电子科技大学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

本科学位论文

BACHELOR THESIS

论文题目 密文重复数据删除机制的频率分析攻击

学科专业	信息安全
学　　号	2015040101018
作者姓名	任彦璟
指导老师	李经纬　副教授

分类号 _____ 密级 _____

UDC 注 1 _____

学　位　论　文

密文重复数据删除机制的频率分析攻击

（题名和副题名）

任彦璟

（作者姓名）

指导老师 李经纬　副教授

（姓名、职称、单位名称）

申请学位级别 本科 学科专业 信息安全

提交论文日期 _____ 论文答辩日期 _____

学位授予单位和日期 电子科技大学

答辩委员会主席 _____

评阅人 _____

注 1：注明《国际十进分类法 UDC》的类号。

摘　要

加密重复数据删除旨在解决大规模数据存储系统中的安全性和存储效率：它确保每个明文都由明文本身内容派生的对称密钥加密，从而为数据存储提供机密性保证，同时保留重复数据删除的存储节省效率。但是，加密重复数据删除的确定性特性也会泄漏明文的频率，从而使加密重复数据删除容易受到频率分析的影响。在本文中，重新审视了频率分析导致的加密重复数据删除的安全漏洞。本文认为加密重复数据删除可能更容易受到精心设计的频率分析攻击，因此攻击提供了高可信度，可确保每个推理的明文确实对应于目标密文的概率很高（即从统计角度来看，高精度）。为此，本文提出了两种新的频率分析攻击，它们可以适应实际重复数据删除工作负载的特性，从而提高频率分析的准确性。本文根据实际情况评估针对实际存储工作负载的建议攻击，并提供有关如何带来实际损失的观察。

关键词： 加密重复数据删除, 频率分析攻击, 聚类分析

ABSTRACT

Encrypted deduplication aims to address security and storage efficiency in large-scale data storage systems: it ensures that each plaintext is encrypted by a symmetric key that is derived from the content of the plaintext itself, so as to provide confidentiality guarantees for data storage while preserving the storage saving effectiveness of deduplication. However, the deterministic nature of encrypted deduplication also leaks the frequencies of plaintexts, thereby making encrypted deduplication vulnerable to frequency analysis. In this paper, we revisit the security vulnerability of encrypted deduplication due to frequency analysis. We argue that encrypted deduplication can be even more vulnerable to carefully crafted frequency analysis attacks, such that the attacks provide high confidence of ensuring that each inferred plaintext indeed corresponds to the target ciphertext with a high probability (i.e., high precision from a statistical perspective). To this end, we propose two new frequency analysis attacks that adapt the characteristics of practical deduplication workloads to increasing the severity of frequency analysis. We empirically evaluate our proposed attacks against real-world storage workloads, and provide observations on how they bring actual damages.

Keywords: Encrypted deduplication, Frequency analysis attacks, Cluster analysis

目 录

第一章 绪 论

1.1 研究工作的背景与意义

1.1.1 研究背景

随着信息技术产业的高速发展，数字信息量呈爆炸式增长。Gartner 研究表明[1]，仅 2015 年的移动数据流量就较 2014 年增长 59%；并且，这一增长率将持至 2018 年末，移动数据流量水平达 1.73 亿 TB。数据的快速增长导致企业面临的存储和管理成本越来越高[2]。另一方面，在存储系统所保存的数据中，高达 60%的数据都是冗余的，随着时间的推移，这些冗余数据的比例将进一步上升[3]。近年来，存储系统中数据高冗余的特点得到越来越多研究人员的关注，利用该特点来节省存储容量是一个热门研究课题。

数据重删技术（data deduplication）是指通过识别数据流中的冗余，只传输或存储唯一数据（unique data），而使用指向已存储数据的指针替换重复副本，以达到节省带宽或存储空间的目的[4]。由于能够有效地降低存储开销，数据重删技术非常适合为管理日益增长的海量数据节省成本。在工业界，EMC Data Domain[5] 和 Avamar[6]、Veritas 的 NetBackup Appliances[7] 以及 Commvault 的开放数据平台[8]都是比较知名的数据重删应用产品；此外，各大云存储厂商（例如 Dropbox、Google Drive、Bitcasa、Moza 等）也纷纷将数据重删技术应用于各自的云服务产品中，以提升经济效益[9]。

如图1-1所示，在支持数据重删的存储系统（统称为数据重删系统）中，重删后的任何数据块都被一个或多个文件引用，而文件则以指向这些数据块的指针的集合形式存储。这种文件共用数据块的存储模式强调了数据块的敏感性，因为一个数据块的泄漏可能扩散影响到共用这个数据块的所有文件。如何保护重删后的数据的隐私，成为信息安全领域的一个研究热点。

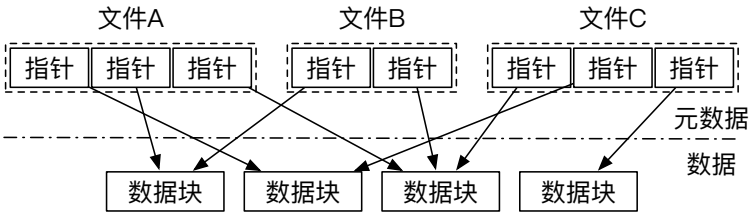

图 1-1 数据重删系统的存储模式

为了保护数据隐私，加密重复数据删除（encrypted deduplication）增加了一层

作用于逻辑数据块的加密操作。如图1-2所示，该加密层基于数据内容来产生加密密钥[10]（例如将数据块的哈希值作为密钥[11]），从而将相同的明文数据块加密为相同的密文数据块。系统计算每个密文数据块的哈希值(称为指纹，fingerprint)，查询指纹索引（fingerprint index）确定该数据块是否已经存储，最后保存仅具有唯一指纹的密文数据块。需要指出的是，部分加密重复数据删除方案[10]采用随机加密算法，但基于明文数据块产生指纹，因此仍然可以通过检查指纹来识别重复数据。

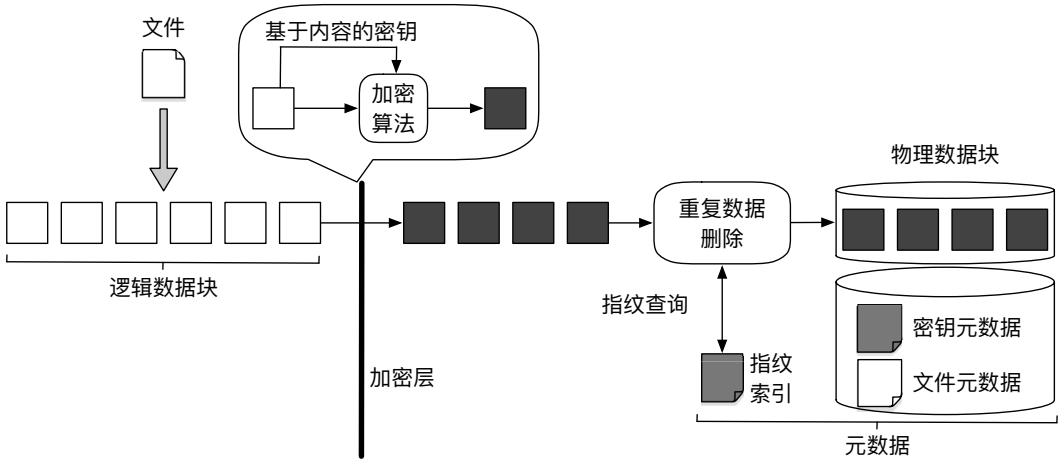

图 1-2 加密重复数据删除系统逻辑视图

除了指纹索引以外，加密重复数据删除系统须存储额外的元数据（metadata），包括:

1. 文件元数据记录了文件内逻辑数据块与相应物理数据块的映射关系，用于重构完整文件。
2. 密钥元数据记录了文件内逻辑数据块的解密密钥，用于恢复相应的明文内容，由于密钥元数据包含密钥信息，需由文件属主的主密钥（master key）加密后以密文形式存储。

1.1.2 问题和动机

数据块频率泄漏问题

由于基于数据块内容产生密钥，加密重复数据删除泄漏了数据块的频率信息，即如果一个明文数据块出现了 n 次，则它对应的密文数据块也将出现 n 次。另一方面，真实数据集中数据块的出现频率往往呈非均匀分布，本文调研了 FSL 和 VM 备份数据集 (数据集信息见 §3.2) 的数据块频率分布特征，发现三种数据集有超过 97% 的数据块的频率低于 100 次，而至多只有 0.04% 的数据块的频率高于 10,000 次（图1-3），这种非均匀的分布特点使攻击者可以利用频率来确定相应数据块。

