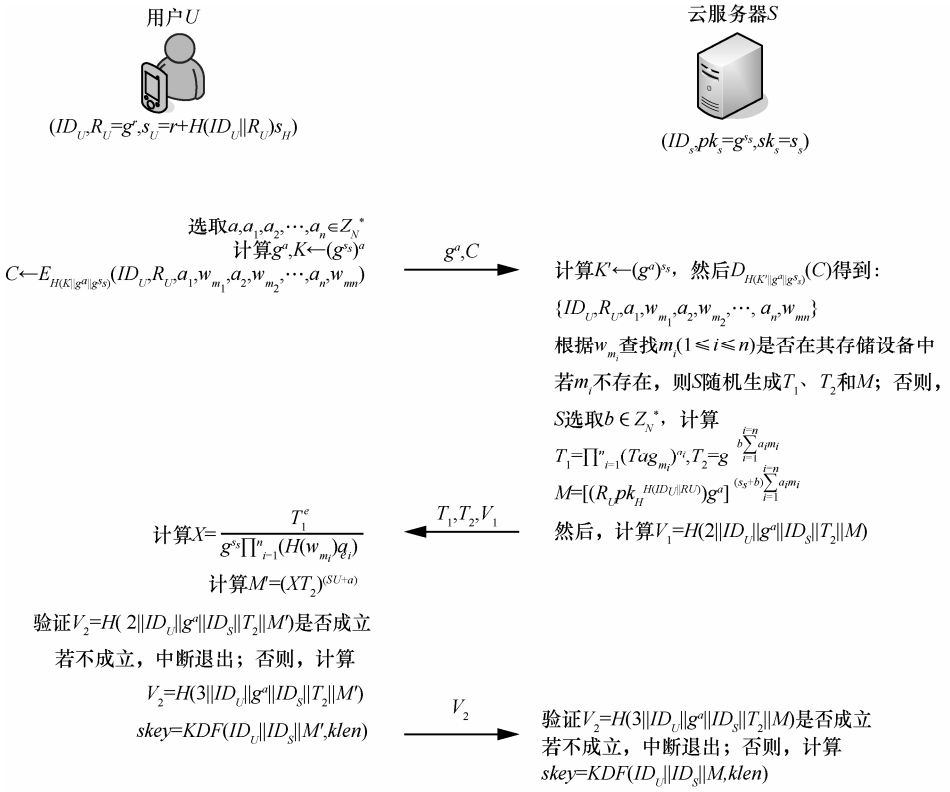


图 2　A-PDP 方案构造

Challenge($pk_S, cert, w_{m_i}(1 \leqslant i \leqslant n)$)：用户 U 在收到数据拥有者 H 的授权 (R_U, s_U) 后，任何时候都可以对云服务器 S 上数据拥有者 H 的数据进行持有性验证。假设用户 U 要验证数据 m，数据分组 m_i 对应的标签为 Tag_{m_i}，数据分组的特征值为 w_{m_i}，其中，$1 \leqslant i \leqslant n$。用户 U 随机选择 $a \in Z_N^*$ 并计算 g^a、$K=(g^{s_S})^a$ 和加密密钥 $H(K \| g^a \| g^{s_S})$，选取 n 个随机数 $a_1, a_2, \cdots, a_n \in Z_N^*$，将挑战消息 $Chal$={ g^a,C}发送给云服务器 S。其中，$C=E_{H(K\|g^a\|g^{s_S})}(ID_U, R_U, g^a, a_1, w_{m_1}, a_2, w_{m_2}, \cdots, a_n, w_{m_n})$。

GenProof($pk_H, sk_S, Tag_m, m, Chal$)：云服务器 S 在收到挑战消息 $Chal=\{g^a, C\}$ 后，执行以下操作。首先，根据自己的私钥 s_S 计算 $K'=(g^a)^{s_S}$ 以及解密密钥 $H(K' \| g^a \| g^{s_S})$，并利用 $D(\cdot)$ 解密 C，得到 $ID_U, R_U, g^a, a_1, w_{m_1}, a_2, w_{m_2}, \cdots, a_n, w_{m_n}$。然后，根据得到的数据特征值查找出所有对应的数据分组和标签。如果所有数据分组都被找到，则随机选取 $b \in Z_N^*$ 计算：g^b、$T_1=\prod_{i=1}^{n}(Tag_{m_i})^{a_i}$、$T_2=g^{b\sum_{i=1}^{n}a_i m_i}$ 和 $M=[(R_U pk_H^{H(ID_U\|R_U)})g^a]^{(s_S+b)\sum_{i=1}^{n}a_i m_i}$；否则，随机生成 T_1、T_2 和 M。最后，根据 T_2 和 M 计算 $V_1=H(2 \| ID_U \| g^a \| ID_S \| T_2 \| M)$，将 $\{T_1, T_2, V_1\}$ 作为挑战响应发送给用户 U。

