无第三方服务器的基于数据流行度的加密去重方案

哈冠雄[1,2]，贾巧雯[3]，陈杭[1,2]，贾春福[1,2]
（1. 南开大学网络空间安全学院，天津 300350；2. 天津市网络与数据安全技术重点实验室，天津 300350；
3. 中国科学院软件研究所，北京 100190）

摘 要：在加密去重系统中，基于流行度为数据设定不同级别的安全保护可有效平衡数据安全与存储效率。现有方案均需引入第三方服务器协助统计数据流行度，而第三方易成为单点故障和效率瓶颈。针对此问题，提出了一个无第三方服务器的基于数据流行度的加密去重方案，基于 Count-Min sketch 算法和 Merkle Puzzles 协议实现数据流行度的安全统计，并通过用户间执行 sPAKE 协议实现不流行数据的加密去重。安全性分析和实验评估表明所提方案是安全且高效的。

关键词：云存储；加密去重；数据流行度；Count-Min sketch 算法；sPAKE 协议
中图分类号：TP309.2
文献标志码：A
DOI: 10.11959/j.issn.1000−436x.2022151

Data popularity-based encrypted deduplication scheme without third-party servers

HA Guanxiong[1,2], JIA Qiaowen[3], CHEN Hang[1,2], JIA Chunfu[1,2]
1. College of Cyber Science, Nankai University, Tianjin 300350, China
2. Tianjin Key Laboratory of Network and Data Security Technology, Tianjin 300350, China
3. Institute of Software, Chinese Academy of Sciences, Beijing 100190, China

Abstract: It is effective to balance data security and storage efficiency for setting different levels of security protection for data based on popularity in encrypted deduplication systems. Existing schemes all need introduce a trusted third-party for recording data popularity, while the third party is prone to a single point of failure and efficiency bottleneck. To address this problem, a popularity-based encrypted deduplication scheme without third-party servers was proposed, which accurately recorded the data popularity based on the Count-Min sketch algorithm and Merkle Puzzles protocol, and achieved encrypted deduplication of unpopular data through the sPAKE protocols performed among users. Security analysis and experimental evaluation show that the proposed scheme is secure and efficient.

Keywords: cloud storage, encrypted deduplication, data popularity, Count-Min sketch algorithm, sPAKE protocol

0 引言

云存储技术[1]的快速发展使越来越多的用户选择将数据外包至云服务器（CS, cloud server）。随着数据量的不断增加，如何高效存储海量数据成为云服务提供商面临的一个关键问题。数据去重[2-3]是一种节省存储开销的有效方法，云服务器可基于去重技术识别用户间上传的重复数据内容，删除其中的冗余部分以节省存储开销。近年来，隐私泄露问题受到了越来越多的关注，数据的隐私保护变得愈发重

收稿日期：2022−06−08；**修回日期：**2022−07−27
通信作者：贾春福，cfjia@nankai.edu.cn
基金项目：国家重点研发计划基金资助项目（No.2018YFA0704703）；国家自然科学基金资助项目（No.61972215, No.61972073, No.62172238）；天津市自然科学基金资助项目（No.20JCZDJC00640）
Foundation Items: The National Key Research and Development Program of China (No.2018YFA0704703), The National Natural Science Foundation of China (No.61972215, No.61972073, No.62172238), The Natural Science Foundation of Tianjin (No.20JCZDJC00640）

要。传统加密算法可将数据转换为与随机比特串不可区分的密文形式，为其提供安全级别较高的语义安全。然而，传统加密算法与去重技术难以兼容。不同用户的加密密钥不同，跨用户上传的相同数据在加密后将变成完全随机的密文，这为重复数据的检测带来了困难。收敛加密（CE, convergent encryption）[4]使用数据哈希值作为加密密钥，首次实现了加密去重。但 Bellare 等[5]指出 CE 只能提供收敛安全性，其仅能为不可预测数据（明文空间无限，敌手无法遍历）提供语义安全。若使用 CE 加密可预测数据，易受到离线字典攻击。

因此，在实现数据去重的同时提供语义安全是一个关键挑战。针对此问题，一些研究工作[6-9]基于数据流行度设计加密去重方案，旨在兼顾存储效率与数据安全。数据流行度指的是数据的所有者数量。具体而言，首先设定一个流行度阈值 t，若外包数据的所有者数量超过 t，则认为是流行数据，否则认为是不流行数据。对于流行歌曲、热门电影等流行数据，系统基于 CE 为其提供安全级别较低的收敛安全性；对于研究报告、医疗记录等不流行数据，系统使用随机加密为其提供安全级别较高的语义安全。基于流行度对用户数据进行分类，可有效平衡存储效率与数据安全。Stanek 等[6]使用真实数据集分析了基于流行度的加密去重系统的存储效率，结果表明当流行数据的数量较多时，方案具有较高的存储效率。其他相关研究工作[7-9]也充分说明了基于流行度设计加密去重方案这一思路的可行性与实用性。

在基于流行度的加密去重中，如何安全准确地统计数据流行度是方案设计中面临的关键挑战。现有方案及其特点如表 1 所示，一些经典的加密去重方案（如 CE[4]、DupLESS[5]）将数据哈希值作为去重标签用于重复性检测，易使不流行数据受到离线字典攻击；一些现有方案[6-7]部署可信第三方协助统计数据流行度，这一方法的弊端主要有以下两点。1) 系统中的所有用户都需要与第三方交互，影响了系统的可扩展性。当用户量较多、数据量较大时，第三方服务器的处理能力有限，易成为系统的效率瓶颈。2) 第三方可得到系统中所有数据的流行信息，因此需要假设第三方是完全可信的，这一假设在真实场景中难以实现。并且，第三方易成为系统中的单点故障，一旦其被敌手攻破，所有不流行数据的语义安全都将被破坏。

表 1 现有方案及其特点

方案	特点	缺陷
文献[4-5]	哈希值作为去重标签	易受离线字典攻击
文献[6-7]	部署可信第三方	存在效率瓶颈与单点故障
文献[8-9]	部署半可信第三方	存在效率瓶颈

文献[8-9]弱化了第三方的安全假设，通过引入半可信（即诚实但好奇）的第三方来统计数据流行度。Ha 等[8]通过部署一个半可信的第三方同态解密服务器统计数据流行度，其中的解密服务器同样易成为系统中的效率瓶颈。Gao 等[9]通过引入第三方密钥管理服务器实现了基于流行度的加密去重，该方案使用收敛密文的哈希值作为去重标签，易使不流行数据受到离线字典攻击。

由于引入第三方服务器会存在效率瓶颈和单点故障的问题，本文提出了一个无第三方服务器的基于数据流行度的加密去重方案。该方案基于 Count-Min sketch（CM-sketch）算法[10]、sPAKE（symmetric password-authenticated key exchange）协议[11]以及 Merkle Puzzles 协议[12]，在不需要部署任何第三方服务器的情况下实现了加密去重系统中的数据流行度精确检测，并可分别为流行数据和不流行数据提供收敛安全和语义安全。本文的主要贡献如下。

1) 本文方案设计了两步检测的方法安全精确地统计数据流行度。首先，基于 CM-sketch 算法在仅使用固定内存空间的前提下实现了数据流行度的初步检测，有效过滤了大量不流行数据。然后，通过用户与云服务器间执行 Merkle Puzzles 协议进一步准确统计数据流行度。流行度检测过程仅由云服务器和用户两方完成，不需要任何第三方服务器。

2) 通过在用户间执行 sPAKE 协议，本文方案实现了不流行数据的去重检测和用户间的密钥传递，云服务器可同时对流行数据和不流行数据实现客户端加密去重，最大限度地节省了系统的通信开销和存储开销。

3) 本文方案设计了密钥验证和密文验证的过程，并结合所有权证明（PoW, proof of ownership）[13]技术，可有效对抗所有权欺骗攻击[13-14]、文件伪造攻击[15]和字典攻击[15]等加密去重系统中的常见攻击，可分别为流行数据和不流行数据提供收敛安全性和语义安全性。

1　相关工作

近年来，国内外的多个研究团队均在加密去重领域进行了深入研究。Douceur 等[4]提出了 CE，首次实现了加密去重。Bellare 等[15]将 CE 形式化定义为消息锁加密（MLE, message-locked encryption），并指出 CE 无法为可预测数据实现语义安全。为此，Bellare 等提出了 DupLESS[5]，其引入一个密钥服务器协助用户加密数据，有效防止了字典攻击。Halevi 等[14]发现了客户端去重中的所有权欺骗问题。针对此问题，文献[13-14,16]提出了所有权证明方案。此外，文献[17-19]针对加密去重系统中的密钥更新与访问控制问题进行了研究。Li 等[20]提出了加密去重系统中的频率分析攻击，并给出了相应的防御方案[20-21]。文献[22-24]针对客户端去重中的侧信道攻击问题提出了防御方案。