基于以上原因，认为加密重复数据删除可能受到频率分析[12]的威胁，拟通过

本课题，深入研究频率分析攻击对加密重复数据删除安全性的影响，以及提高频率分析攻击效果的方法。

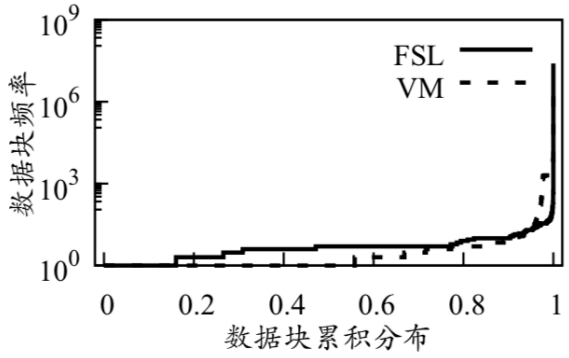

图 1-3 两种真实数据集的数据块频率分布

1.1.3 研究意义

本课题研究将填补频率分析攻击研究空白，对理解加密重复数据删除的实际安全性，并降低其在非适合场景下的误用风险具有重要作用。

尽管加密重复数据删除的频率泄漏已是学术界公认的安全问题，但针对性的频率分析攻击研究仍为空白 (即利用频率泄漏来获取隐私数据仍是一个开放性问题)，致使部分厂商盲目地将加密重复数据删除技术应用于商业产品[13,14] 和开源系统[15–18] 中。本项目将研究加密重复数据删除技术在频率分析攻击下的实际安全性，以指导其在适合场景下被正确使用。

1.2 国内外研究历史与现状

1.2.1 加密重复数据删除

在传统对称加密方式下，每个用户具有不同的密钥，不同用户之间的相同明文会被加密为不同密文，难以执行 (密文) 重复数据删除操作。

消息锁定加密（message-locked encryption，MLE）确立了加密重复数据删除的密码学基础[10]: 基于数据内容产生密钥 (称为 MLE 密钥)，从而将相同明文加密为相同密文。最流行的 MLE 实例是收敛加密（convergent encryption，CE）[11]，它使用明文的哈希值作为 MLE 密钥，并基于密文哈希值计算指纹，以识别重复数据（如图1-2）。基于 CE 的加密重复数据删除方案还包括:

1. 哈希收敛加密（hash convergent encryption，HCE）[11] 与 CE 具有相同的 MLE 密钥产生规则，但基于明文哈希值计算指纹。
2. 随机收敛加密（random convergent encryption，RCE）[11] 使用随机密钥加密

以产生非确定的密文，同时也基于明文哈希值来进行重复检查。

3. 收敛扩散（convergent dispersal，CD）[19] 使用明文哈希值作为秘密共享（secret sharing）的输入种子，在兼容重复数据删除的基础上提高了密文存储的可靠性。

上述 MLE 实例基于明文产生 MLE 密钥（CE、HCE 和 CD）或指纹（HCE 和 RCE），如果明文是可预测的 (即所有可能的明文的数量有限)，这些方案易于受到离线暴力破解攻击[10,20]。为了抵御该攻击，DupLESS[20] 基于第三方密钥服务器实现了服务器辅助 MLE（server-aided MLE），确保无法从离线明文派生出相应的 MLE 密钥。以服务器辅助 MLE 为基础，现有研究进一步解决了加密重复数据删除的故障容错[21,22]、透明价格模型[23]、点对点密钥管理[24]、层次密钥管理[25] 等问题。围绕 MLE 扩展功能的一系列研究包括: 兼容加密重复数据删除的数据完整性审计协议[26]；支持动态访问控制的加密重复数据删除系统 REED[27,28]。

无论是 MLE 还是服务器辅助 MLE，均须为相同的明文产生相同的密文（CE、HCE、CD 和 DupLESS）或指纹（HCE 和 RCE），泄漏了数据的出现频率。一些理论研究[29–31] 基于零知识证明、全同态加密、双线性对运算等底层密码技术实现了随机加密，但这些方案存在计算复杂性高、依赖多轮信息交互等问题，难以应用到系统实践中。本文关注可实际应用的加密重复数据删除系统/方案的频率泄漏问题，研究其安全性影响和防御对策。

1.2.2 频率分析攻击

频率分析[32] 是一种针对确定性加密（deterministic encryption）的密码分析技术，被应用于破解匿名查询日志[33]、破坏关键词隐私[34–38]、重构密文数据库记录[12,39–43] 等实际攻击。本项目研究针对数据块的频率分析攻击，与现有攻击目标（查询日志条目、关键词、数据库记录等）相比，数据块数量极其庞大（呈千万级），并且大量数据块具有相同频率，致使当前的频率分析攻击算法难以适用。

面向加密重复数据删除，已有工作[44] 提出了基于数据块局部性（chunk locality）[45–47] 的频率分析攻击。本课题的部分技术路线（**??**）就是在该工作[44] 的基础上提出来的，并进一步研究频率分析攻击的准确率和依赖条件，以及面向真实系统的攻击原型。除了频率分析和暴力破解（§ 1.2.1）之外，加密重复数据删除还可能遭受边信道攻击[9,48,49]、副本伪造攻击[10]、基于数据块长度的攻击[50] 等威胁，但这些攻击可通过所有权证明[48]、守卫解密 (guarded decryption)[10]、固定长度分块等措施进行防御，而本文所研究的频率分析攻击超出了现有保护措施的防御范畴。

1.3 课题的研究内容、研究目标、以及拟解决的关键问题

1.3.1 研究内容

加密重复数据删除的频率分析攻击

在传统频率分析模式下，攻击者能够访问明文逻辑数据块集合 M 和密文逻辑数据块集合 C（M 和 C 包含重复的明文和密文数据块）。攻击者根据出现频率分别对 M 和 C 中的数据块进行排序，然后将 C 中的密文数据块映射为 M 中与其具有相同排名的明文数据块。但是，传统频率分析在加密重复数据删除中难以形成有效的攻击，主要原因是：M 和 C 的原始内容可能存在差异 (例如 M 和 C 来源于同一个文件系统在不同时间点的备份镜像)，将打乱数据块频率排序的对应关系；并且，在频率排序过程中可能存在大量明文和密文数据块具有相同的频率，频率分析难以排序这些数据块来形成正确的对应关系。

为了提高传统频率分析的攻击效果，首先研究基于数据特征的新型频率分析攻击技术。然后，分别从抵抗频率排序干扰和降低攻击发生条件两方面改进攻击技术。最后，实现针对真实系统的频率分析攻击原型，并分析该攻击对各类数据安全性的影响。

1.3.2 研究目标

针对以上研究内容，预期实现如下研究目标:

1. 在理论上，构造针对加密重复数据删除的频率分析攻击，揭示实践中的安全隐患。
2. 在技术上，以理论研究为支撑，设计并实现针对加密重复数据删除系统/方案的频率分析攻击工具，并在真实系统中进行理论验证和攻击效果测试。

1.3.3 拟解决的关键问题

本课题致力于解决传统频率分析攻击方法在针对加密重复数据删除方案/系统进行攻击时难以解决的如下问题:

1. 明密文数据块的的原始内容可能存在差异 (例如一组明密文集合分别来源于同一个文件系统在不同时间点的备份镜像)，将打乱数据块频率排序的对应关系。
2. 在频率排序过程中可能存在大量明文和密文数据块具有相同的频率，基本频率分析攻击难以排序这些数据块来形成正确的对应关系。

1.4 论文结构及安排

本文的其余部分安排如下：第二章将介绍本文中将使用的概念的背景知识，第三章将介绍本文的威胁模型与假设，第四章将给出本文提出的第一种攻击方法（基于分布的攻击），第五章 将给出本文提出的第二种攻击方法（基于聚类的攻击），第六章将给出本文实验采用的相关数据来源的介绍，第七章将对两种攻击进行实验分析，第八章将给出应对本文提出的攻击方法的一些讨论，第九章将介绍重复数据删除与相应攻击的相关工作，最后第十章对本文进行总结。

第二章 预备知识

在本章中，将介绍普通和加密重复数据删除方法的基础知识，以及加密重复数据删除仍然容易受到攻击的原因。

2.1 重复数据删除

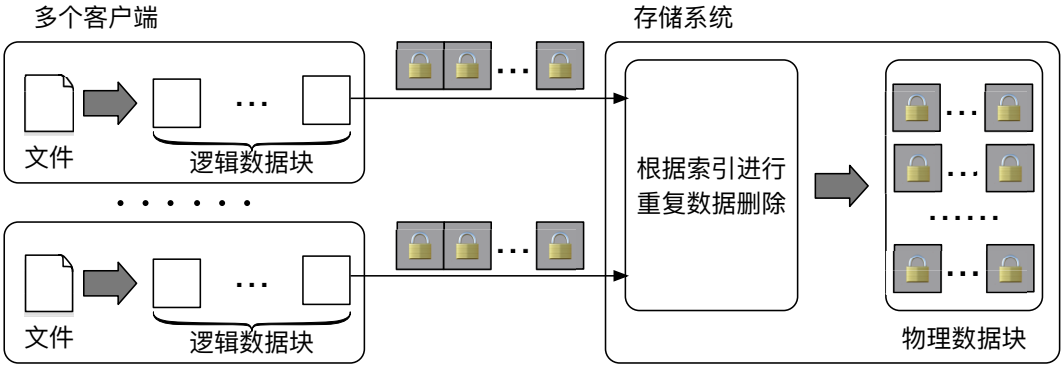

图 2-1 重复数据删除的工作流程概览

本文专注于基于块的重复数据删除，它以称为数据块的小型数据单元为粒度运行。图2-1总结了重复数据删除工作流程。

具体地，重复数据删除系统首先通过文件切块过程将客户端的文件（例如，备份文件）分割成逻辑数据块，根据每一个逻辑数据块的内容使用 HASH 算法（哈希算法）计算得到其对应的唯一标签（又成为指纹）。如果两个数据块具有相同的标签，那么认为两个数据块相同（不同逻辑数据块计算得到相同的标志的概率忽略不计[51]），即数据块内容和其标志完全一致；若两个逻辑数据块标签不一致，显然两逻辑数据块不同。重复数据删除系统的存储系统仅存储相同逻辑数据块的副本（称为物理数据块），并且每个相同的逻辑数据块通过一个较小空间开销的索引引用相同的物理数据块。