CheckProof()：用户 U 在收到挑战响应后，计算 $X=\dfrac{T_1^e}{g^{s_S\prod_{i=1}^{n}(H(w_{m_i})\cdot a_i)}}$ 并计算 $M'=(XT_2)^{s_U+a}$。判断等式 $V_1=H(2 \| ID_U \| g^a \| ID_S \| T_2 \| M')$ 是否成立。如果等式不成立，则说明云服务器 S 存储的数据不完整，退出当前会话；否则，计算 $V_2=H(3 \| ID_U \| g^a \| ID_S \| T_2 \| M')$ 以及 $skey=KDF\ (ID_U \| ID_S \| M', klen)$。将 V_2 发送给云服务器 S。

CheckAuthority()：云服务器 S 收到 V_2 后，验证等式是否成立。若等式不成立，退出当前会话；否则，说明用户 U 拥有授权并计算出了相同的会话密钥。最后，计算 $skey=KDF(ID_U \| ID_S \| M, klen)$。

4 正确性分析

本方案中，数据拥有者对所有数据分组的标签进行了签名，使包括云服务器在内的其他实体无法伪造标签，用户在验证消息前无需存储或获取相应的标签。云服务器将标签聚合后生成 T_1 并发送给用户，用户无需得到每个数据分组实际的标签数据，仅需验证标签的聚合结果，并从中获得关于数据分组的信息。本方案的关键点是用户 U 计算出的 M' 与云服务器 S 计算出的 M 是否相等。下面将数据分组标签代入到方案中，给出方案的正确性分析。

由于 $Tag_{m_i}=(g^{s_S(H(w_{m_i})+m_i)})^d$，有

$$\begin{aligned} X &= \frac{T_1^e}{g^{s_S\prod_{i=1}^{n}(H(w_{m_i})\cdot a_i)}} \\ &= \frac{(\prod_{i=1}^{n}(Tag_{m_i})^{a_i})^e}{g^{s_S\prod_{i=1}^{n}(H(w_{m_i})a_i)}} \\ &= \frac{\prod_{i=1}^{n}(g^{s_S(H(w_{m_i})+m_i)})^{da_i e}}{g^{s_S\sum_{i=1}^{n}(H(w_{m_i})a_i)}} \\ &= g^{\sum_{i=1}^{n}s_S(m_i a_i)} \end{aligned}$$

由于 $T_2=g^{b\sum_{i=1}^{i=n}a_i m_i}$，所以有

$$\begin{aligned} M' &= (XT_2)^{(s_U+a)} \\ &= \left(g^{\sum_{i=1}^{n}s_S(m_i a_i)} g^{b\sum_{i=1}^{n}a_i m_i}\right)^{(s_U+a)} \\ &= g^{(s_U+a)(s_S+b)\sum_{i=1}^{n}a_i m_i} \end{aligned}$$

由于 $R_U=g^r$，$s_U=r+h_U s_H=r+H(ID_U \| s_H)\cdot s_H$，所以有

$$\begin{aligned} M &= [(R_U pk_H^{H(ID_U\|R_U)})g^a]^{(s_S+b)\sum_{i=1}^{n}a_i m_i} \\ &= [(g^r g^{s_H H(ID_U\|R_U)})g^a]^{(s_S+b)\sum_{i=1}^{n}a_i m_i} \\ &= (g^{r+s_H H(ID_U\|R_U)+a})^{(s_S+b)\sum_{i=1}^{n}a_i m_i} \\ &= g^{(s_U+a)(s_S+b)\sum_{i=1}^{n}a_i m_i} \end{aligned}$$

所以，当云服务器拥有挑战对应的所有数据分组的情况下，若云服务器按照方案设计应答用户挑战请求，则用户和云服务器可以计算出相同的 M 值。否则，将导致 $V_1=H(2 \| ID_U \| g^a \| ID_S \| T_2 \| M')$ 验证失败。

5 安全性分析

根据 2.2 节中的攻击模型，在随机预言机模型中证明本方案的安全性。

定理 1 若 CDH 假设在 QR_N 中成立，则 A-PDP 方案在随机预言模型下具有可验证性。

证明 根据可验证性攻击模型，$\mathcal{C}$ 在随机预言模型下为 $\mathcal{A}$ 模拟 A-PDP 方案，并回答 $\mathcal{A}$ 提出的所有数据分组 m_i 的标签询问以及所有数据的散列询

问。若 $\mathcal{A}$ 可以在多项式时间内以不可忽略的概率攻破 A-PDP 方案的可验证性，则挑战者 $\mathcal{C}$ 可以利用 $\mathcal{A}$ 破解 QR_N 上 CDH 问题。具体证明过程如下。

Setup：$\mathcal{C}$ 运行 Setup(1^k) 算法为云服务器 S 生成公私钥对，为数据拥有者 H 生成公私钥对和标签签名公私钥对。其中，云服务器 S 的公私钥对 $pk_S = g^{s_S}, sk_S = s_S$，数据拥有者 H 的公私钥对 $pk_H = g^{s_H}, sk_H = s_H$，标签签名公私钥对为($d$,$e$)。根据 A-PDP 方案生成用户授权 (R_U, r_U)。这里将数据拥有者 H 的公钥 pk_H、云服务器 S 的公钥 pk_S 和私钥 sk_S 发送给 $\mathcal{A}$。