在加密去重这一研究领域中，基于数据流行度设计加密去重方案[6-9]是一个重要的研究分支。Stanek 等[6]基于门限密码系统和双层加密实现了基于流行度的加密去重，当超过流行度阈值 t 个的用户上传某一密文后，云服务器可对外层的随机密文进行解密，保留内层的收敛密文，实现加密去重。Puzio 等[7]利用完美哈希实现了基于流行度的加密去重，用户可在与云服务器的交互中获取外包数据的流行信息。文献[6-7]需要引入可信第三方协助统计数据流行度，具有一定的局限性。为此，Ha 等[8]基于同态加密实现了数据流行度的安全统计，客户端上传外包数据的随机标签至云服务器，云服务器通过与同态解密服务器的交互进行流行度检测。Gao 等[9]基于双层加密和密钥共享设计了基于流行度的加密去重方案，将第三方的安全假设降低为诚实但好奇的。然而，现有方案都需要引入第三方服务器，易出现效率瓶颈等问题，影响了方案在真实场景中的实用性。为此，本文提出了无第三方服务器的基于数据流行度的加密去重方案。

2　预备知识

2.1　CM-sketch

CM-sketch 可在误差较小的前提下利用有限的内存空间描述数据的频率特征，其内部的数据结构是一个 $w\times d$ 的二维数组，其中 $w=\left\lceil \dfrac{\mathrm{e}}{\varepsilon} \right\rceil, d=\left\lceil \ln\dfrac{1}{\delta} \right\rceil$，参数 ε、δ 表示在 $1-\delta$ 的概率下统计结果的总误差小于 ε。CM-sketch 算法主要由以下 3 个算法组成。

1) Setup(ε,δ)。输入参数 ε,δ，初始化一个 $w\times d$ 的二维数组 count，将所有元素值置 0；随机选定 d 个两两独立的哈希函数 $h_1(\cdot),h_2(\cdot),\cdots,h_d(\cdot)$，其中 $h_i(\cdot)$ $(i\in[1,d])$ 的输出结果的长度为 w。

2) Count(x,count)。输入元素 x 和数组 count，计算哈希值 $h_1(x),h_2(x),\cdots,h_d(x)$ 用于更新 count，令 count 中被哈希值映射到的位的计数值加 1，即 $\text{count}[i][h_i(x)]=\text{count}[i][h_i(x)]+1$，其中 $i\in[1,d]$。

3) Freq(x,count)。输入元素 x 和数组 count，输出 x 的频率信息。计算哈希值 $h_1(x),h_2(x),\cdots,h_d(x)$，取 $\{\text{count}[i][h_i(x)]\,|\,i\in[1,d]\}$ 中的最小值作为元素 x 的频率信息输出。

2.2　sPAKE

sPAKE[11,25]是传统密钥协商的一个扩展。当运行协议的双方共享同一个口令时，可协商出相同的密钥；若双方口令不同，则各自得到一个随机密钥，双方均无法获得对方密钥的任何信息。

本文使用了目前已知最高效的 sPAKE[11]，参与协议的双方仅需执行 2 次幂运算，具有很高的执行效率，并且协议在随机预言机模型下是可证明安全的。具体的协议流程如图 1 所示。本文方案使用数据哈希值取代了原协议中的口令。若参与 sPAKE 协议的双方拥有相同的哈希值，可协商出相同的输出结果 K_1 和 K_2，否则，双方得到完全随机的输出结果。

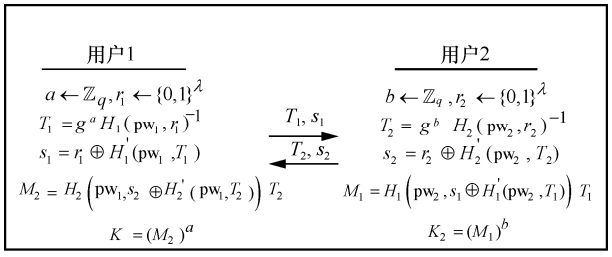

图 1　sPAKE 协议流程

2.3　Merkle Puzzles

Merkle Puzzles 基于对称密码原语实现了安全的密钥协商。假设协商密钥的双方分别为 Alice 和 Bob，他们执行以下步骤协商密钥。

1) Alice 穷举密钥集合中的所有密钥 $\{k_1,k_2,\cdots,k_n\}$，并生成 n 个密文 $\{C_i=\text{Enc}(k_i,0),\ \text{Enc}(k_i,x_i),\ \text{Enc}(k_i,s_i)\}_{i=1}^n$，其中 x_i 和 s_i 为随机值，Enc$(\cdot)$ 为对称加密算法。Alice 将 $\{C_i\}_{i=1}^n$ 随机排序后发送给 Bob。

2) Bob 随机选择一个密文 $C_j=(c_1,c_2,c_3)$，使用 $\{k_1,k_2,\cdots,k_n\}$ 中的所有密钥尝试解密 c_1。当解密结果为 0 时，Bob 确定对应的密钥为 k_j，然后使用 k_j 解密 c_2 和 c_3 得到 x_j 和 s_j，并将 x_j 返回给 Alice。

3) Alice 收到 s_j 后密钥协商结束，双方得到的协商密钥为 k_j。

本文基于上述设计思路和 Yu 等[26]对 Merkle Puzzles 的使用方式，将 Merkle Puzzles 应用于加密去重场景下的数据流行度统计中。

2.4 收敛加密

收敛加密[4]使用消息内容的哈希值作为密钥加密数据以实现密文去重，由以下 3 个算法组成。

1) CE.KG(M)：输入消息 M，输出收敛密钥 k_c。

2) CE.Enc(k_c,M)：输入收敛密钥 k_c 和消息 M，输出收敛密文 C。

3) CE.Dec(k_c,C)：输入收敛密钥 k_c 和收敛密文 C，输出消息 M。

3 系统模型与定义

3.1 系统架构

本文方案的系统架构如图 2 所示，包括用户 U 和云服务器 CS。CS 为 U 提供数据存储服务，基于数据的所有者数量将其划分为流行数据和不流行数据，可对 U 的外包数据进行去重以节省存储开销。U 外包加密数据至 CS 以节省本地存储开销，并且需要在上传数据后在 CS 的协助下与其他上传数据的用户运行 sPAKE 协议，以确定后续上传数据的流行情况。

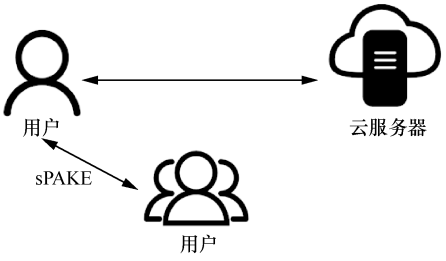

图 2　系统架构

3.2 威胁模型

本文主要考虑以下两类敌手。

1) 内部敌手。内部敌手指系统中诚实但好奇的云服务器和合法用户，其会诚实地执行方案中设定的协议流程，同时也会试图窥探用户的数据信息。

2) 外部敌手。外部敌手指系统中的恶意攻击者，其可在与云服务器交互的任何阶段发起攻击。具体来说，外部敌手可通过窃听信道得到用户上传的数据内容，发起字典攻击试图恢复数据信息，发起所有权欺骗攻击试图篡改数据流行度[9]或骗取数据所有权，发起密钥伪造攻击或文件伪造攻击试图破坏数据完整性。

3.3 安全目标

本文方案的安全目标如下。

1) 数据机密性。任何敌手均无法恢复不流行数据的任何信息。对于流行数据，本文方案提供收敛安全性。

2) 数据完整性。任何敌手不能篡改用户的外包数据，所有合法用户均可正确恢复其外包数据并验证数据完整性。

3.4 符号定义

本文使用的符号及其含义如表 2 所示。

表 2　　符号及其含义

符号	含义
t	流行度阈值
sh(·)	短哈希函数
$H(\cdot)$	哈希函数（如 SHA-256）
$H(k,\cdot)$	密钥为 k 的 HMAC
Enc(·)	对称加密算法（如 AES）
Dec(·)	对称解密算法（如 AES）
$E_{\mathrm{pk}}(\cdot)$	RSA 加密算法
$D_{\mathrm{sk}}(\cdot)$	RSA 解密算法
k_c	收敛密钥
k_r	随机密钥
C_F	收敛密文
C_r	随机密文
H_{CE}	收敛密文的哈希值
H_C	随机密文的哈希值
p_F	所有权证明值
pt	所有权证明的辅助信息
num	数据的所有者数量