基于块的重复数据删除比基于文件的重复数据删除更精细，因此通常具有更高的存储效率（节省更多的存储空间）。基于块的重复数据删除有两种基本的数据块分块方法：固定大小的数据分块方法和可变大小的数据分块方法。固定大小的数据分块方法将数据分割为大小相同的逻辑数据块，而可变大小的数据分块方法（也称为内容定义的数据块分块方法）通常以内容相关的方式指定数据块的边界（例如，通过 Rabin 指纹识别[52]）将文件数据分区为可变大小的数据块。固定大小的分块方法简单而快速，而可变大小的分块方法可以在内容发生变换时保持

存储效率，因为即使文件添加或删除了某些内容，大多数数据块仍保持不变。可变大小的分块通常用在备份系统中（例如，[45,46]），但固定大小的分块显然对某些工作负载更加有效（例如，VM 备份数据集[53]）。本文的工作同时涉及固定大小和可变大小的分块方法产生的数据块。

加密的重复数据删除解决了外包环境（例如，云存储）中的数据块机密性，同时保持了重复数据删除的有效性。出于安全考虑，用户希望在将自己的数据外包之前对其数据进行加密，以确保个人数据的隐私性。传统的对称加密要求多个客户端通过其（不同的）密钥加密其文件数据，即使相同的明文数据块也会被加密为不同的密文数据块，服务器无法感知到这些密文数据块所对应的的原始明文数据块内容是否相同。消息锁定加密（MLE）[10] 给加密重复数据删除提供了新的处理方法。最基本的 MLE 实例化是基于每个明文的内容（例如，基于块的重复数据删除中的逻辑数据块）导出其对称密钥（称为 MLE 密钥），并使用 MLE 密钥加密明文以形成对应的密文（例如，加密的逻辑数据块）。因此，MLE 可以确保将相同的明文加密为相同的密文。最后，存储系统从每个密文中导出其对应的标签并执行重复数据删除。

2.2 重复数据删除中存在的威胁

2.2.1 MLE 加密具有确定性

MLE 可以实现为不同的实例化（参见章节：第九章）。其中最受欢迎的是融合加密（CE）[11]，它已经在各种存储系统中实现[13–17,54–57]。CE 的主要思想是根据每个明文数据块的内容通过 HASH 函数（哈希函数）导出其对应的 MLE 密钥。这种方法可以确保具有相同内容的逻辑数据块加密得到的密文数据块始终一致，即可使用密文数据块的标签进行重复数据删除。但也存在部分 MLE 实例[10] 允许将相同的明文加密成不同的密文。当然，他们确保用于重复数据删除的标签仍然来自明文，因此他们依然可以通过检查标签的相同性来执行重复数据删除。

现有的 MLE 实例都建立在确定性加密这一概念的基础之上，确定性加密可以确保密文（或标签）从明文确定性地导出，与传统的对称加密相比，可以保留重复数据删除的有效性。虽然确定性加密提供了机密性保证，但是本文认为攻击者可以利用这种确定性来推理给定密文的原始明文。

2.2.2 理论上加密重复数据删除存在信息泄漏

实际的加密重复数据删除系统会暴露一些关于原始块的信息，称为泄漏。最基本的泄漏是数据块的频率，这一点已在先前的工作中得到承认[29,31,44,50]。具体

地说，MLE 会将相同的明文数据块加密成相同的密文数据块，这一过程泄漏了每个原始明文数据块出现在输入数据中的次数。本文认为频率泄漏难以避免，因为这会破坏明文和密文数据块的一对一映射关系，进而降低重复数据删除系统的性能[29]或存储效率[44]。

已有的研究工作已经探索了实际加密重复数据删除系统存在的的其他类型的泄漏[44,50]。首先，一些重复数据删除系统[45,47]为了实现更高的性能，需要按照数据块在原始文件中出现的顺序来操作数据块；这暴露了铭文数据块的逻辑顺序，基于该逻辑顺序可以识别输入文件中每个数据块的位置。其次，一些工作[11,20,58]为了降低存储开销，不会填充任何冗余的密文数据块，这暴露了原始明文数据块的数量信息，因为密文数据块应该与原始明文数据块具有相同的数量。

在已有工作[44,50]中，通过构建攻击原语来推理原始明文数据块[44]或根据特定文件的存在[50]来攻击加密重复数据删除方案。但现有的其他关于加密重复数据删除的工作仍未探索过如何基于推理得到的信息发起实际攻击。本文对加密重复数据删除的泄漏问题进行了全面研究，以求回答以下两个问题:

1. 如何在各种泄漏中通过启用基元来推理原始内容?
2. 以推理内容为基础的实际攻击的具体含义是什么?

2.2.3 观察得到的实际系统中的泄漏

下面将讨论在最先进的加密重复数据删除方案/系统中观察到的各种泄漏。

- **频率泄漏**

  加密的重复数据删除方案应用了确定性加密（例如，MLE），其中相同的明文数据块将被加密成相同的密文数据块。这泄漏了每个数据块的频率信息（即原始数据块出现在输入数据中的次数）。频率泄漏是 MLE 难以解决的问题，因为它很难被预防（否则，将导致重复数据删除性能显着下降[29]或存储空间开销大幅提高[44]）。

- **数据块长度信息泄漏**

  为了减轻存储开销，建议从业者在没有填充方案[50]的情况下实现块加密。例如，Farsite[11]，Tahoe-LAFS[58]和 DupLESS[20]在计数器模式下使用 AES 加密，但它们不会使用任何其他数据填充密文数据块。因为密文数据块应该与原始明文数据块具有相同的大小，这会泄漏原始数据块大小信息[50]。

  对于可变大小的数据块，这种泄漏更加严重，由于数据块边界通过匹配特定内容的方法来识别（例如，通过 Rabin 指纹识别[52]）。具有不同大小的两个密文数据块进一步暴露了它们来自于不同明文数据块的信息。

- **数据块逻辑顺序泄漏**

  在部署加密重复数据删除时无意中发生泄漏，使得数据块逻辑顺序被外部所获的。

2.3 频率分析攻击

频率分析已在一些已有的工作[12,34,35,35–40,42,43]中用来构建攻击。例如针对数据库中的结构化关系数据的攻击，通过经典频率分析[12]，探索表列的相关性或排序。

本文旨在应用频率分析来探索加密重复数据删除的漏洞。我们的重点不是解决如何获得用于推理攻击的相关明文。在统频率分析模式下，攻击者能够访问明文逻辑数据块集合 M 和密文逻辑数据块集合 C（M 和 C 包含重复的明文和密文数据块）。攻击者根据出现频率分别对 M 和 C 中的数据块进行排序，然后将 C 中的密文数据块映射为 M 中与其具有相同排名的明文数据块。

2.4 本章小结

本章介绍了普通与加密重复数据删除，现有加密重复数据删除中存在的信息泄漏，以及频率分析攻击的典型应用和其在加密重复数据删除中的基本运用方式。

第三章 威胁模型

本章为针对加密重复数据删除的频率分析攻击制定了威胁模型。

3.1 威胁模型定义

本文分别使用 M 和 C 来表示明文数据块（即加密前的逻辑数据块）及其对应的密文数据块（即加密后的逻辑数据块）；$|M|$ 和 $|C|$ 来表示明文数据块（即加密前的逻辑数据块）的大小及其对应的密文数据块（即加密后的逻辑数据块）的大小。

首先将普通文件建模为由 n 个逻辑明文数据块构成的有序列表（即重复数据删除之前的逻辑块），该列表记为 $\mathbf{M} = \langle \hat{M}^{(1)}, \hat{M}^{(2)}, \ldots, \hat{M}^{(n)} \rangle$。每个逻辑明文数据块 $\hat{M}^{(i)}$（其中 $1 \leq i \leq n$）通过 MLE 加密得到相应的密文数据块 $\hat{C}^{(i)}$。由 $\mathbf{M}$ 的所有加密结果形成由 n 个逻辑密文数据块组成的有序列表记为 $\mathbf{C} = \langle \hat{C}^{(1)}, \hat{C}^{(2)}, \ldots, \hat{C}^{(n)} \rangle$。在 $\mathbf{C}$ 中的逻辑密文数据块也会被排序以反映加密重复数据删除存储系统中重复数据删除处理过程的顺序。