TagQuery：$\mathcal{A}$ 可以选择若干分组 $m_i (1 \leqslant i \leqslant n)$，并发送 m_i 给 $\mathcal{C}$，$\mathcal{C}$ 按照如下方式回答 $\mathcal{A}$ 的标签询问。

若已对数据 $m_i (1 \leqslant i \leqslant n)$ 进行过 TagGen 询问，则 $\mathcal{C}$ 从标签列表里找到对应的 Tag_{m_i}，将 Tag_{m_i} 发送给 $\mathcal{A}$；若没有对数据 m_i 进行过 TagGen 询问，则 $\mathcal{C}$ 运行标签产生算法，计算出数据 m_i 的标签 $Tag_{m_i} = (H(w_i)g^{m_i})^d$，然后将 Tag_{m_i} 发送给 $\mathcal{A}$，同时存储 (m_i, Tag_{m_i}) 到标签列表中。

HashQuery：$\mathcal{A}$ 可以对随机预言机进行询问，从而得到某个消息的散列值。$\mathcal{C}$ 按照如下方式回答攻击者对消息 X 的散列询问。

若消息 X 已经被进行过散列询问，则 $\mathcal{C}$ 从散列列表里找到对应的散列值 h_x，将 h_x 发送给 $\mathcal{A}$；若消息 X 没有被进行过散列询问，则 $\mathcal{C}$ 随机选取一个数值 h_x，将 h_x 作为消息 X 的散列值发送给 $\mathcal{A}$，同时存储 $h_x = H(X)$ 到标签列表中。

Challenge：$\mathcal{C}$ 任选一个验证实例，在该实例中任选一个随机数 g^x 作为协议中的 g^a，随机选取 $a_1, a_2, \cdots, a_n \in Z_N^*$，并计算 $K = (g^x)^{s_S}$ 以及 $H(K \| g^x \| g^{s_S})$。然后 $\mathcal{C}$ 再任选一个随机数 g^y，满足 y 是 $\{m_1, \cdots, m_n\}$ 中一个数据分组，标签为 $Tag_y = (H(w_y)g^y)^d$，特征值为 w_y。$\mathcal{C}$ 计算 $C = E_{H(K \| g^x \| g^{s_S})}(ID_U, R_U, g^x, a_1, w_{m_1}, \cdots, a_y, w_y, \cdots, a_n, w_{m_n})$。其中，有一个 m_i 即为选定的 y，其对应的随机数为 a_y。$\mathcal{C}$ 将挑战信息 $Chal = (g^x, C)$ 发送给 $\mathcal{A}$。这里的挑战模拟是合理的，因为 $\mathcal{A}$ 要在不知道数据 y 的情况下，成功地证明他拥有 y。

Forge：$\mathcal{A}$ 利用私钥 s_S，解密 C 得到 $(ID_U, R_U, g^x, a_1, w_{m_1}, \cdots, a_y, w_y, \cdots, a_n, w_{m_n})$。$\mathcal{A}$ 的目的是产生包含数据分组 y 的挑战 $Chal$ 的应答信息 V_y。若 $\mathcal{A}$ 可以成功地生成挑战信息 $Chal = (g^x, C)$ 的应答信息 V_y，使 CheckProof($pk_H, pk_S, cert, Chal, V_y$) 的结果为 success，则 $\mathcal{C}$ 可以 $\mathcal{A}$ 破解 CDH 问题。因为，若 $\mathcal{A}$ 要使 CheckProof 成功，即要得出 V_y 使其满足 $V_y = H(2 \| ID_U \| g^a \| ID_S \| T_2 \| M)$。也就是说，$\mathcal{A}$ 要得出这样的 V_y，必然进行了 $H(2 \| ID_U \| g^a \| ID_S \| T_2 \| M)$ 的散列询问。其中，

$$M = (XT_2)^{(s_U+x)} = \left(g^{s_S\left(\sum_{i=1,i\neq j}^{n} m_i a_i + y a_y\right)} g^{b\left(\sum_{i=1,i\neq j}^{n} a_i m_i + + a_y y\right)} \right)^{s_U+y}$$

$$= g^{(s_U+x)(s_S+b)\sum_{i=1,i\neq j}^{n} a_i m_i} g^{(s_U+x)(s_S+b)\cdot a_y y}$$

，b 为 $\mathcal{A}$ 选择的随机数。因此，$\mathcal{C}$ 可以查找散列列表获得 $g^{(s_U+x)(s_S+b)\sum_{i=1,i\neq j}^{n} a_i m_i} g^{(s_U+x)(s_S+b)a_y y}$。由于 $\mathcal{C}$ 知道 s_U、s_S、所有 a_i 以及除了消息 y 以外的所有 m_i，所以 $\mathcal{C}$ 可以获得 $g^{(s_U+x)(s_S+b)\sum_{i=1,i\neq j}^{n} a_i m_i}$，也就获得了 $g^{(s_U+x)(s_S+b)a_y y}$。由于 $g^{(s_U+x)(s_S+b)a_y y} = (g^{x a_y y})^{s_S+b} (g^{s_U a_y y})^{s_S+b}$，所以 $\mathcal{C}$ 可以获得 g^{xy}，也就破解了 QR_N 上 CDH 问题。由于 QR_N 上 CDH 假设在多项式时间内是成立的，所以定理 1 得证。