4 设计思路

基于流行度设计加密去重系统存在以下2个挑战。

1) 如何安全准确地统计数据流行度。若使用数据哈希值（如 SHA-256 值）进行流行度检测，不流行数据易受到离线字典攻击。若使用随机标签[9]进行流行度检测，在不引入第三方服务器的情况下难

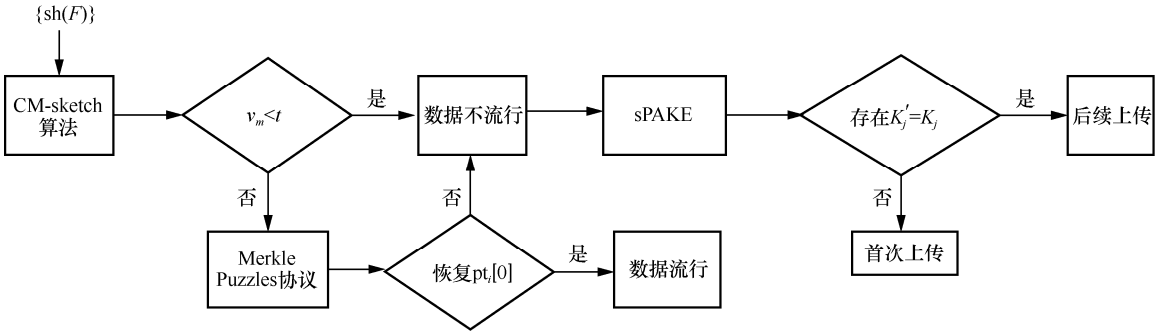

图 3　方案流程

以判断数据流行度。

2) 如何实现不流行数据的加密去重。由于需要为不流行数据提供语义安全，数据哈希值不可用于去重检测，并且用户需要使用随机密钥加密不流行数据。如何实现不流行数据的跨用户去重检测，并在不同用户间传递随机密钥以实现加密去重成为设计中的关键挑战。

4.1　流行度检测的设计思路

针对第一个挑战，本文设计了一个两步检测的方案来安全准确地统计数据流行度，如图 3 所示。首先，使用外包数据的短哈希值进行流行度的初步检测，旨在过滤掉大量不流行数据。短哈希值[26-27]可通过截取哈希值的部分位数来实现（如截取 SHA-256 值的前 13 位）。由于短哈希值具有较高的冲突率，敌手无法基于短哈希值进行字典攻击。然而，当数据量较大时，统计大量短哈希值的所有者数量可能会给云服务器带来较大的内存开销。为此，本文在初步检测中使用了 CM-sketch 算法，其特点是可以基于固定长度的内存空间进行流行度统计，有利于系统的可扩展性。即使数据量不断增加，用于统计流行度的内存空间仍可保持不变。

CM-sketch 的统计结果存在一定误差，与真实结果相比可能偏大。若 CM-sketch 的统计值 v_m 小于流行度阈值 t，则数据一定不流行；若 v_m 大于或等于 t，需要进行后续检测。在后续检测中，云服务器与用户执行 Merkle Puzzles 协议以检测外包数据的哈希值（如 SHA-256 值）是否与现有流行数据的哈希值匹配。若用户可恢复出正确的 $pt_i[0]$，则数据流行；否则，数据不流行。在后续检测中，云服务器可准确判断数据是否流行。

4.2　不流行数据加密去重的设计思路

云服务器若检测到外包数据是流行数据，可基于 CE 实现加密去重；若检测到外包数据是不流行数据，如图 3 所示，可基于 sPAKE 协议进行去重检测，并在用户间传递随机密钥，以实现不流行数据的加密去重。假设用户 U 的外包文件为 F，云服务器索引短哈希值 sh(F) 对应的不流行数据的用户列表为 $\{U_i\}_{i=1}^{l}$（云服务器的存储结构如图 4 所示），要求 U 与 $\{U_i\}_{i=1}^{l}$ 使用数据哈希值运行 sPAKE 协议。协议结束后，云服务器可通过协议的输出结果进行去重检测，判断本次数据上传为首次上传或后续上传。若数据为首次上传，用户执行首次上传的流程；若数据为后续上传，执行密钥传递的数据过程在用户间安全地传递随机密钥，实现不流行数据的加密去重。

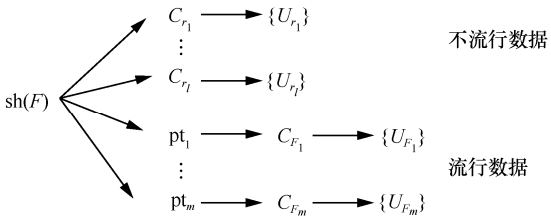

图 4　云服务器的存储结构

5　方案详细设计

本文方案流程主要分为系统初始化、流行度检测、不流行数据上传、流行数据上传和数据下载。

5.1　系统初始化

用户生成各自的 RSA 公私钥对 (pk,sk)。为防止在线字典攻击，用户为每个外包文件设置一个参与 sPAKE 协议的数量上限 N_P [27]。若用户在某一时间段 Δ 内收到了对某一个文件超过 N_P 次的 sPAKE 请求，则拒绝执行协议。云服务器 CS 设定系统的安全参数 λ 和流行度阈值 t。若数据的所有者数量 num $\geqslant t$，则认为是流行数据；否则，认为是不流行数据。此外，云服务器设定 CM-sketch 的参数。CM-sketch 由一个二维数组组成，包含 r 行、w 列，共 $r \times w$ 个计数器。云服务器发布用于 CM-sketch

的 r 个独立的短哈希函数 $\{\mathrm{sh}_i(\cdot)\}_{i=1}^{r}$，每个短哈希函数均可将数据对应到 CM-sketch 的 r 行中的一个计数器 $j(1\leqslant j\leqslant w)$ 上。

5.2 流行度检测

流行度检测的流程如算法 1 所示，具体可分为基于 CM-sketch 的初步检测和基于 Merkle Puzzles 的后续检测，详细流程如下。

1) 基于 CM-sketch 的初步检测。假设外包文件为 F，用户 U 计算短哈希值 $\mathrm{sh}(F)$ 和 $\{\mathrm{sh}_i(F)\}_{i=1}^{r}$ 并将其发送至云服务器 CS。CS 将 $\{\mathrm{sh}_i(F)\}_{i=1}^{r}$ 中的每个值映射到 CM-sketch 的 r 行中的计数器上，并将相应计算器的值加 1。统计流行度时，云服务器使用 r 个计数值中的最小值 v_m 作为 $\mathrm{sh}(F)$ 当前的流行度统计值。CM-sketch 的统计值与真实结果相比可能偏大。因此，若 $v_m<t$，则 F 一定是不流行数据，CS 与 U 执行不流行数据上传的步骤；若 $v_m\geqslant t$，需要进行后续检测进一步准确判断流行度。

2) 基于 Merkle Puzzles 的后续检测。云服务器 CS 在流行数据列表中检索 $\mathrm{sh}(F)$（云服务器的存储结构如图 4 所示）。若无法检索到 $\mathrm{sh}(F)$，则 F 是不流行数据；否则，CS 检索 $\mathrm{sh}(F)$ 对应的流行数据列表中的所有辅助信息 $\left\{\mathrm{pt}_i=\left(H(H_{F_i},r_{M_i}),\mathrm{Enc}(H_{F_i},r_{M_i})\right)\right\}_{i=1}^{m}$ [26]（其生成细节见 5.3.4 节），其中 H_{F_i} 表示文件哈希值。CS 发送 $\{\mathrm{pt}_i[1]\}_{i=1}^{m}$ 至用户 U。U 基于文件 F 的哈希值 H_F 解密 $\{\mathrm{pt}_i[1]\}_{i=1}^{m}$，得到随机值集合 $\{r'_{M_i}\}_{i=1}^{m}$，计算哈希值 $\{H(H_F,r'_{M_i})\}_{i=1}^{m}$ 并返回 CS。CS 检查是否存在 $H(H_F,r'_{M_j})=\mathrm{pt}_j[0]$。若存在，说明 F 为流行数据；否则，F 为不流行数据。至此，流行度检测结束，CS 可精确统计文件 F 的流行度。