相同的逻辑明文和密文可能分别出现在 $\mathbf{M}$ 和 $\mathbf{C}$ 中的不同位置。本文将一个唯一的明文数据块表示为 M（通过其指纹的唯一性确定），通过 MLE 加密得到相应的唯一密文数据块 C。每个 M 对应于 $\hat{M}^{(i)}$ 的一个或多个相同副本，同理，每个 C 对应于 $\hat{C}^{(i)}$ 的一个或多个相同副本。

3.2 对手的目标和相关假设

本文考虑存在一个对手，它准备以以下两个明确的目标为基准推理一组密文数据块-明文数据块对（用 $\{(C, M)\}$ 表示）。

- **推理率高:**
  在所有的正确密文数据块-明文数据块对中，推理出大部分正确的密文数据块-明文数据块对（即统计学上的高召回率或低阴性率）。
- **推理精度高:**
  在所有推理得到的密文数据块-明文数据块对中大部分的密文数据块-明文数据块对的配对是正确的（即统计术语中的高精度或低误报率）。

本文假设该对手是诚实但好奇的，它可以被动地监视密文数据块流 $\mathbf{C}$ 被写入存储系统中的过程，并利用来自 $\mathbf{C}$ 的不同类型的泄漏信息（参见章节：3.3）。鉴于可用的泄漏信息种类，对手仅可通过唯密文攻击的方法推理 $\mathbf{C}$ 中每个密文数据块对应的原始明文数据块。当然，对手也有可能知道有限的密文数据块-明文数据

块对的集合来发起已知明文攻击，这种情况进一步加深了攻击的严重性[44]。但在本文中不会考虑前述的已知明文攻击方式。因此，本文将 **C** 视为对手所能观察到的的视图，并为该视图的不同属性建模。

本文假设对手无法访问任何包含有关如何操作和存储数据块的信息的元数据（由于不对元数据应用重复数据删除，因此可以通过传统的对称加密来保护这些元数据，这种操作可使得对手无法获知该类型的信息）。此外，本文假设对手没有主动对重复数据删除系统进行攻击的能力，因为这种问题可以通过现有方法予以阻止。例如是恶意客户端可以在客户端重复数据删除中声明对未授权文件拥有所有权[9,48,49]；该问题可以通过所有权证明[48,59,60]或服务器端重复数据删除[9,21]予以解决。另一个例子是恶意存储系统可以修其改存储的数据；该问题可以通过远程完整性检查来进行检测和解决[61,62]。

3.3 三种信息的泄漏

本文在加密重复数据删除存储中考虑三种类型的泄漏方式，这些泄漏使得对手能够基于它们推理信息：

- **频率：**

  由于加密重复数据删除的加密过程的确定性，重复数据删除前 **C** 中每个密文数据块的频率（即重复副本数量）可以映射得到 **M** 中相应明文数据块的频率。

- **顺序：**

  某些加密重复数据删除存储系统[45–47]为了更高的运行性能，在存储密文数据块是保持了其对应的明文数据块的原始顺序。因此，**C** 中的密文的顺序可以映射到 **M** 中的明文的顺序。

- **大小：**

  可变大小的数据块分块方法（参见章节：第二章）产生会产生各种不同大小的明文数据块，如果在对明文数据块进行加密前没有进行数据填充（为了避免增大存储开销[11,20,50,58]），则产生的密文数据块大小与其对应的原始明文数据块大小一致（**C** 中的密文大小可以映射到 **M** 中相应明文的大小）且密文数据块集合中的密文数据块也具有众多不同的大小。当然，该泄漏可以通过使用块密码算法使得明密文数据块大小不一致来避免。

除上述三种泄漏以外，对手还可以获得一些辅助信息，这些信息提供了与 **M** 相关的数据特征的基本情况（任何推理攻击都必须提供辅助信息[12,33–36,38,42,44,50]）。在这项工作中，将辅助信息视为先前已知的明文数据块（例如，通过旧用户备份或

VM 磁盘映像得到）构成的有序列表，由 **A** 表示。显然，攻击有效性与严重性取决于 **A**（即先前已知的明文数据块）和 **M**（即要推理的明文数据块）之间的相关性。本文的重点不是解决对手如何获取辅助信息的问题。例如，可能是由于粗心的数据发布[63]、被的盗存储设备[64]和云存储泄漏[65]。相反，根据这些信息，本文研究了可以获得的的辅助信息如何与各种泄漏渠道相结合，给加密重复数据删除带来信息泄漏。

本文将频率分析[66]作为攻击方法。经典频率分析对 **C** 中的唯一密文数据块和 **A** 中的唯一明文数据块按频率（即对应于每个唯一密文数据块或唯一明文数据块的相同副本的数量）进行排序。然后，简单的将 **C** 中的每个唯一密文数据块与 **A** 中具有相同频率等级的唯一明文数据块相关联。基本的频率分析攻击实际效果非常糟糕，在接下来的章节中，本文针对加密重复数据删除设计了较为复杂同时效性极高的频率分析攻击。

3.4 本章小结

本章介绍了针对重复数据删除的威胁模型的定义、本课题研究中将要用到的各种信息来源，以及本文攻击方案的假设的相关说明。

第四章 基于分布的频率分析攻击方法

基于分布的攻击利用 **C** 和 **A** 的数据块的逻辑顺序信息来增强频率分析的有效性。它通过比较 **C** 和 **A** 的相对频率分布，以减少误报结果。本文还展示了如何利用数据块的大小信息来进一步提高推理精度。

4.1 背景知识：基于数据块局部性的频率分析的攻击

基于分布的频率分析攻击建立在已有的基于数据块局部性的频率分析攻击[44]的基础上，它展示了频率分析如何导致备份工作负载中的信息发生泄漏。基于数据块局部性的频率分析攻击针对每日或每周为主要数据的完整副本（例如，应用程序状态，文件系统快照和 VM 磁盘映像）创建的完全备份（简称为备份）。它旨在推理不同版本备份间的密文数据块-明文数据块对。它通过假定辅助信息 **A** 来自一些较旧的备份，来推理最新备份的明文数据块（即 **M**）。

基于数据块局部性的频率分析攻击利用了数据块局部性这一属性[45–47]，这是实际备份工作负载中的常见现象。具体而言，局部性是指在重复数据删除之前，相邻的数据块在不同版本的备份中往往以相同的逻辑顺序存在。主要原因是每个备份的更新通常聚集在一些小范围内的数据块中，而剩余的大范围内的数据块在不同版本的备份中保持不变（保持相同的顺序）。

根据局部性原理，基于数据块局部性的频率分析攻击利用逻辑顺序信息来发现密文数据块和明文数据块的邻近信息。具体来说，对于给定的唯一密文数据块 C，对手首先标识所有相同副本的集合 $\{\hat{C}^{(i)}\}$。对于每个 $\hat{C}^{(i)}$，它会考虑 $\hat{C}^{(i)}$ 的左右邻居，即 $\hat{C}^{(i-1)}$ 和 $\hat{C}^{(i+1)}$。它将左右邻居的集合分别提取到关联数组 $\mathbf{L_C}$ 和 $\mathbf{R_C}$ 中。关联数组分别存储每个唯一密文 C 的映射及其与左右邻居的共现频率。同时，对手还会根据 **A** 的逻辑顺序信息构造关联数组 $\mathbf{L_A}$ 和 $\mathbf{R_A}$。

然后，基于数据块局部性的频率分析攻击通过每个推理的密文数据块-明文数据块对的邻居进行迭代频率分析。它首先在 **C**（将要进行推测的密文数据块集合）和 **A**（辅助信息-明文数据块的集合）中通过频率分析推理得到出现频率最高的 u 组密文数据块-明文数据块对 $\{(C,M)\}$（u 是该攻击开始时用于选取明密文数据块对数量的参数，一般设为 5）。因为基于观察得到高频率数据块的频率等级（相对排名）在不同版本的备份中是稳定的，所以推理结果可能是真实的（即，目标密文数据块与推理的明文数据块是正确的唯一映射）。对于每个推理的明密文数据块对 (C,M)，攻击发现它们的左右邻居 C 和 M 的共现频率最高，由于局部性的原因，

M 的左右邻居可能分别是 C 的相应左右邻居的原始明文数据块。因此，将 C 和 M 的最高频率左（右）邻居添加到推理的密文数据块-明文数据块对的集合中（从存储密文数据块 C 的左右邻居的集合的关联数组 $\mathbf{L_C}$ 和 $\mathbf{R_C}$ 中获得；对于辅助信息：明文数据块 A，同理，从 $\mathbf{L_A}$ 和 $\mathbf{R_A}$ 中获得）。最后，不断迭代此过程，直到检查完每个推理得到的的密文数据块-明文数据块对的邻居。