定理 2　若 CDH 假设在 QR_N 中成立，则 A-PDP 方案在随机预言模型下可以提供用户的授权验证性。

证明　定理 2 说明的是外部攻击者没有授权，不能对任何数据分组进行持有性验证。该外部攻击者即没有授权数据，也不知道云服务器和数据拥有者的密钥。授权验证性的目标是：外部攻击者知道数据分组 m_i 及其标签的情况下，仍然不能验证 m_i 是否存储在云服务器当中。具体证明如下。

Setup(1^k)：$\mathcal{C}$ 运行 Setup(1^k) 算法产生数据拥有者 H 的公私钥对为 $pk_H = g^{s_H}$、$sk_H = g^{s_H}$ 以及用于标签签名的公私钥对(d,e)。在产生云服务器 S 的公钥时，$\mathcal{C}$ 选择一个随机数 g^y 作为其公钥。$\mathcal{C}$ 按照如下方式为用户 U 发放授权。$\mathcal{C}$ 选择 $r \in Z_N^*$ 并计算 $R_U = g^r$，然后 $\mathcal{C}$ 随机选择 g^x，令 $g^x = h_U = H(ID_U \| R_U)$。$\mathcal{C}$ 可以这样做的原因是 $\mathcal{C}$ 控制着所有散列函数的返回值，最后 $\mathcal{C}$ 计算 $s_U = r + h_U s_H$ 并将 (R_U, s_U) 发送给用户 U。外部攻击者可以获得的是数据拥有者和云服务器的公钥、用户 U 相关信息 (ID_U, R_U)、数据分组 m_i 及相应标签。

HashQuery、TagQuery：$\mathcal{A}$ 可以进行标签询

问以及散列询问，$\mathcal{C}$ 对这些询问的应答方式与可验证性证明相同。

SendQuery：$\mathcal{A}$ 根据标签，生成针对若干数据分组 $m_i(1\leqslant i\leqslant n)$ 的挑战信息 $Chal=(g^z,C)$ 并发送给 $\mathcal{C}$。$\mathcal{C}$ 收到挑战信息后，运行 GenProof(pk_H, sk_S,Tag_{m_i},m_i,$Chal$) 算法。这里需要注意的是，由于 $\mathcal{C}$ 不知道云服务器公钥 g^y 中的 y，因此，无法直接计算出 $K=(g^a)^y$ 以及 $H(K\|g^a\|g^y)$，但是 $\mathcal{C}$ 可以查看随机预言机 H。若 $\langle h,K\|g^a\|g^y\rangle$ 在散列列表中，则意味着已经有实体对散列函数进行了 $H(K\|g^a\|g^y)$ 询问，那么 $\mathcal{C}$ 可以得到 $h=H(K\|g^a\|g^y)$。若 $\langle h,K\|g^a\|g^y\rangle$ 不在散列列表中，则 $\mathcal{C}$ 选取一个随机数 h 作为该散列询问的返回值，同时，将该消息对存储在散列列表中。所以，虽然 $\mathcal{C}$ 不知道 y，但可以利用随机预言机为 $\mathcal{A}$ 正确的模拟本文方案。当 $\mathcal{C}$ 得到 $h=H(K\|g^a\|g^y)$ 后，解密 C 得到数据分组 m_i 的特征值。若所有数据分组 $m_i(1\leqslant i\leqslant n)$ 都存在，$\mathcal{C}$ 选择一个随机数 $b\in Z_p^*$ 并计算 $T_1=\prod_{i=1}^{n}(Tag_{m_i})^{a_i}$，$T_2=g^{b\sum_{i=1}^{n}a_im_i}$，$M=[(R_U pk_H^{H(ID_U\|R_U)})g^a]^{(y+b)\sum_{i=1}^{n}a_im_i}$ 以及 $V_1=H(2\|H(ID_U\|g^a\|ID_S\|T_2\|M))$。虽然 $\mathcal{C}$ 不知道 y，但知道 r、a、b、s_H、g^y 以及所有的 a_i 和 m_i，所以仍然可以计算出 M，生成多个数据分组 $m_i(1\leqslant i\leqslant n)$ 的凭证 V_1，并将 (T_1,T_2,V_1) 发送给用户 U。若有一个分组 m_i 不存在，则 $\mathcal{C}$ 随机生成 (T_1,T_2,V_1) 作为挑战相应返回给 $\mathcal{A}$。

Challenge：$\mathcal{C}$ 随机生成一个数据分组 m_i，生成相应的标签 $Tag_{m_i}=(H(w_i)g^{m_i})^d$、特征值 w_{m_i}、随机数值 a_i，将 m_i、Tag_{m_i}、w_{m_i}、a_i 发送给 $\mathcal{A}$。