算法 1 流行度检测

1) $U\rightarrow\mathrm{CS}$: 发送 $\left(\mathrm{sh}(F),\{\mathrm{sh}_i(F)\}_{i=1}^{r}\right)$ 至 CS

2)CS: 输入 $\{\mathrm{sh}_i(F)\}_{i=1}^{r}$ 至 CM-sketch，得到统计结果 v_m

3) if $v_m\geqslant t$ && $\mathrm{sh}(F)$ 在流行数据列表中, then

4) $\mathrm{CS}\rightarrow U$: 发送 $\{\mathrm{pt}_i[1]\}_{i=1}^{m}$ 至 U

5) $U\rightarrow\mathrm{CS}$: 发送 $\{H(H_F,r'_{M_i})\}_{i=1}^{m}$ 至 CS

6)if 存在 j, 满足 $\{H(H_F,r'_{M_i})==\mathrm{pt}_j[0]$, then

7) return 流行数据

8) else, return 不流行数据

9) end if

5.3 不流行数据上传

不流行数据上传的流程包括去重检测、数据首次上传、数据后续上传和流行度转换。

5.3.1 去重检测

去重检测的流程如算法 2 所示。若检测到文件 F 不流行，云服务器 CS 索引 $\mathrm{sh}(F)$ 对应的所有不流行数据的用户列表，在每个列表中选择一个当前在线的用户组成检查者列表 $\{U_i\}_{i=1}^{l}$，要求 U 与 $\{U_i\}_{i=1}^{l}$ 中的用户运行 sPAKE 协议。在 sPAKE 协议中，U 输入文件 F 的哈希值 H_F，每个 U_i 输入各自的文件哈希值 H_{F_i}，协议中的通信数据均由 CS 转发，用户间不需要直接通信。协议结束后，每个 U_i 得到 sPAKE 协议的输出值 K_i，U 得到输出值列表 $\{K'_i\}_{i=1}^{l}$，U 和 $\{U_i\}_{i=1}^{l}$ 将输出值发送至 CS。CS 收到 U 发送的 $\{K'_i\}_{i=1}^{l}$ 和 $\{U_i\}_{i=1}^{l}$ 发送的 $\{K_i\}_{i=1}^{l}$ 后，检查是否存在 $K'_j=K_j$。若存在，说明 U 的外包文件 F 与 U_j 此前上传的文件相同，U 执行数据后续上传的流程；否则，U 执行数据首次上传的流程。

算法 2 去重检测

1)CS: 构造检查者列表 $\{U_i\}_{i=1}^{l}$

2) $\mathrm{CS}\rightarrow(U,\{U_i\}_{i=1}^{l})$: 发送 sPAKE 请求至 U 和 $\{U_i\}_{i=1}^{l}$

3) U 和 $\{U_i\}_{i=1}^{l}$: 分别输入文件哈希值 H_F 和 $\{H_{F_i}\}_{i=1}^{l}$ 至 sPAKE 协议

4) $(U,\{U_i\}_{i=1}^{l})\rightarrow\mathrm{CS}$: 分别发送协议输出结果 $\{K'_i\}_{i=1}^{l}$ 和 $\{K_i\}_{i=1}^{l}$ 至 CS

5)CS:

6) if 存在 j, 满足 $K'_j==K_j$, then

7) return 后续上传

8) else, return 首次上传

9) end if

5.3.2 数据首次上传

用户 U 生成收敛密钥 $k_c=\mathrm{CE.KG}(F)$，对 F 进行收敛加密得到收敛密文 C_F=CE.Enc(k_c,F)，计算哈希值 $H_{\mathrm{CE}}=H(C_F)$；生成随机密钥 $k_r\leftarrow\{0,1\}^{\lambda}$ 和随机值 $s\leftarrow\{0,1\}^{\lambda}$，对 F 进行随机加密得到 C_r=Enc(k_r,F)；计算所有权证明值 p_F=$H(s,F)$；上传 C_r 至 CS，本地存储 $(H_F,k_r,k_c,p_F,s,H_{\mathrm{CE}})$。

5.3.3 数据后续上传

数据后续上传的流程如算法 3 所示。方案实现了不流行数据的客户端加密去重，若数据为后续上传，则不需要用户 U 上传完整的数据内容，仅需进行所有权证明和密钥验证的过程，可有效节省系统的通信开销。在后续上传中，CS 发送 U 的公钥 pk_U 至检查者列表中的 U_j。U_j 计算 $C_k = k_r \oplus p_F$ 和 $C_s = E_{\mathrm{pk}_U}(s)$ 并将其返回至 CS，二者用于密钥传递和所有权证明。CS 将 (C_k, C_s) 转发至 U。U 使用私钥 sk_U 解密 C_s 得到 $s=D_{\mathrm{sk}_U}(C_s)$；使用 s 和 F 计算所有权证明值 $p_F=H(s,F)$，恢复随机密钥 $k_r = C_k \oplus p_F$；计算收敛密钥 $k_c = \mathrm{CE.KG}(F)$ 和哈希值 $H_{\mathrm{CE}} = H(C_F)$；本地存储 $(H_F, k_r, k_c, p_F, s, H_{\mathrm{CE}})$。

算法 3　数据后续上传

1) $\mathrm{CS} \to U_j$: 发送 pk_U 至 U_j

2) $U_j \to \mathrm{CS}$: 计算 $C_k = k_r \oplus p_F$ 和 $C_s = E_{\mathrm{pk}_U}(s)$，发送 (C_k, C_s) 至 CS

3) $\mathrm{CS} \to U$: 发送 (C_k, C_s) 至 U

4) $U \to \mathrm{CS}$: 计算 $s=D_{\mathrm{sk}_U}(C_s)$，$p_F=H(s,F)$，$k_r = C_k \oplus p_F$，$k_c = \mathrm{CE.KG}(F)$，$H_{\mathrm{CE}} = H(C_F)$，$C_r^{'}=\mathrm{Enc}(k_r, F)$ 和 $H_C = H(C_r^{'})$，发送 H_C 至 CS

5) CS:

6) if $H_C == H(C_r)$, then

7) 在数据的所有者列表中加入 U，并令 num 加 1

8) if num $== t$, then

9) return 流行度转换

10) else, return 成功

11) end if

12) else, return 错误

13) end if

为防止密钥传递时出现伪造情况，方案设计了密钥验证过程。U 使用 k_r 计算随机密文 $C_r^{'}=\mathrm{Enc}(k_r, F)$，计算哈希值 $H_C = H(C_r^{'})$ 并发送至 CS。CS 检索本地存储的随机密文 C_r，验证 H_C 是否与 $H(C_r)$ 相等。若不相等，说明存在密钥伪造攻击，CS 不进行数据去重以防止数据完整性被破坏；若相等，将 U 加入用户列表当中，并将文件的所有者数量 num 加 1。若此时 num 小于 t，上传过程结束；若 num 等于 t，表明此次数据上传后 F 变为流行数据，CS 需与 U 进行流行度转换。

5.3.4 流行度转换

在流行度转换中，用户 U 发送收敛密文 C_F 至 CS，CS 进行如下的密文验证过程。CS 计算哈希值 $H(C_F)$ 发送至 U_j，U_j 检查 $H(C_F)$ 与本地存储的 H_{CE} 是否相等。若不相等，可能存在文件伪造攻击，CS 不进行数据去重以防止数据完整性被破坏；若相等，说明 U 与 U_j 计算的收敛密文一致，云服务器存储 C_F，并删除此前存储的随机密文 C_r 以节省存储开销。最后，U 生成随机值 $r_M \leftarrow \{0,1\}^{\lambda}$，计算辅助信息 $\mathrm{pt} = (H(H_F, r_M), \mathrm{Enc}(H_F, r_M))$ 并发送至 CS，用于后续其他用户的流行度检测。

5.4 流行数据上传

若外包文件 F 是流行数据，方案可实现客户端去重，用户 U 只需与云服务器 CS 进行如下的所有权证明[9]。CS 生成随机种子 u 发送至 U。U 计算收敛密文 C_F，根据 u 生成随机索引序列 $\{u_1, u_2, \cdots, u_{n_p}\}$，计算所有权证明值 $p_F = H(H(C_F[u_1]), H(C_F[u_2]), \cdots, H(C_F[u_{n_p}]))$ 并返回至 CS。CS 基于本地存储的收敛密文 C_F 与 $\{u_1, u_2, \cdots, u_{n_p}\}$ 判断 p_F 是否正确。若正确，CS 将 U 加入所有者列表中，U 存储收敛密钥 k_c；否则，所有权证明失败。