因为数据块的顺序由数据块局部性保留，所以它可以使用频率分析来推理在相应的邻居中具有相同的共现频率等级的新的密文数据块-明文数据块对。M 的左右邻居可能是 C 的相应邻居的原始明文。因此，攻击通过这些新推理的密文数据块-明文数据块对的邻居进一步迭代相同的频率分析，从而增加攻击的严重性。

然而，基于数据块局部性的频率分析攻击存在一个主要问题：它引入了大量误报（即，不正确的密文数据块-明文数据块对）。由于频率分析的主要思想是将密文映射到具有相同频率等级的明文，因此对频率等级的任何干扰（例如，跨备份的更新）都可能导致推理出不正确的密文数据块-明文数据块对，这又将导致对其邻居的推理发生错误。虽然基于数据块局部性的频率分析攻击被证明可以有效地推理出真正的密文数据块-明文数据块对的很大一部分，但是对手对于判断每个推理的密文数据块-明文数据块对的真实性置信度较低。例如，根据评估显示，攻击可以在某些情况下可以正确地推理出 15.2 % 的密文数据块-明文数据块对，但在其推理结果中大约 65.2 % 的推理结果是错误的。

基于数据块局部性的频率分析攻击方法对频率排名非常敏感，如果频率排序受到任何干扰都会极大的降低推理精度。而这些推理错误的明密文数据块对将会干扰到对其相邻数据块的推理迭代过程，导致错误更加严重。即使通过下调参数 u 的大小来限制返回数据块的数量，以提高推理的正确率（仅将频率显著高于其他数据块对的数据块对返回），但会导致推利率的严重下滑（推理得到的数据块对总数大幅减少）。

4.2 基于分布的频率分析攻击方法定义

基于分布的频率分析攻击的攻击扩展了基于数据块局部性的频率分析攻击 [44] 以显著消除误报。它利用备份工作负载中的数据块局部性性，就像基于数据块局部性的攻击一样。在基于数据块局部性的攻击的基础上，对于 $\mathbf{C}$ 中的每个唯一密文 C，根据该数据块与其邻居的共现频率来测量其相对频率分布；同时，用同样的方法计算 $\mathbf{A}$ 中每个唯一明文 M 的相对频率分布。根据观察，对于推理正确的密文数据块-明文数据块对 (C, M)，C 和 M 应该具有相似的相对频率分布（即，它们与它们各自的邻居的共现频率是相似的）。对于此的理解是，如果 M 是 C 的原始

明文，那么 M 和 C 的相对频率分布可能是相似的。基本原理是 **A** 和 **C** 从同一个源映射得到，因此具有大量未被修改的数据块，所以相对频率分布得到保留。本文将此观察视为局部性属性的更一般化概念，并根据该条件过滤可能不正确的密文数据块-明文数据块对，以提高推理的准确性。

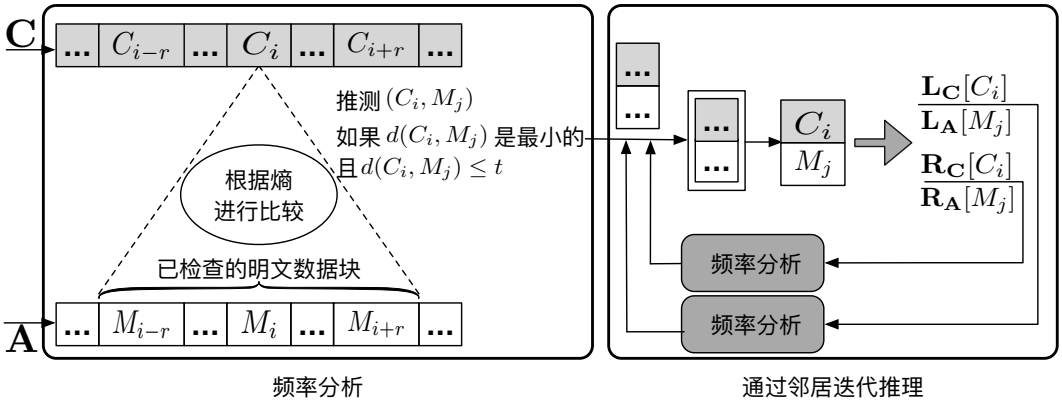

图 4-1 基于分布的频率分析攻击工作流程

图4-1展示了基于分布的频率分析攻击方案的工作流程。

首先按 **C** 和 **A** 中的频率对具有唯一性的密文和明文进行排序。与基于数据块局部性的频率分析攻击 [44] 一样，通过参数 u 将基础频率分析配置为最多返回 u 个频率最高的明密文数据块对。

接下来，对于每个具有唯一性的密文数据块 C_i（其中 $i, 1 \le i \le u$，代表了该密文数据块的频率排名），检查相近频率排名范围内明文数据块 $M_{i-r}, \ldots, M_i, \ldots, M_{i+r}$（从 $i-r$ 到 $i+r$），其中 r 是可配置参数（默认为 10），表示可以解决的频率排名干扰的最大范围。

对于每个 C_i（其中 $1 \le i \le u$）和相应的 M_j（其中 $i-r \le j \le i+r$），我们通过熵来表示它们的相对频率分布，熵是信息理论中的一个关键概念，用于衡量随机来源产生的信息量。在这里，我们采用熵来量化概率分布的随机性 [?]。具体来说，我们定义了两个随机变量，分别用 X 和 Y 来描述 C_i 与其左右邻居的共现事件，这样事件“$X = C$”表示 C 是 C_i 的左邻居，而事件 $Y = C$ 表示 C 是 C_i 的右邻居。因此，我们基于 $\mathbf{L_C}$ 和 $\mathbf{R_C}$ 来计算两个事件发生的概率：

$$\begin{aligned} \Pr[X = C] &= \frac{\mathbf{L_C}[C_i][C]}{\sum_{C' \in \mathbf{L_C}[C_i]} \mathbf{L_C}[C_i][C']}, \\ \Pr[Y = C] &= \frac{\mathbf{R_C}[C_i][C]}{\sum_{C' \in \mathbf{R_C}[C_i]} \mathbf{R_C}[C_i][C']}, \end{aligned} \tag{4-1}$$

其中 $\mathbf{L_C}[C_i]$ 和 $\mathbf{R_C}[C_i]$ 分别了存储 C_i 的左右邻居，而 $\mathbf{L_C}[C_i][C']$ 和 $\mathbf{R_C}[C_i][C']$ 存储 C_i 与其左右邻居 C' 的共现频率。X 和 Y 都遵循 C_i 的相对频率分布，本文用

$e(\mathbf{L_C}[C_i])$ 和 $e(\mathbf{R_C}[C_i])$ 表征它们的随机性，分别为：

$$\begin{aligned} e(\mathbf{L_C}[C_i]) &= \sum_{C \in \mathbf{L_C}[C_i]} \log_2 \frac{1}{\Pr[X=C]}, \\ e(\mathbf{R_C}[C_i]) &= \sum_{C \in \mathbf{R_C}[C_i]} \log_2 \frac{1}{\Pr[Y=C]}. \end{aligned} \tag{4-2}$$

类似的，本文根据 $\mathbf{L_A}$ 和 $\mathbf{R_A}$ 对每个明文数据块 M_j（其中 $i-r \le j \le i+r$）计算出其对应的熵 $e(\mathbf{L_A}[M_j])$ 和 $e(\mathbf{R_A}[M_j])$，然后本文利用欧几里得距离来量化 C_i 和 M_j 的相对频率分布的相似性，记为 $d(C_i, M_j)$：

$$d(C_i, M_j) = \sqrt{[e(\mathbf{L_C}[C_i]) - e(\mathbf{L_A}[M_j])]^2 + [e(\mathbf{R_C}[C_i]) - e(\mathbf{R_A}[M_j])]^2}. \tag{4-3}$$

显然，只有当他们的熵的欧几里德距离 $d(C_i, M_j)$ 很小时，C_i 和 M_j 才有相似的相对频率分布。因此，如果满足以下条件，则可将 (C_i, M_j) 标识为推理成功的密文数据块-明文数据块对：

- **R1:** $d(C_i, M_j)$ 在 $j \in i-r \le j \le i+r$ 中取到最小值。
- **R2:** $d(C_i, M_j)$ 不大于预先设定的变量 t（例如，t 的默认值为 1）。

值得注意的是，C_i 所对应的原始明文数据块可能不在检查的明文数据块 $M_{i-r}, \ldots, M_{i+r}$ 的范围内。在这种情况下，部分 M_j（$i-r \le j \le i+r$）可能仍然满足条件 R1。这种情况下，本文希望通过 R2 过滤不正确的密文数据块-明文数据块对。

然后，继续采用先前基于数据块局部性的频率分析攻击[44]中的迭代方法（参见章节：4.1）来扩展推理的密文数据块-明文数据块对的覆盖范围。具体来说，对于每个推理的 (C_i, M_j)，应用基于分布的频率分析方案（见上文）通过 C_i 和 M_j 的邻居迭代推理出更多的密文数据块-明文数据块对，在没有新的密文数据块-明文数据块对可以被推理时停止迭代，最终完成基于分布的频率分析攻击的全部过程。