Judge：$\mathcal{A}$ 要判断云服务器 S 是否持有某数据分组 m_i。$\mathcal{A}$ 首先冒充用户者 U 向 $\mathcal{C}$ 发起挑战。$\mathcal{A}$ 选择 g^z，并计算 $K=(g^y)^z$，$H(K\|g^z\|g^y)$ 以及 $C=E_{H(K\|g^z\|g^{sS})}(ID_U,R_U,g^z,a_i,w_{m_i})$ 的值，然后将挑战信息 $Chal=(g^z,C)$ 发送给 $\mathcal{C}$。$\mathcal{C}$ 收到该消息后，执行 SendQuery 的方式生成 (T_1,T_2,V_1) 并返回给 $\mathcal{A}$。$\mathcal{A}$ 根据 (T_1,T_2,V_1) 判断数据分组 m_i 是否存储在云服务器 S 中。若 $\mathcal{A}$ 输出的结果与用户 U 运行算法 CheckProof(pk_H,pk_S,$cert$,w_{m_i},$Chal$,V) 得到的结果相同，则说明 $\mathcal{A}$ 成功计算出了 V_1，即获得了 $V_1=H(2\|H(ID_U\|g^z\|ID_S\|T_2\|M))$。那么 $\mathcal{C}$ 可以查看散列列表找到对应的散列询问，从而得到 $M=[(R_U pk_H^{H(ID_U\|R_U)})g^z]^{(y+b)a_im_i}$。在 QR_N 上 CDH 假设下，$\mathcal{A}$ 不可能正确地计算出 V_1。因为将 M 变形后可得 $M=(g^{s_U}g^z)^{(y+b)a_im_i}=g^{s_U(y+b)a_im_i}g^{z(y+b)a_im_i}$。$\mathcal{A}$ 知道 a_i、z 和 m_i，则可以计算出 $g^{z(y+b)a_im_i}$。但是 $\mathcal{A}$ 却无法计算出 $g^{s_U(y+b)a_im_i}$ 中的 $g^{s_Uya_im_i}$。若 $\mathcal{A}$ 能计算出 $g^{s_Uya_im_i}$，由于 $\mathcal{C}$ 知道 r、s_H、a_i 和 m_i，根据 $g^{y(r+xs_H)a_im_i}=(g^{yr}g^{xys_H})^{a_im_i}$，$\mathcal{C}$ 可以得到 g^{xy}，计算出了 $CDH(g^x,g^y)$，即破解了 QR_N 上 CDH 问题。由于 QR_N 上 CDH 假设在多项式时间内是成立的，所以定理 2 得证。

定理 3 若 CDH 假设在 QR_N 中成立，A-PDP 方案在随机预言模型下可以提供认证密钥协商安全性。

证明 根据 2.2 节中给出的认证密钥协商安全性攻击模型，只需证明内部攻击者 $\mathcal{A}$ 不能获得其他授权用户与云服务器之间的会话密钥即可，证明过程与授权验证性证明类似，这里只给出简要的分析。授权验证性的攻击者是没有合法授权的外部攻击者，而认证密钥协商安全性的攻击者可以拥有除了目标用户 U 以外所有授权用户的授权信息，但他仍然不能获得 U 和云服务器 S 之间的会话密钥。根据会话密钥公式 $skey=KDF(ID_U\|ID_S\|M,klen)$，$\mathcal{A}$ 可以获得 ID_U、ID_S 以及 $klen$，但不能获得 M。参考授权验证性的证明过程，若 $\mathcal{A}$ 能正确获取 M，则 $\mathcal{C}$ 可以利用攻击者破解 QR_N 上 CDH 问题。所以，$\mathcal{A}$ 拥有 $cert_U$ 不能对获得目标用户 U 的会话密钥产生任何优势。定理 3 得证。

6 性能分析

6.1 理论分析

本方案在同一个会话中既实现了数据持有性证明又实现了用户身份认证与密钥协商。为进行效率比较，除 Hahn 等方案[17]外，分别挑选 2 组经典、高效的 AKA 方案和 PDP 方案。其中，AKA 方案选择 Cremers 和 Feltz 提出的 SIG (NAXOS)方案[18]和 Krawczyk 提出的 HMQV 方案[19]，而 PDP 方案选择 Ateniese 等提出的 S-PDP 方案[3]和 Wang 提出的 PPDP 方案[10]。对各方案的运算量、存储消耗以及通信消耗等方面进行了分析，分析结果如表 1 所示。表 1 中，P 表示预处理，U 表示用户或验证者，S