5.5 数据下载

用户 U 向 CS 发出文件 F 的下载请求。CS 根据 F 的流行情况返回收敛密文 C_F 或随机密文 C_r。U 使用 k_c 或 k_r 解密 C_F 或 C_r 以恢复 F，并通过验证 $H(F)$ 与 H_F 是否相等来检测数据完整性是否被破坏。

6 安全性分析

本节分析了方案的正确性以及在内部敌手和外部敌手攻击下的安全性并做出如下安全假设。1) 方案中使用的对称加密和 RSA 加密具有语义安全性；2) 方案中使用的哈希函数（短哈希函数除外）满足抗碰撞性，且可看作随机预言机；3) 方案中的 sPAKE 协议是安全的，若参与协议的双方输入不同，则无法获取对方输入的任何信息；若双方输入相同，则可得到相同的协议输出结果。

6.1 正确性分析

本节分析方案在不流行数据首次上传、后续上传以及流行数据上传 3 种情况下的解密正确性。

在不流行数据的首次上传中，用户 U 本地安全存储了收敛密钥 k_c 和随机密钥 k_r。基于传统对称加密算

法和 CE 的解密正确性，下载数据时 U 可使用 k_c 或 k_r 恢复正确的数据内容。在不流行数据的后续上传中，U 收到 CS 发送的 C_k 和 C_s。基于 RSA 解密的正确性，拥有正确私钥的 U 可恢复随机值 s。若拥有完整的数据内容，U 可计算出收敛密钥 k_c 和所有权证明值 p_F=$H(s,F)$，并恢复出随机密钥 k_r。因此，U 可通过方案中的密钥传递过程得到正确的随机密钥 k_r 和收敛密钥 k_c，在数据下载时可使用 k_c 或 k_r 恢复正确的数据内容。流行数据上传时，U 本地存储了收敛密钥 k_c，下载数据时可使用 k_c 恢复正确的数据内容。

6.2 内部敌手的安全性分析

本节分析方案在内部敌手攻击下的安全性。根据 3.2 节描述的威胁模型，内部敌手是诚实但好奇的云服务器或用户。基于 sPAKE 协议[11]的安全性，参与协议的诚实用户不会获取其他用户的任何数据信息。因此，本节中的内部敌手 A_{I} 指诚实但好奇的云服务器。本节分别从流行度检测的安全性、流行数据的收敛安全性和不流行数据的语义安全性三方面进行安全性分析。

6.2.1 流行度检测的安全性

在流行度检测中，A_{I} 可得到短哈希值 $\mathrm{sh}(F)$、$\{\mathrm{sh}_i(F)\}_{i=1}^{r}$ 和哈希值集合 $\{H(H_F,r'_{M_i})\}$。由于短哈希值具有高碰撞率，A_{I} 无法将短哈希值与用户文件一一对应，因此无法通过离线字典攻击恢复数据信息。由于随机值 r'_{M_i} 对 A_{I} 保密，且哈希函数可看作随机预言机，$\{H(H_F,r'_{M_i})\}$ 与等长的随机比特串具有不可区分性，A_{I} 无法获得数据信息。

6.2.2 流行数据的收敛安全性

对于流行文件 F，A_{I} 可得到其收敛密文 C_F、密文哈希值 H_{CE}、所有权证明值 p_F 和辅助信息 pt。基于收敛安全性[5]的定义，A_{I} 通过 (C_F,H_{CE}) 恢复不可预测的流行数据的概率是可忽略的；p_F 为 C_F 抽样后的哈希值，同样不泄露不可预测数据的任何信息；pt[0] 和 pt[1] 分别为随机值 r_M 的哈希值和对称密文，由于哈希函数可看作随机预言机且对称加密具有语义安全性，因此 pt 与等长的随机比特串具有不可区分性，不泄露数据信息。

6.2.3 不流行数据的语义安全性

本节通过定义安全游戏的方式形式化分析不流行数据的语义安全性。安全游戏将方案的安全性规约到对称加密、RSA 加密和密码本加密的安全性上。本节定义安全游戏 G_0 模拟真实场景下的方案，其中敌手 $\mathcal{A}$ 模拟诚实但好奇的云服务器，挑战者 $\mathcal{C}$ 模拟诚实执行协议的用户。本节将安全游戏中的哈希函数看作随机预言机，挑战者 $\mathcal{C}$ 计算哈希值时需询问随机预言机。对于每次询问，预言机将返回一个均匀随机的输出。对于重复出现的询问，预言机将返回相同的结果。安全游戏 G_0 的具体描述如下。

1) 初始化阶段。挑战者生成随机密钥 k_r、随机种子 s 和 RSA 公私钥对 $(\mathrm{pk}_u,\mathrm{sk}_u)$。

2) 挑战阶段。敌手 $\mathcal{A}$ 输出 2 个等长的挑战明文 (M_0, M_1)。$\mathcal{C}$ 随机选择 $b\in_R\{0,1\}$，计算 $P_F=H(s,M_b)$，返回 $C_r=\mathrm{Enc}(k_r,M_b)$、$C_s=E_{\mathrm{pk}_u}(s)$ 和 $C_k=k_r\oplus p_F$ 至 $\mathcal{A}$。

3) 猜测阶段。$\mathcal{A}$ 输出 b'，如果 $b=b'$，则 $\mathcal{A}$ 赢得 G_0。将 $\mathcal{A}$ 在 G_0 中具有的概率优势定义为 $\mathrm{Adv}(\mathcal{A})=\left|\Pr[b=b']-\frac{1}{2}\right|$。

定义 1 若对于任意的概率多项式时间（PPT, probabilistic polynomial time）敌手 $\mathcal{A}$，存在一个可忽略的值 ε，满足 $\mathrm{Adv}(\mathcal{A})\leqslant\varepsilon$，则认为本文方案中诚实但好奇的云服务器无法破坏不流行数据的语义安全性。

定理 1 若方案中使用的对称加密和 RSA 加密是语义安全的，则在随机预言机模型下本文方案中的云服务器无法破坏不流行数据的语义安全性。

这里通过定义多个不可区分的安全博弈游戏来证明定理 1。

G_0^{I}：该游戏与 G_0 相同。

G_1^{I}：该游戏与 G_0^{I} 类似，区别仅在于挑战者 $\mathcal{C}$ 将随机密文 C_r 替代为等长的随机比特串 str_r。由于对称加密具有语义安全性，在密钥 k_r 对敌手 $\mathcal{A}$ 保密的情况下，G_1^{I} 与 G_0^{I} 具有不可区分性。

G_2^{I}：该游戏与 G_1^{I} 类似，区别仅在于 $\mathcal{C}$ 将 C_s 替换为等长的随机比特串 str_s。由于 RSA 加密具有语义安全性，在私钥 sk_U 对敌手 $\mathcal{A}$ 保密的情况下，G_2^{I} 与 G_1^{I} 具有不可区分性。

G_3^{I}：该游戏与 G_2^{I} 类似，区别仅在于 $\mathcal{C}$ 将 p_F 替换为等长的随机比特串 str_p。由于秘密值 s 对敌手 $\mathcal{A}$ 保密，且 $H(s,\cdot)$ 可被看作随机预言机，因此 $\mathcal{C}$ 可使用随机比特串 str_p 仿真 $H(s,M_b)$。G_3^{I} 与 G_2^{I} 在功能上具有不可区分性。

G_4^{I}：该游戏与 G_3^{I} 类似，区别仅在于 $\mathcal{C}$ 将 G_3^{I} 中的

$C_k = k_r \oplus \mathrm{str}_p$ 替代为等长的随机比特串 str_k。由于密码本加密具有语义安全性，G_4^{I} 与 G_3^{I} 具有不可区分性。

在 G_4^{I} 中，敌手 $\mathcal{A}$ 收到的信息为 $(\mathrm{str}_r, \mathrm{str}_s, \mathrm{str}_k)$，均为均匀随机的值，与 (M_0, M_1) 无关。因此，$\mathcal{A}$ 在猜测阶段只能随机输出 b'，其概率优势是可忽略的，即 $\mathrm{Adv}(\mathcal{A}) = \left|\Pr[b = b'] - \frac{1}{2}\right| \leqslant \varepsilon$。由于 G_4^{I} 与 G_0 具有不可区分性，而 G_0、$\mathcal{A}$ 和 (M_0, M_1) 分别模拟了本文方案、云服务器和不流行数据，因此可认为诚实但好奇的云服务器无法破坏不流行数据的语义安全性，定理 1 得证。