4.2.1 基于分布的频率分析攻击方法总结

总而言之，基于分布的频率分析攻击通过在频率分析中考虑相对频率分布来提供更一般化的局部性概念。它通过三个参数进行配置和调整：

- u：确定通过频率分析返回的密文数据块-明文数据块对的最大数量。
- r：确定要考虑的扰动范围的最大值。
- t：确定欧几里德距离的阈值以过滤可能不正确的推理结果。

先前基于数据块局部性的频率分析攻击[44]可以被视为基于分布的频率分析攻

击在 $r = 0$ 的参数配置下（即没有处理频率排名中的任何干扰）和 $t \to \infty$（即没有过滤任何不正确的推理结果）的特殊情况。

4.3 对数据块大小信息泄漏的利用

本文提出了基于分布的频率分析攻击的高级变体，它使用数据块大小信息来进一步减少误报。具体来说，假设 **C** 中每个密文数据块的大小反映了其原始明文数据块的大小。

本文的想法建立在这样一个基本事实的基础上：如果密文 C 对应于明文 M，则 C 的大小近似于 M 的大小。这是因为加密重复数据删除在要加密的内容中保留了块（即，由对称加密操作的基本单元）的数量信息。本文通过使用这个事实来进一步过滤不正确的密文数据块-明文数据块对 (C, M)（C 和 M 中的块数不同）。

在本课题中，假设每个块的大小都是 16 字节（AES 加密的典型配置）。对于每个经过基于分布的频率分析攻击方法得到的明文数据块-密文数据块对 (C_i, M_j)，分别将 C_i 和 M_j 中的块数导出为 $b(C_i)$ 和 $b(M_j)$：

$$\begin{aligned} b(C_i) &= \lceil \frac{\mathsf{size}(C_i)}{16} \rceil, \\ b(M_j) &= \lceil \frac{\mathsf{size}(M_j)}{16} \rceil, \end{aligned} \tag{4-4}$$

其中 $\mathsf{size}(C_i)$ 和 $\mathsf{size}(M_j)$ 分别是 C_i 和 M_j 的确切大小，$\lceil x \rceil$ 返回了大于或等于 x 的最小整数。在基于分布的频率分析攻击中提出的过滤条件 R1 和 R2（参见章节：4.2）的基础上，给出以下过基于数据块中块数信息的滤条件:

- **R3:** $b(C_i)$ equals $b(M_j)$.

对于使用可变大小的数据分块方法的加密重复数据删除中，不同的数据块可能具有不同的大小，因此使用 R3 可以有效进行过滤。然而，在使用固定大小的数据分块方法的机密重复数据删除中，R3 对于任意一组 (C, M) 始终成立。但此时，仍然可以通过应用 R1 和 R2 来提高频率分析攻击的精度。

4.4 本章小结

本章介绍了本文提出的第一种改进的频率分析方法：基于分布的频率分析攻击方法。给出了该方法的形式定义、方法描述以及对数据块大小信息泄漏的扩展利用方法。

第五章 基于聚类的频率分析攻击方法

本章放松了基于分布的频率分析攻击中的攻击者的攻击条件要求，提出了基于聚类的频率分析攻击，它不需要使用明文数据块的细粒度排序信息。相反，它利用相似性这一属性来推理来自相似的数据段（即，由数据块块聚合形成的更大的数据单元）的原始数据块，而不依赖于每个数据段中的数据块的排序。本章首先介绍相似性的概念，然后展示如何使该属性进行推理攻击。

5.1 背景知识

5.1.1 min-wise independence 条件

对于 min-wise independence 置换族[67] 有如下定义：设 S_n 是 n 元集合 $[n]$ 上所有置换组成的 n 元对称群。称置换族 $F \subseteq S_n$ 满足 min-wise independence 条件是指：对任意 $X = \{x_1, x_2, \cdots, x_m\} \in [n]$ 和任意 $x_i - inX$，从集合 F 中随机、均匀地选取函数 h，计算 $H(X) = \{h(x_1), h(x_2), \cdots, h(x_m),\}$，有下式成立：

$$Pr(min(H(X)) = h(x_i)) = \frac{1}{|X|} \tag{5-1}$$

即 X 中的所有元素在 h 的作用下都有相等的概率成为 $H(X)$ 中的最小元。在实际应用中，完全满足 min-wise independence 条件的哈希函数很难实现，通常使用近似满足 min-wise independence 条件[67] 的哈希函数即可。

5.1.2 Broder 定理

Broder 定理[68] 的定义为：两集合 S_1 和 S_2，$H(S_i) = \{h(x_k) | \forall x_k \in S_i\}$，h 是满足 min-wise independence 条件的哈希函数，$min(S_i)$ 代表集合 S_i 中的最小元，则

$$\Pr[\min\{\mathrm{H}(S_1)\} = \min\{\mathrm{H}(S_2)\}] = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} \tag{5-2}$$

5.1.3 相似性

相似性[69] 指出来自同一来源的备份文件可能类似并且共享大部分相同的数据块。备份文件之间的相似性可以通过 Broder 定理[68] 来量化。具体来说，如果将每个文件视为一个数据块的集合 S（即忽略它们的顺序），Broder 定理指出如果两组数据块共享相同的最小数据块哈希值的概率很高，则两个集合包含的数据块可能

大部分都相同，反之亦然：

$$\Pr[\min\{\mathrm{H}(S)\} = \min\{\mathrm{H}(S')\}] = \frac{|S \cap S'|}{|S \cup S'|} \tag{5-3}$$

其中 $\mathrm{H}(\cdot)$ 是一个从 min-wise independence 置换族中随机统一选择的哈希函数，$\min\{\mathrm{H}(S)\}$ 是 S 的最小数据块的哈希。为了便于描述，本文使用 MinHash 来表示集合中元素的最小哈希值。

针对重复数据删除的各个方面（性能方面[28,47,69]，安全方面[44]）的已有工作已经利用了相似性这一属性来保持存储效率。具体来说，它们仅针对共享相同最小数据块哈希的文件（即类似的文件）采用 MinHash 作为一个数据块集合中所有数据块加密使用的密钥进行重复数据删除。根据 Broder 定理可以推理得到这类文件可能具有大量相同的数据块，因此这种近似精确的重复数据删除只会导致存储效率的轻微降低。与先前的方法不同[28,44,47,69]，本文应用相似性来提高频率分析攻击的有效性。

5.2 基于聚类的频率分析攻击方法定义

现在提出基于聚类的频率分析攻击方法（图：5-1），它基于相似性在加密重复数据删除中推理密文数据块的原始明文数据块。

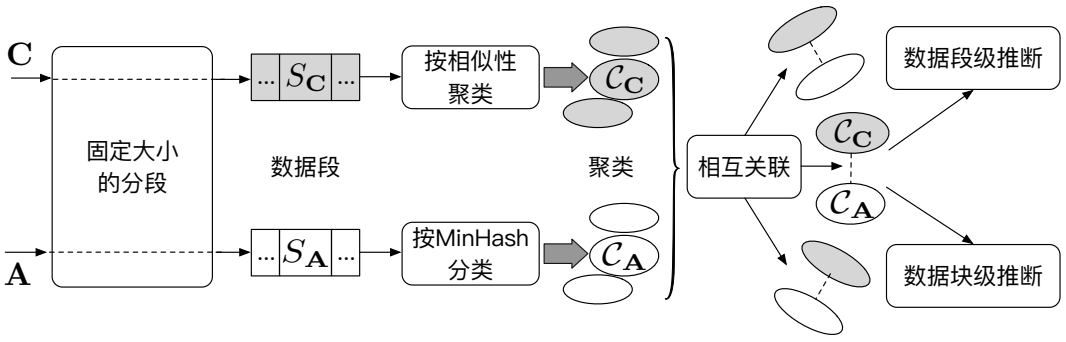

图 5-1 基于聚类的攻击方法的工作流程：用 MinHash 表示明文数据段的最小块散列

为了利用相似，本文首先在数据块概念的基础上引入数据段的概念。具体来说，将 **C** 划分为为一些粗粒度的数据单元，成为密文数据段。密文数据段使用 $S_{\mathbf{C}}$ 来表示，其中包含有 **C** 中的多个相邻的密文数据块。在本课题中，应用了定长分段方法，以确保每个数据段具有相同分的大小（例如，默认大小为 4MB）。同样的，对于攻击者所具有的辅助信息 **A**，也同样使用定长分段方法产生多个明文数据段，每个数据段由 $S_{\mathbf{A}}$ 表示。

注意，一些可变大小的分段方案[28,46]在其内容与特定模式匹配的数据块之后设置分段边界，从而解决固定大小分段方案所面临的边界移位问题。但是，本文不能在攻击中使用这些可变大小的分段方案[28,46]。其原因是密文数据段中的数据块的原始内容受到对称加密的保护，攻击者无法确保密文和明文数据段的边界匹配采用的是相同的模式。该问题会导致密文和明文数据段不兼容，从而会降低基于聚类的攻击方法所推理的数据段级的数据量（请参阅本章后半部分）。