表 1　　**方案性能对比**

方案	类型	运算量			存储损耗		通信损耗		交互次数
		P	*U*	*S*	*U*	*S*	*U*	*S*	
SIG(NAXOS)[18]方案	AKA	2*e*	3*e*+1*sig*+3*h*+1*ver*	3*e*+1*sig*+3*h*+1*ver*	*O*(1)	*O*(1)	*O*(1)	*O*(1)	2
HMQV[19]方案	AKA	2*e*	2*e*+2*h*	2*e*+2*h*	*O*(1)	*O*(1)	*O*(1)	*O*(1)	2
S-PDP[3]方案	PDP	2*e*+1*h*	(*n*+3)*e*+(*n*+1)*h*+(2*n*+3)*f*	(*n*+1)*e*+(*n*+1)*h*+2*nf*	*O*(1)	*O*(*n*)	*O*(1)	*O*(1)	2
PPDP[10]方案	PDP	2*e*+1*h*	2*p*+(*n*+1)*e*+2*nf*	*ne*+2*nf*	*O*(1)	*O*(*n*)	*O*(1)	*O*(1)	2
Hahn[17]方案	PDP	1*e*+(2*n*-1)*h*	(*n*+1)*e*+(3*n*+2)*h*+(*n*+2)*f*	(3*n*+2)*e*+(3*n*+3)*h*+(*n*+1)*f*	*O*(*n*)	*O*(*n*)	*O*(log *n*)	*O*(1)	3
本文方案	PDP+AKA	1*e*+1*h*	5*e*+1*E*+3*h*+(*n*+1)*f*	(*n*+4)*e*+4*h*+1*E*+1*f*	*O*(1)	*O*(*n*)	*O*(*n*)	*O*(1)	3

表示服务器端。表 1 内容中，*n* 表示数据分组的数量，*p* 表示一次双线性对运算，*e* 表示一次指数运算，*h* 表示散列运算，*E* 表示对称加解密运算，*sig* 表示签名运算，*ver* 表示签名验证运算，*f* 表示伪随机函数运算。

运算量方面。Hahn 等方案[17]利用数据持有证明完成认证，不支持密钥协商。若不考虑本文方案，同时完成数据持有性证明和认证密钥协商需要 PDP 方案和 AKA 协议组合。本文以效率较高的 HMQV 方案和 S-PDP 方案组合为例。该组合方案的运算量将是 HMQV 方案和 S-PDP 方案运算量的叠加。组合方案的用户端运算量为 2*e*+2*h*+(*n*+3)*e*+(*n*+1)*h*+(2*n*+3)*f*=(*n*+5)*e*+(*n*+3)*h*+(2*n*+3)*f*，Hahn 等方案[17]用户端运算量为(*n*+1)*e*+ (3*n*+2)*h*+(*n*+2)*f*，而本文方案的用户端运算量仅为 5*e*+1*E*+(*n*+1)*f*+3*h*。本文方案用户端中运算较大的指数运算 *e* 的数量不随数据分组数量 *n* 变化。这意味着验证的数据分组数量越大，本文方案用户端的性能优势越明显。其原因在于，本文方案的标签生成算法和方案设计降低了用户验证云服务器数据持有凭证的运算量，而认证过程复用了数据持有性证明运算的中间结果，使方案在无需额外运算量和交互步骤的情况下，实现了功能目标。

存储消耗方面。本文方案与组合方案相比，用户端和服务器端的存储消耗相同。本文方案中，用户端仅存储授权信息，而服务器端存储数据分组及标签。需要说明的是，本文方案中数据拥有者对所有标签进行了签名，使其他实体或攻击者无法伪造标签，用户不需要获得每个数据分组的标签，仅需验证标签聚合结果是否正确。而 Hahn 等方案[17]中，用户需要存储完整的 Merkle 树，不适合数据分组数量较大的情况。

通信消耗方面。本文方案效率不如其他方案，原因是用户需要传输为每个数据分组选取的随机数和用于查找数据分组的特征值，而其他对比 PDP 方案均指定了数据分组集合，没有数据分组查找过程，不符合实际应用情况。若本文方案利用伪随机函数选择待验证的数据分组并生成所需的随机数，通信损耗可以降至 *O*(1)，但用户必须存储所有数据标签，以防止云服务器利用无关数据生成持有证明。本文方案以通信损耗换取存储空间和计算性能，并减少了系统参数中的伪随机函数，降低了系统复杂度，使本文方案具有较强的用户端设备适应性，适应运算能力和存储能力相对较弱的移动设备或云终端。

交互次数方面。本文方案用户和服务器之间的有 3 次交互，而 2 个经典 AKA 方案仅需 2 次交互。但这 2 个 AKA 方案中均未提供会话密钥确认功能，也就是说不能确定对方是否能正确的计算出会话密钥。通常情况下，若实现会话密钥确认，AKA 方案至少需要额外增加 2 次交互，一次交互用于用户确认服务器的会话密钥，另一次交互用于服务器确认用户的会话密钥。所以，若组合方案增加会话密钥确认功能，则共需 6 次交互，而本文方案需要 3 次交互，次数减少一半，其原因是方案利用数据持有性证明的确认步骤既作为对云服务器的认证，又作为会话密钥的确认，这是普通 AKA 方案无法实现的。Hahn 等方案[17]交互次数与本文方案相同，但该方案不支持密钥协商且认证的发起者是云服务器。在云环境中，用户在线情况变化较快，不应由云服务器发起认证请求，至少应先由用户发送含身份信息的消息给云服务器。所以，Hahn 等方案[17]合理的交互数应为 4 次。