6.3　外部敌手的安全性分析

外部敌手 A_{E} 可通过截获通信数据并发起字典攻击破坏数据机密性；通过篡改数据内容破坏数据完整性；发起密钥和文件伪造攻击破坏数据完整性；使用部分数据内容进行所有权欺骗攻击试图获取数据所有权或篡改数据流行度，本节针对上述攻击进行安全性分析。

1) 字典攻击。不流行数据均为随机加密后的密文，基于对称加密的语义安全性，A_{E} 无法进行离线字典攻击。对于不可预测的流行数据，基于 CE 提供的收敛安全性，A_{E} 无法运行离线字典攻击。由于用户为每个文件设置了进行 sPAKE 协议次数的上限 N_P，A_{E} 无法通过频繁运行 sPAKE 协议进行在线字典攻击。

2) 数据篡改攻击。A_{E} 可在用户下载数据时截获通信数据并进行篡改。用户下载数据后可通过验证 $H(F)$ 与 H_F 是否相等来检测数据完整性是否被损坏。若检测到完整性损坏，用户可重新下载数据，并删除已被篡改的数据。

3) 密钥伪造攻击。在不流行数据上传过程中，若 A_{E} 为 sPAKE 协议中的检查者，则使用伪造的密钥 k_F' 进行密钥传递发起密钥伪造攻击，试图破坏其他用户的数据完整性。在密钥验证过程中，数据上传者发送密文哈希值 H_C 至云服务器。基于哈希函数的抗碰撞性，云服务器可通过比对哈希值识别出密钥伪造攻击，并中止数据去重。

4) 文件伪造攻击。在流行度转换过程中，A_{E} 可通过上传正确的文件哈希值 H_F 和伪造的收敛密文 C_F^* 发起文件伪造攻击，试图破坏数据完整性。在密文验证的过程中，云服务器计算哈希值 $H(C_F)$ 并发送至检查者 U_j，U_j 验证 $H(C_F)$ 是否与本地存储的 H_{CE} 相等，并将比较结果返回给云服务器。基于哈希函数的抗碰撞性，云服务器可在密文验证过程中检测出文件伪造攻击，通过中止数据去重防止数据完整性被破坏。

5) 所有权欺骗攻击。在流行数据上传的过程中，A_{E} 需要与云服务器进行所有权证明，基于文献[9]中的安全分析，仅拥有部分数据内容的 A_{E} 通过所有权证明的概率是可忽略的。在不流行数据上传的过程中，A_{E} 需要拥有完整的数据内容才能计算出 $H(F,s)$ 以恢复密钥 k_r。基于哈希函数的雪崩效应，仅拥有部分数据内容的 A_{E} 无法恢复 k_r，从而无法进行所有权欺骗攻击。

综上，外部敌手无法通过上述攻击方式破坏数据完整性和机密性。

6.4　与现有方案的安全性对比

表 3 将本文方案与现有基于流行度的加密去重方案的安全性进行了对比。与文献[6-7]相比，本文方案引入了所有权证明和完整性验证，可有效对抗所有权欺骗攻击并可提供数据完整性。与文献[8]相比，本文在不流行数据上传的过程中设计了密钥传递过程，可实现不流行数据的加密去重以节省存储空间（表 3 中的数据去重指同时实现流行数据和不流行数据的去重）。与文献[9]相比，本文使用短哈希值进行流行度检测，可为不流行数据提供语义安全。另外，本文方案最关键的特性是在不需要部署任何第三方服务器的情况下，实现了数据流行度的安全统计以及不流行数据的加密去重，这使本文方案不存在性能瓶颈和单点故障等缺陷。

表 3　　安全性对比

方案	语义安全	数据完整性	所有权证明	抗伪造攻击	数据去重
文献[6]	√	×	×	√	×
文献[7]	√	×	×	√	×
文献[8]	×	√	√	√	√
文献[9]	√	√	√	√	×
本文方案	√	√	√	√	√

7　性能分析

本节分别从算法复杂度和实验评估两方面分析方案的性能。

7.1　算法复杂度

表 4 分析了不流行数据首次/后续上传和流行度转换 3 种不同情况下的客户端计算开销，鉴于流行数据上传的过程较为简单，且本文方案与现有方案[6,8-9]的开销几乎相同，因此这里不做分析。为方便表述，方案中哈希值、加密密钥和随机值的大小

表 4 客户端计算开销

方案	不流行数据首次上传	不流行数据后续上传	流行度转换
文献[6]	$(2\mathrm{SE}+H)O(l_F)+\mathrm{TE}O(\lambda)$	$(2\mathrm{SE}+H)O(l_F)+\mathrm{TE}O(\lambda)$	$(2\mathrm{SE}+H)O(l_F)+\mathrm{TE}O(\lambda)$
文献[8]	$(2\mathrm{SE}+H+\mathrm{PoW})O(l_F)+\mathrm{HE}O(\lambda)$	$(2\mathrm{SE}+H+\mathrm{PoW})O(l_F)+\mathrm{HE}O(\lambda)$	$(2\mathrm{SE}+H+\mathrm{PoW})O(l_F)+\mathrm{HE}O(\lambda)$
文献[9]	$(2\mathrm{SE}+H)O(l_F)+\mathrm{SS}O(\lambda)$	$(2\mathrm{SE}+H)O(l_F)+\mathrm{SS}O(\lambda)$	$(2\mathrm{SE}+H)O(l_F)+\mathrm{SS}O(\lambda)$
本文方案	$(2\mathrm{SE}+H+\mathrm{PoW})O(l_F)$+sPAKE	$(2\mathrm{SE}+2H+\mathrm{PoW})O(l_F)$+$\mathrm{SK}O(l_\lambda)$+sPAKE	$(2\mathrm{SE}+2H+\mathrm{PoW})O(l_F)$+$\mathrm{SK}O(\lambda)$+sPAKE+PT

统一用 λ 表示，l_F 和 l_λ 分别表示外包数据大小和随机值 s 的 RSA 密文大小；SE、H、TE、HE、SS、PoW、sPAKE、SK、PT 分别表示对称加密、哈希函数、门限加密[6]、同态加密[8]、秘密共享[9]、所有权证明[8,13]、sPAKE[11]、RSA 解密和生成辅助信息 pt 的计算开销。由表 4 可以看出，现有方案在不同情况下的计算开销类似，而本文方案在 3 种不同情况下的计算开销则略有不同。由于实现了不流行数据的客户端去重，本文方案中流行度转换和不流行数据后续上传的计算开销略高于不流行数据首次上传。

另外，与现有方案[6,8-9]相比，本文方案数据上传的计算开销差距不大，都主要集中在对外包数据的哈希和对称加密上；本文方案增加的计算开销主要在于 sPAKE 协议的执行和密文/密钥的验证过程。由 7.2.1 节的实验结果可知，sPAKE 协议对方案的性能影响不大。在不流行数据后续上传中，由于引入了密钥验证的过程，本文方案与现有方案[6,8-9]相比增加了一次数据哈希的操作，这增加了一定的计算开销。本文方案引入的计算开销主要用于消除系统对第三方服务器的依赖，提高方案在真实场景中的应用价值。

7.2 实验评估

本节通过仿真实验对本文方案进行了性能评估。实验使用戴尔 Inspiron 5680 台式计算机模拟实现了方案中的云服务器和用户，其配备有 3.20 GHz Intel(R) Core(TM) i7-8700 六核处理器和 8 GB 运行内存，运行操作系统为 64 bit Ubuntu 20.04.1。对于加密去重系统来说，存储效率和加密上传的时间开销是 2 个关键的性能指标。本文方案同时实现了流行数据和不流行数据的去重，达到了最佳的去重效率。因此，本节主要针对方案中数据上传的时间开销进行分析。本节中的所有实验结果均为 50 次以上实验结果的平均值。实验中短哈希值位数为 13，哈希函数使用 SHA-256 算法，对称加密使用 AES，密钥长度为 256 位。

7.2.1 Merkle Puzzles 和 sPAKE 的性能分析

本文方案的数据上传过程中，Merkle Puzzles 和 sPAKE 的时间开销会随着系统中的数据量和用户量而变化。本节首先测试了不同文件数量下 Merkle Puzzles 的时间开销，如图 5 所示。从图 5 可以看出即使文件数量为 1 024 个，Merkle Puzzles 的时间开销也仅为 210 μs 左右。由于 Merkle Puzzles 仅需对长度较短的随机值进行对称加解密和哈希操作，这部分的时间开销极小，在数据加密上传的整体时间开销中占比极低，几乎可忽略不计。