本方法通过类似的数据段来推理密文-明文对。给出 $S_{\mathbf{M}}$ 是对应于密文数据段 $S_{\mathbf{C}}$ 的明文数据段（即 $S_{\mathbf{M}}$ 中的每个明文数据块对应于 $S_{\mathbf{C}}$ 中的某些密文数据块，反之亦然）。根据 Broder 定理，如果一个明文数据段 $S_{\mathbf{M}}$ 与辅助信息中的明文数据段 $S_{\mathbf{A}}$ 具有相同的最小数据块哈希（称为 h），则 $S_{\mathbf{M}}$ 和 $S_{\mathbf{A}}$ 可能拥有拥有大量相同的明文数据块。这意味着 $S_{\mathbf{C}}$ 中的密文数据块很可能是从 $S_{\mathbf{A}}$ 中的明文数据块映射得到的。换言之，攻击者首先按照最小数据块哈希对 $\mathbf{A}$ 中的所有明文数据段进行分类，以此获得多个明文数据段集合（聚类的不同类别）。将每个明文数据段集合用 $\mathcal{C}_{\mathbf{A}} = \{S_{\mathbf{A}}\}$ 表示，对应于该集合中包含的所有明文数据段共有的唯一最小数据块哈希。同理，攻击者将密文数据段也进行上述分类操作，对于每个分类的集合用 $\mathcal{C}_{\mathbf{C}} = \{S_{\mathbf{C}}\}$ 表示）。然后，攻击者从对应于某一最小数据块哈希 h 的 $\mathcal{C}_{\mathbf{A}}$ 中推理出 $\mathcal{C}_{\mathbf{C}}$ 的原始明文数据。

在本文中，定义任意两个密文数据段 $S_{\mathbf{C}}$ 和 $S'_{\mathbf{C}}$ 的聚类距离 $d(S_{\mathbf{C}}, S'_{\mathbf{C}})$ 为 1 减去相同密文的分数：

$$d(S_{\mathbf{C}}, S'_{\mathbf{C}}) = 1 - \frac{|S_{\mathbf{C}} \cap S'_{\mathbf{C}}|}{|S_{\mathbf{C}} \cup S'_{\mathbf{C}}|}. \tag{5-4}$$

在推理过程中，相同的密文可能会在 $S_{\mathbf{C}}$ 或 $S'_{\mathbf{C}}$ 中重复出现，而 $|S_{\mathbf{C}} \cap S'_{\mathbf{C}}|$ 和 $|S_{\mathbf{C}} \cup S'_{\mathbf{C}}|$ 会分别返回其交集和并集中具有唯一性的密文数据块的数量。显然，$d(S_{\mathbf{C}}, S'_{\mathbf{C}})$ 越小，则说明 $S_{\mathbf{C}}$ 和 $S'_{\mathbf{C}}$ 更可能对应相同的最小数据块哈希。在本章中，采用凝聚法层次聚类（AHC）[70]方法来根据距离信息聚合相似的密文数据段。具体来说，攻击者从其拥有的每个密文数据段开始，将每个密文数据段视为一个单独的聚类类别，然后计算各个聚类之间的距离，将两个距离最近的（最相似的）聚类合并在一起，不断迭代这一过程直到任意两个聚类间的距离均大于给定最大距离参数 k 时停止迭代操作。

对于最终得到的每个密文聚类 $\mathcal{C}_{\mathbf{C}}$，攻击者在考虑频率分布的前提下通过频率分析将其与一些明文聚类 $\mathcal{C}_{\mathbf{A}}$ 相关联。该操作方法基于如下观察：相同的密文数据段（明文数据段）可以在相同或不同的密文数据段（明文数据段）中重复出现，并

且相同的密文（明文）数据段也可以在相同的密文（明文）聚类中重复出现。同时，攻击者检查每个聚类中明文数据块或密文数据块的频率分布，并且认为相似的聚类（即，对应于相同的最小数据块哈希）中数据块的频率分布也是相似的。

本文按如下的顺序来设计频率分析攻击方案。首先，将所有可用的密文和明文聚类分别按它们包含的逻辑密文数据块和明文数据块的总数进行排序。然后，计算一个关联数组 **F** 用于存储相应聚类中每个具有唯一性的密文数据块或明文数据块的频率。基于 **F**，计算密文数据块 C 存在于密文聚类 $\mathcal{C}_{\mathbf{C}}$ 中的概率 $\Pr[C \in \mathcal{C}_{\mathbf{C}}]$，并进一步计算 $\mathcal{C}_{\mathbf{C}}$ 的熵 $e(\mathcal{C}_{\mathbf{C}})$：

$$\begin{aligned} \Pr[C \in \mathcal{C}_{\mathbf{C}}] &= \frac{\mathbf{F}[\mathcal{C}_{\mathbf{C}}][C]}{\sum_{C' \in \mathcal{C}_{\mathbf{C}}} \mathbf{F}[\mathcal{C}_{\mathbf{C}}][C']} \\ e(\mathcal{C}_{\mathbf{C}}) &= \sum_{C \in \mathcal{C}_{\mathbf{C}}} \log_2 \frac{1}{\Pr[C \in \mathcal{C}_{\mathbf{C}}]} \end{aligned} \tag{5-5}$$

其中 $\mathbf{F}[\mathcal{C}_{\mathbf{C}}][C]$ 存储了 $\mathcal{C}_{\mathbf{C}}$ 中密文数据块 C 的频率。同理，计算出明文聚类 $\mathcal{C}_{\mathbf{A}}$ 的熵 $e(\mathcal{C}_{\mathbf{A}})$。与基于分布的频率分析攻击（参见4.2）类似，本文使用参数 (u, r, t) 来配置频率分析攻击方案。如果密文聚类 $\mathcal{C}_{\mathbf{C}}$ 和明文聚类 $\mathcal{C}_{\mathbf{A}}$ 满足如下三个条件则可说明两个聚类相似：

- $\mathcal{C}_{\mathbf{C}}$ 的排名数值大小不大于 u。
- $\mathcal{C}_{\mathbf{C}}$ 和 $\mathcal{C}_{\mathbf{A}}$ 的排名数值差距大小不大于 r。
- $e(\mathcal{C}_{\mathbf{C}})$ 和 $e(\mathcal{C}_{\mathbf{A}})$ 的差异是最小的，且不超过 t.

然后，对于每个相似的聚类对 $(\mathcal{C}_{\mathbf{C}}, \mathcal{C}_{\mathbf{A}})$，攻击者在两个级别分别进行明密文对的推理工作。

- **数据段级推理**

  如果 $\mathcal{C}_{\mathbf{C}}$ 和 $\mathcal{C}_{\mathbf{A}}$ 拥有相同数量的逻辑数据块（即密文数据块或明文数据块）以及相同的熵，则 $\mathcal{C}_{\mathbf{C}}$ 与 $\mathcal{C}_{\mathbf{A}}$ 完全映射的概率很高。在这种情况下，攻击者对粗粒度的数据段进行攻击。具体来说，攻击者首先在 $\mathcal{C}_{\mathbf{C}}$ 中基于其密文的频率分布计算每个密文数据段 $S_{\mathbf{C}}$ 的熵，并在 $\mathcal{C}_{\mathbf{A}}$ 中基于其明文的频率分布计算每个明文数据段 $S_{\mathbf{A}}$ 的熵。

  如果 $S_{\mathbf{C}}$ 和 $S_{\mathbf{A}}$ 中逻辑数据块的总数和熵一致，则攻击者可以推理密文数据段 $S_{\mathbf{C}}$ 是由命文数据段 $S_{\mathbf{A}}$ 映射得到的。根据实验表明，基于聚类的频率分析攻击中大部分推理正确的内容来自于数据段级推理。同时，攻击者还可以利用额外的对抗性知识（例如，逻辑块顺序）来进一步恢复这些推理出的数据段中的每个数据块的明文。

- **数据块级推理**

  如果 $\mathcal{C}_{\mathrm{C}}$ 和 $\mathcal{C}_{\mathrm{A}}$ 中具有的逻辑块数量或熵不同，攻击者对细粒度的数据块进行攻击。具体来说，攻击者分别对 $\mathcal{C}_{\mathrm{C}}$ 和 $\mathcal{C}_{\mathrm{A}}$ 中的具有唯一性的密文数据块和明文数据块按照出现频率进行排序，以此根据频率排名的顺序推理明文数据块-密文数据块对。

  然而，实验发现数据块级推理在本文给出的实验数据集中表现不佳。可能的原因是每个聚类包括大量逻辑数据块，这降低了频率分析的有效性。但即便如此，本文预计数据块级推理可以在实践中正确推理出更多的密文数据块-明文数据块对，当某些聚类中的逻辑数据块数量有限时效果会更佳。

5.2.1 基于聚类的频率分析攻击方法总结

总而言之，基于聚类的频率分析攻击利用相似性，在类似的聚类类别中使用频率分析攻击以推理密文数据块-明文数据块对。除 u，r 和 t 外，它还由参数 k 配置，该参数指定了配对两个聚类类别的距离上限。