6.2　实验仿真

下面先对实验环境和实验过程中选择的算法及参数进行说明，然后给出实验结果。

基于 MIRACL 库[20]实现了本文方案和组合方案，并通过测试程序得到了 2 种方案在不同数据分

组数量情况下方案各步骤的运行时间。实验环境由宿主机和测试平台 2 个部分组成，二者通过网络连接。宿主机是一台普通 PC，用于交叉编译方案程序和测试程序，运行 64 位 Linux 系统，发行版本为 Ubuntu 12.04.5。测试平台是一块嵌入式开发板，用于运行方案程序和测试程序。测试平台的主处理器为飞思卡尔公司的 i.mx6 系列四核 Cortex-A9 ARM 处理器，主频为 1.2 GHz，拥有 512 MB DDR2 内存，运行 32 位 Linux 3.1 系统，测试平台的运算性能如表 2 所示。

表 2　　测试平台运算性能

基础运算	平均消耗时间
1 024 位 DH(17 位幂)	0.82 ms
1 024 位 DH(1 024 位幂)	11.69 ms
1 024 位 DH(160 位幂)	1.36 ms
AES 加/解密	2.07 μs/16 byte
SHA1	4.35 μs /64 byte
伪随机数生成算法	0.33 μs /20 byte

本文方案和组合方案选用相同的算法和安全参数，具体如下。安全参数 k 为 1 024 bit，选用 1 024 位 RSA 算法，选用 1 024 位 DH 算法（160 位幂），选用 SHA1 作为摘要算法，选用 AES 作为对称加解密算法，选用 MIRACL 库自带的随机数生成算法。RSA 算法的安全大素数 p 和 q 均为 512 bit，N 为 1 024 bit，d 为 1 024 bit，e 为常数 65 537。模 N 二次剩余群 QR_N 的生成元 g 为 1 024 bit。

本文方案的性能实验结果如图 3 所示。其中，预处理是指数据有拥有者生成数据分组的标签。从图 3 可以看出，随着数据分组数量的增加，用户端运算时间的变化很小，服务器端运算时间有所增加，而预处理运算时间显著增加。实验环境中，当分组数量为 10 万时，用户端完成全部运算的时间为 4 023 ms。图 4 给出了本文方案和组合方案用户端的运算时间对比。得到图 4 实验结果的原因是 RSA 运算和 DH 运算中的指数运算的时间消耗较大，相应步骤处理一个数据分组需要的指数运算越多则运算时间变化越明显。而根据表 2 统计的运算量，本文方案用户端所需的指运算数量固定不变，而预处理和服务器端所需的 RSA 和 DH 运算量均与数据分组数量成比例。

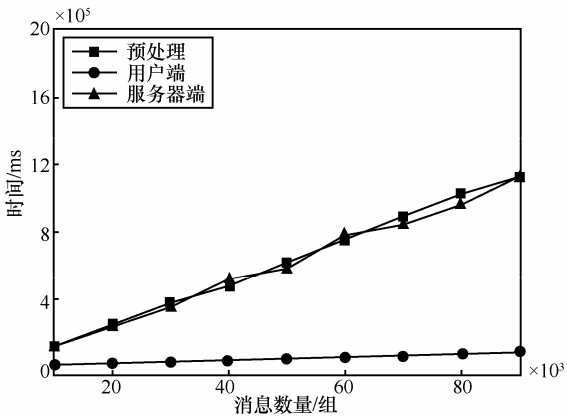


图 3　A-PDP 方案性能

根据实验结果推算，测试平台的预处理性能仅为 76.6 kbit/s，处理普通大小的文件也需要较长时间。但预处理过程是数据拥有者独立执行的，与云服务器和用户无关。在方案实际部署时，数据拥有者可以把预处理交给高性能的可信第三方完成。因此，预处理性能对方案部署不会产生较大影响。另外，云服务器的运算性能通常也远高于嵌入式测试平台，方案部署时服务器端运算消耗的时间会大幅减少。影响方案整体性能的关键是用户端的运算量，而本文方案有效降低了用户端运算量，且在数据分组数量增加时，用户端运算量增加较小。

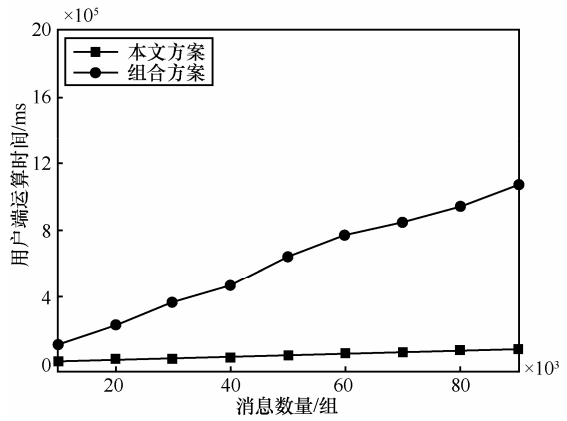


图 4　用户端运算时间对比

7　结束语

云应用中，通过一次会话同时实现认证密钥协商和数据持有性证明的密码方案具有研究意义和实用价值。本文提出了一种支持身份认证的数据持有性证明方案，给出了方案定义和具体构造，并在随机预言机模型中证明了方案的安全性。根据性能分析结果，本文方案用户端运算量低、交互步骤少，

满足云应用对方案功能、性能和安全性等方面的多重需求。下一步将研究密文搜索与数据持有性证明方案融合的可行性。

参考文献：

[1] BUYYA R, YEO C S, VENUGOPAL S, et al. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility[J]. Future Generation Computer Systems, 2009, 25(6):599 -616.