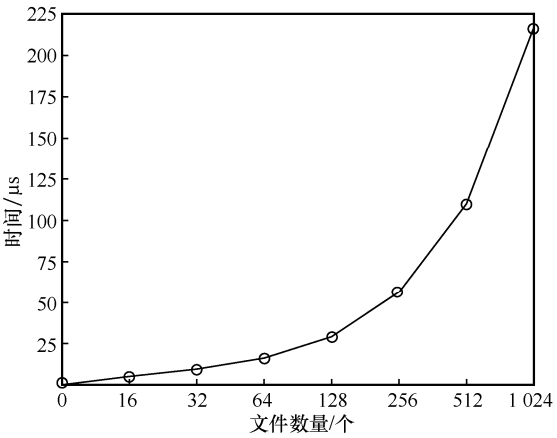

图 5 不同文件数量下 Merkle Puzzles 的时间开销

此外，本节测试了执行不同次数 sPAKE 的时间开销，如图 6 所示。从图 6 可以看出，相比于 Merkle Puzzles，sPAKE 的时间开销明显更大。当用户执行 160 次 sPAKE 协议时，时间开销约为 4.2 s。虽然本文使用了目前已知最高效的 sPAKE 协议[11]，但 sPAKE 协议的参与方仍需执行 2 次幂运算。当 sPAKE 协议的执行次数较多时，仍会产生一定的时间开销。但是，本文方案为防止敌手的在线字典攻击，需要对执行 sPAKE 协议的次数进行限制。因此，系统中执行 sPAKE 协议的时间开销的上限是固定的。当系统设定参与 sPAKE 协议的数量上限为 80 时，执行 sPAKE 协议的时间开销的上限约为 2.1 s。总体而言，由于次数限制，执行 sPAKE 协议的时间开销在整体数据加密上传的过程中占比不大，这一点可以通过 7.2.2 节中的实验分析看出。

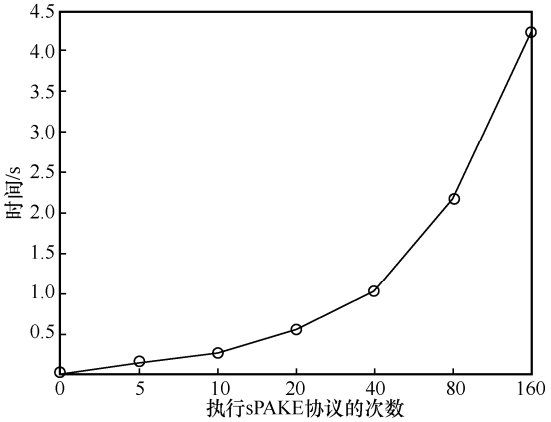

图 6　执行不同次数 sPAKE 协议的时间开销

7.2.2　数据加密上传的时间开销

本文方案中的数据加密上传可分为 4 种情况：不流行数据首次上传，不流行数据后续上传、流行度转换和流行数据上传。7.2.2 节和 7.2.3 节的实验中均分别将 Merkle Puzzles 使用的文件数量和执行 sPAKE 协议的次数设定为 20 个和 5 次。

首先，本节使用不同大小的测试文件对这 4 种情况下的数据加密上传的时间开销进行了评估，如图 7 所示。从图 7 可以看出，当数据变得流行后，数据加密上传的时间开销明显降低。当测试文件的大小为 1 024 MB 时，流行数据上传的时间开销仅为不流行数据首次上传的 42.6%。流行度转换的时间开销较高，原因在于流行度转换过程中需要生成随机密文以进行密钥验证和密文验证，并且需要加密上传收敛密文至云服务器。但是在数据上传过程中，系统大概率仅需进行一次流行度转换过程。因此，这一过程的时间开销对大多数用户的影响较小。

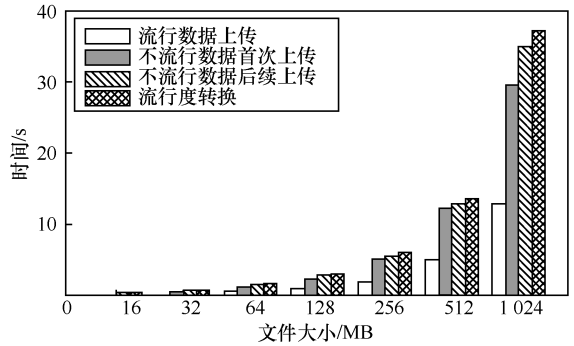

图 7　不同文件大小对数据加密上传的时间开销的影响

本节还统计了数据加密上传的时间开销中的各部分占比情况，如图 8 所示，其中，P、UP_F、UP_S 和 Conv 分别表示流行数据上传、不流行数据首次上传、不流行数据后续上传和流行度转换 4 种情况。由图 8 可知，对于流行数据上传而言，收敛加密占比极高，约占 81%。在其他 3 种情况中，数据加密（收敛加密+随机加密）的时间开销分别占总开销的 66%、50%和 49%。在不流行数据后续上传和流行度转换过程中，用户需要在密钥验证过程中计算随机密文的哈希值，这部分的时间开销占比较高，约占总开销的 20%。方案中引入的所有权证明和 sPAKE 协议的时间开销相对于其他部分占比较低，在不流行数据后续上传和流行度转换的过程中占比均约为 9%。

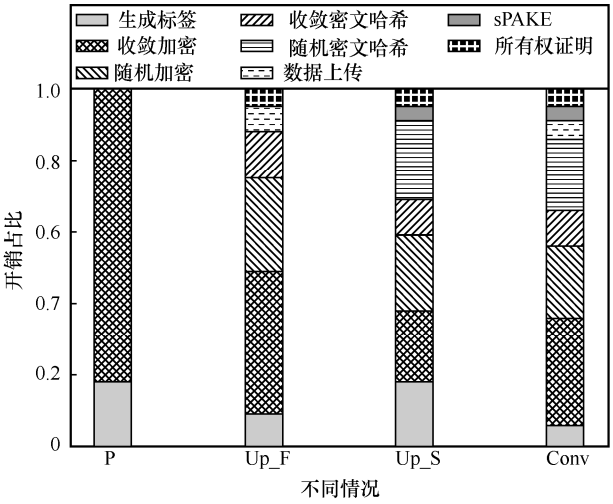

图 8　数据加密上传的时间开销中各部分的占比情况

本节还评估了执行不同次数 sPAKE 协议对数据加密上传的时间开销的影响，如图 9 所示。从图 9 可以看出，不同次数 sPAKE 协议的执行对不流行数据后续上传的时间开销的影响有限。当测试文件的大小为 1 024 MB，执行 5 次 sPAKE 协议时，不流行数据后续上传的时间开销约为执行 160 次 sPAKE 协议时的时间开销的 90%，二者差距不大。

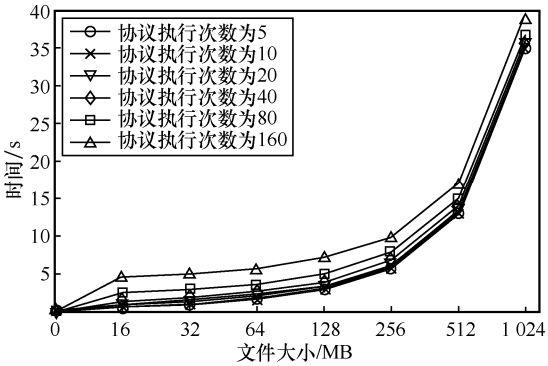

图 9　执行不同次数 sPAKE 协议对数据加密上传的时间开销的影响

7.2.3　与现有方案的性能对比

本节对比分析了本文方案与文献[6,8]方案的不

流行数据加密上传的时间开销，如图 10 所示。由于本文方案未部署第三方服务器，引入了 sPAKE、Merkle Puzzles 以及密钥/密文验证等过程，因此与现有方案相比增加了一些计算开销。本文方案的不流行数据上传分为首次上传和后续上传，文献[6,8]方案中不流行数据上传不区分首次上传和后续上传。本文方案的首次上传过程并未引入过多的计算开销，但后续上传过程中需要进行密钥传递和密钥验证的过程，与现有方案相比增加了一定的计算开销。当测试文件大小为 1 024 MB 时，与现有方案相比，本文方案的不流行数据首次上传和后续上传分别增加了约 1%和 13%的时间开销。