[2] WU J, SHEN Q, WANG T, et al. Recent advances in cloud security[J]. Journal of Computers, 2014, 5(10):2156-2163.

[3] ATENIESE G, BURNS R, CURTMOLA R, et al. Provable data possession at untrusted stores[C]//Proceedings of the 14th ACM Conference on Computer and Communications Security. ACM, 2007: 598-609.

[4] BOWERS K D, JUELS A, OPREA A. Proofs of retrievability: theory and implementation[C]// ACM Cloud Computing Security Workshop, CCSW 2009. Chicago, Il, USA, 2009:43-54.

[5] CHOUDHURY A J, KUMAR P, SAIN M, et al. A strong user authentication framework for cloud computing[C]// IEEE Asia-Pacific Services Computing Conference. Jeju, Korea, 2011: 110-115.

[6] LIU C, CHEN J, YANG L T, et al. Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates[J]. IEEE Transactions on Parallel & Distributed Systems, 2014, 25(9):2234-2244.

[7] GRITTI C, SUSILO W, PLANTARD T. Efficient dynamic provable data possession with public verifiability and data privacy[M]// Information Security and Privacy. Springer International Publishing, 2015:395-412.

[8] WANG B, LI B, LI H. Panda: public auditing for shared data with efficient user revocation in the cloud[J]. IEEE Transactions on Services Computing, 2015 (1): 92-106.

[9] SHEN S T, ZENG W G. Delegable provable data possession for remote data in the clouds[M]// Information and Communications Security. Springer Berlin Heidelberg, 2011:93-111.

[10] WANG H. Proxy provable data possession in public clouds[J]. IEEE Transactions on Services Computing, 2013, 6(4):551-559.

[11] RUAN H M, LEI C L. Fine-grained audit privilege control for integrity audit on cloud storage[C]// 2014 Ninth Asia Joint Conference on Information Security (ASIA JCIS). IEEE, 2014: 156-163.

[12] XU J, CHEN W, JI S, et al. A novel preserving client privacy and designate verifier auditing scheme for cloud storage[J]. International Journal of Security and Its Applications, 2015, 9(1): 295-304.

[13] NIMMY K, SETHUMADHAVAN M. Novel mutual authentication protocol for cloud computing using secret sharing and steganography[C]//Applications of Digital Information and Web Technologies. IEEE, 2014:101-106.

[14] HAO Z, ZHONG S, YU N, et al. A time-bound ticket-based mutual authentication scheme for cloud computing[J]. International Journal of Computers Communications & Control, 2011, VI(2):227-235.

[15] HUANG J J, JUANG W S, FAN C I, et al. Robust and privacy protection authentication in cloud computing[J]. International Journal of Innovative Computing, Information and Control, 2013, 9(11): 4247-4261.

[16] NAGARAJU S, PARTHIBAN L. SecAuthn: provably secure multifactor authentication for the cloud computing systems[J]. Indian Journal of Science and Technology, 2016, 9(9).

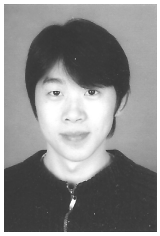
[17] HAHN C, KWON H, KIM D, et al. Enhanced authentication for outsourced educational contents through provable block possession[J]. Multimedia Tools & Applications, 2015,23:1-20.

[18] CREMERS C, FELTZ M. Beyond eCK: perfect forward secrecy under actor compromise and ephemeral-key reveal[J]. Designs, Codes and Cryptography, 2015, 74(1): 183-218.

[19] KRAWCZYK H. HMQV: a high-performance secure Diffie-Hellman protocol[C]//Advances in Cryptology–CRYPTO 2005. Springer Berlin Heidelberg, 2005: 546-566.

[20] SCOTT M. MIRACL - multiprecision integer and rational arithmetic C/C++ library (1988-2007)[EB/OL]. http://www3.cs.stonybrook.edu/~algorith/implement/shamus/distrib/ miracl3.zip.

作者简介：



李昊星（1982-），男，满族，河南方城人，西安电子科技大学博士生，主要研究方向为网络与系统安全、云数据安全。

李凤华（1966-），男，湖北浠水人，博士，中国科学院信息工程研究所副总工、研究员、博士生导师，主要研究方向为网络与系统安全、信息保护、隐私计算。

宋承根（1987-），男，贵州锦屏人，博士，北京电子科技学院讲师，主要研究方向为信息安全、密码学。

阎亚龙（1976-），男，山西兴县人，北京电子科技学院高级工程师，主要研究方向为信息安全工程。