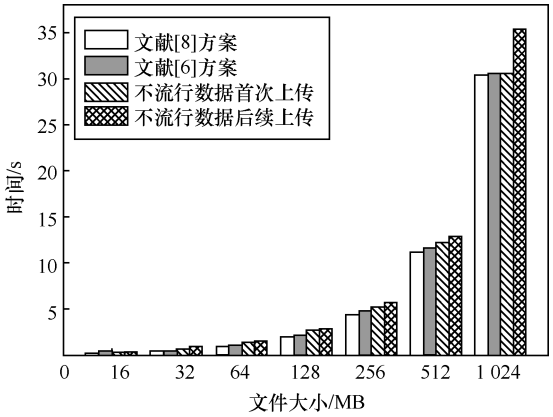

图 10　不同方案的不流行数据加密上传的时间开销

8　结束语

本文提出了一个无第三方服务器的基于数据流行度的加密去重方案，在不需要部署任何第三方服务器的情况下实现了数据流行度的精确统计，并为不流行数据提供了语义安全和加密去重。首先，基于 CM-sketch 算法实现了数据流行度的初步统计；然后，基于 Merkle Puzzles 协议实现了数据流行度的精确统计；最后，通过用户间执行 sPAKE 协议实现了不流行数据的客户端加密去重。本文还考虑了加密去重场景下常见的字典攻击、文件伪造攻击和所有权欺骗攻击等，在所提方案中设计了密钥验证和密文验证的过程，并结合了速率限制和所有权证明等过程为用户数据提供了全面的安全保障。如何进一步提高方案的数据加密上传的性能是接下来需要研究的重要课题。

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] 熊金波, 张媛媛, 李凤华, 等. 云环境中数据安全去重研究进展[J]. 通信学报, 2016, 37(11): 169-180.
XIONG J B, ZHANG Y Y, LI F H, et al. Research progress on secure data deduplication in cloud[J]. Journal on Communications, 2016, 37(11): 169-180.

[3] SHIN Y, KOO D, HUR J. A survey of secure data deduplication schemes for cloud storage systems[J]. ACM Computing Surveys, 2017, 49(4): 1-38.

[4] DOUCEUR J R, ADYA A, BOLOSKY W J, et al. Reclaiming space from duplicate files in a serverless distributed file system[C]// Proceedings of the 22nd International Conference on Distributed Computing Systems. Piscataway: IEEE Press, 2002: 617-624.

[5] BELLARE M, KEELVEEDHI S, RISTENPART T. DupLESS: server-aided encryption for deduplicated storage[C]//Proceedings of the 22nd USENIX Conference on Security. Berkeley: USENIX Association, 2013: 179-194.

[6] STANEK J, KENCL L. Enhanced secure thresholded data deduplication scheme for cloud storage[J]. IEEE Transactions on Dependable and Secure Computing, 2018, 15(4): 694-707.

[7] PUZIO P, MOLVA R, ÖNEN M, et al. PerfectDedup: secure data deduplication[C]//Proceedings of the 10th International Workshop on Data Privacy Management. Berlin: Springer, 2015: 150-166.

[8] HA G X, CHEN H, JIA C F, et al. A secure deduplication scheme based on data popularity with fully random tags[C]//Proceedings of 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). Piscataway: IEEE Press, 2021: 207-214.

[9] 高文静, 咸鹤群, 程润辉. 基于双层加密和密钥共享的云数据去重方法[J]. 计算机学报, 2021, 44(11): 2203-2215.
GAO W J, XIAN H Q, CHENG R H. A cloud data deduplication method based on double-layered encryption and key sharing[J]. Chinese Journal of Computers, 2021, 44(11): 2203-2215.

[10] CORMODE G, MUTHUKRISHNAN S. An improved data stream summary: the Count-Min sketch and its applications[J]. Journal of Algorithms, 2005, 55(1): 58-75.

[11] MCQUOID I, ROSULEK M, ROY L. Minimal symmetric PAKE and 1-out-of-N OT from programmable-once public functions[C]//Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM Press, 2020: 425-442.

[12] MERKLE R C. Secure communications over insecure channels[J]. Communications of the ACM, 1978, 21(4): 294-299.

[13] XU J, CHANG E C, ZHOU J Y. Weak leakage-resilient client-side deduplication of encrypted data in cloud storage[C]//Proceedings of the 8th ACM SIGSAC symposium on Information, Computer and Communications Security. New York: ACM Press, 2013: 195-206.

[14] HALEVI S, HARNIK D, PINKAS B, et al. Proofs of ownership in remote storage systems[C]//Proceedings of the 18th ACM Conference on Computer and Communications Security. New York: ACM Press, 2011: 491-500.

[15] BELLARE M, KEELVEEDHI S, RISTENPART T. Message-locked encryption and secure deduplication[C]//Advances in Cryptology - EUROCRYPT 2013. Berlin: Springer, 2013: 296-312.

[16] 熊金波, 李素萍, 张媛媛,等. 共享所有权证明: 协作云数据安全去重新方法[J]. 通信学报, 2017, 38 (7): 18-27.

XIONG J B, LI S P, ZHANG Y Y, et al. PoSW: novel secure deduplication scheme for collaborative cloud applications[J]. Journal on Communications, 2017, 38(7): 18-27.

[17] XU R H, JOSHI J, KRISHNAMURTHY P. An integrated privacy preserving attribute-based access control framework supporting secure deduplication[J]. IEEE Transactions on Dependable and Secure Computing, 2021, 18(2): 706-721.

[18] 贾春福, 哈冠雄, 李瑞琪. 密文去重系统中的数据访问控制策略[J]. 通信学报, 2020, 41(5): 72-83.
JIA C F, HA G X, LI R Q. Data access control policy of encrypted deduplication system[J]. Journal on Communications, 2020, 41(5): 72-83.

[19] 贾春福, 哈冠雄, 武少强, 等. 加密去重场景下基于 AONT 和 NTRU 的密钥更新方案[J]. 通信学报, 2021, 42(10): 67-80.
JIA C F, HA G X, WU S Q, et al. AONT-and-NTRU-based rekeying scheme for encrypted deduplication[J]. Journal on Communications, 2021, 42(10): 67-80.

[20] LI J W, LEE P P C, TAN C F, et al. Information leakage in encrypted deduplication via frequency analysis[J]. ACM Transactions on Storage, 2020, 16(1): 1-30.

[21] LI J W, YANG Z R, REN Y J, et al. Balancing storage efficiency and data confidentiality with tunable encrypted deduplication[C]//Proceedings of the Fifteenth European Conference on Computer Systems. New York: ACM Press, 2020: 1-15.

[22] HARNIK D, PINKAS B, SHULMAN-PELEG A. Side channels in cloud services: deduplication in cloud storage[J]. IEEE Security & Privacy, 2010, 8(6): 40-47.

[23] ZHANG Y, MAO Y L, XU M Z, et al. Towards thwarting template side-channel attacks in secure cloud deduplications[J]. IEEE Transactions on Dependable and Secure Computing, 2021, 18(3): 1008-1018.

[24] YU C M, GOCHHAYAT S P, CONTI M, et al. Privacy aware data deduplication for side channel in cloud storage[J]. IEEE Transactions on Cloud Computing, 2020, 8(2): 597-609.

[25] ABDALLA M, POINTCHEVAL D. Simple password-based encrypted key exchange protocols[C]//The Cryptographers' Track at the RSA Conference. Berlin: Springer, 2005: 191-208.

[26] YU C M. POSTER: efficient cross-user chunk-level client-side data deduplication with symmetrically encrypted two-party interactions[C]//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM Press, 2016: 1763-1765.

[27] LIU J, ASOKAN N, PINKAS B. Secure deduplication of encrypted data without additional independent servers[C]//Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. New York: ACM Press, 2015: 874-885.

[作者简介]

哈冠雄（1995− ），男，回族，天津人，南开大学博士生，主要研究方向为云数据安全、密码学应用、加密数据去重。

贾巧雯（1992− ），女，天津人，中国科学院软件研究所博士生，主要研究方向为并行编程和软件验证、计算机系统安全。

陈杭（1998− ），女，天津人，南开大学硕士生，主要研究方向为密码学应用、加密去重。

贾春福（1967− ），男，河北文安人，博士，南开大学教授、博士生导师，主要研究方向为网络与信息安全、可信计算、恶意代码分析、密码学及应用等。