尽管可能受到固定大小数据段的边界偏移问题的影响，但本文认为基于聚类的频率分析攻击对 VM 磁盘映像的攻击效果是显著的。VM 映像文件在创建时被分配了固定的大小，并且在其生存期内无法改变。在开始写入数据前，VM 映像中的所有空间均由全零数据进行填充。在将数据存入映像的过程中，通过将需要用到的部分空间进行重写来完成数据存入过程。在章节7.3中，本文研究了针对 VM 磁盘映像的基于聚类的频率分析攻击的有效性。

5.3 本章小结

本章介绍了本文提出的第二种改进的频率分析方法：基于聚类的频率分析攻击方法。给出了该方法的相关背景知识、形式定义、方法描述以及最佳的适用场景。

第六章 数据集与实际系统调研

6.1 数据集调研

本文调研与使用以下数据集用于评估提出的两种攻击方法的效果，通过模拟攻击评估重复数据删除中信息泄漏带来的安全隐患（参见第二章）。

6.1.1 FSL 数据集

表 6-1 FSL 数据集特征

类别	备份中的特征		
	# 逻辑数据块数（百万）	# 唯一数据块数（百万）	近似逻辑大小（GB）
User004	1.0-1.3	0.7-0.9	10GB/备份 × 14 个
User007	3.5-5.2	2.3-3.6	39GB/备份 × 14 个
User012	25.0-26.4	8.9-9.7	244GB/备份 × 14 个
User013	1.8-5.7	1.2-4.2	22GB/备份 × 14 个
User015	13.4-20.5	9.0-11.0	158GB/备份 × 14 个
User028	6.0-10.3	3.5-6.8	77GB/备份 × 14 个

表6-1总结了 FSL 实验数据集的相关特征信息，其中 # 逻辑数据块数和 # 唯一数据块数表示每个实验快照中逻辑数据块数量和具有唯一性的数据块数量。

此数据集由 Stony Brook 大学的文件系统和存储实验室（FSL）收集[71–73]。本文使用 fslhomes 快照进行实验，每个快照包括 48 位数据块哈希的有序列表，这些哈希是由平均大小为 8KB 的可变大小分块生成的，以及相应的元数据信息（例如，数据块大小、文件名、扩展名等）。本文从 2013 年 1 月 22 日到 5 月 21 日选择快照，并选择在整个持续时间内具有完整每日快照的六个用户（即 User004，User007，User012，User013，User015 和 User028）用于后续实验分析。本文以周为单位选取备份数据，因此为每个用户选出 14 个每周完整备份。

6.1.2 VM 数据集

表6-2总结了 VM 实验数据集的相关特征信息，其中 # 逻辑数据块数和 # 唯一数据块数表示每个实验快照中逻辑数据块数量和具有唯一性的数据块数量。由于所有的 VM 映像都具有 10GB 的固定大小，因此其具有相同数量的逻辑块。。

此数据集是从某大学 2014 年春季的编程课程中收集的。原始数据集包括为课程中注册的学生提供的大量 VM 镜像快照，每个快照的大小固定为 10GB，每个快

表 6-2 VM 数据集特征

类别	备份中的特征		
	# 逻辑数据块数（百万）	# 唯一数据块数（百万）	近似逻辑大小（GB）
User1	2.6	∼0.9	25GB/备份 × 13 个
User2		1.1-1.3	28GB/备份 × 13 个
User3		0.9-1.1	26GB/备份 × 13 个
User4		∼0.9	25GB/备份 × 13 个
User5		0.9-1.0	25GB/备份 × 13 个
User6		0.9-1.1	27GB/备份 × 13 个

照中的数据用固定大小为 4KB 的数据块通过 SHA-1 算法计算得到的哈希的列表表示。本文使用 6 个用户（即 User1-User6）的数据，每个用户的数据由 13 个每周备份组成。

6.1.3 MS 数据集

表 6-3 MS 数据集特征

类别	备份中的特征		
	# 逻辑数据块数（百万）	# 唯一数据块数（百万）	近似逻辑大小（GB）
Win7	61.6-61.8	61.1-61.3	425GB/备份 × 4 个
Serv-03	10.6-10.7	8.4-8.5	78GB/备份 × 4 个
Serv-08	∼6.5	∼3.8	48GB/备份 × 4 个
Vista-B	∼7.6	∼2.0	55GB/备份 × 4 个
Vista-U	∼21.0	∼10.4	159GB/备份 × 4 个

表6-3总结了 MS 实验数据集的相关特征信息，其中 # 逻辑数据块数和 # 唯一数据块数表示每个实验快照中逻辑数据块数量和具有唯一性的数据块数量。

此数据集由 Microsoft[74] 收集，并在 SNIA[75] 上公布。原始数据集包含 2009 年 9 月 5 日至 10 月 31 日的 Windows 文件系统快照。每个快照由通过 Rabin Fingerprinting[52] 获得的不同平均大小的 40 位数据块哈希表示。本文使用以下类别的操作系统快照用于试验：Windows 7（Win7），Microsoft Windows Server 2003（Serv-03），Microsoft Windows Server 2008（Serv-08），Microsoft Windows Vista Business Edition（Vista-B）和 Microsoft Windows Vista Ultimate Edition（Vista-U）。

在每个类别中，选择四个快照（这些快照的平均数据块大小为 8KB。

6.2 实际系统调研

6.2.1 SDFS

SDFS [76] 是一种分布式、可扩展的文件系统，旨在为应用程序提供具有较强灵活性的内联重复数据删除。该系统有两种部署方法：

- **单节点配置**：实际运行时将数据存储在本地文件系统中。
- **多节点配置**：实际运行中将数据存储在远端存储服务商/服务器中。

6.2.1.1 系统的整体结构

SDFS 系统的整体结构如下所示：

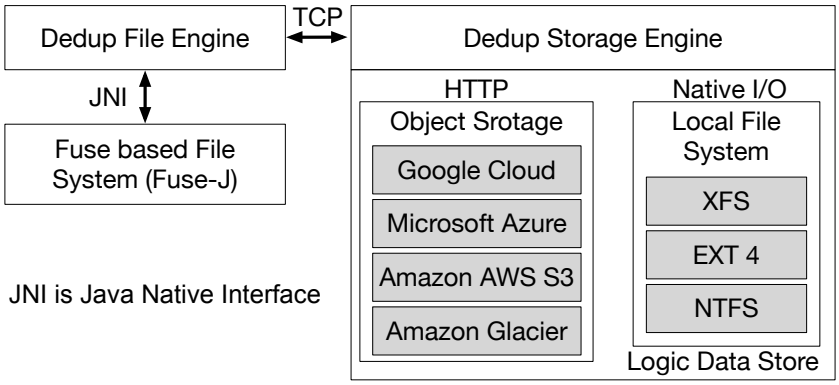

图 6-1 SDFS 文件系统的运作流程

该系统主要由以下四部分组成，其中 Dedup File Engine 和 Dedup Storage Engine 是 SDFS 系统的主要组件：

- **Fuse Based File System (Fuse-J)**
- **Dedup File Engine**
- **Dedup Storage Engine**
- **Logic Data Store**

6.2.1.2 Dedup File Engine

本节将说明 Dedup File Engine 的工作原理以及相关的系统实现。Dedup File Engine 的工作流程如下所示：

SDFS 将 Dedup File Engine 通过 Java Native Interface（JNI）连接到由 FUSE-J 驱动的实际文件系统的的卷，使得 Dedup File Engine 可以通过系统文件系统直接进行文件操作。

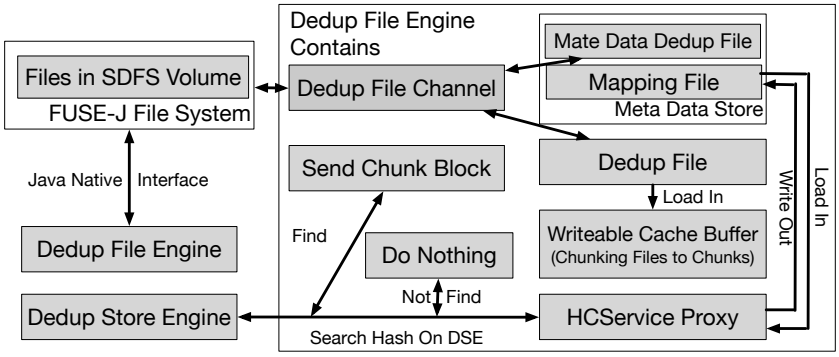

图 6-2 SDFS 文件系统的 Dedup File Engine 运作流程

Fuse-J Java Native Interface

- 文件统计和高级文件操作是通过获取和设置扩展文件属性（getfattr setfattr）来完成的。
- 每个挂载到 SFDS 的卷都拥有其独立的的 Fuse-J 接口和 Dedup File Engine 实例。

Dedup File Channel

本模块 Fuse-J 和 DedupFile 之间用于 I/O 命令的接口。该接口提供了一些方法（例如获取文件元数据/读取或写入文件）用于获取将进行重复数据删除的文件。

Meta Data Store

在 SDFS 文件系统中，文件通过两个不同的元数据进行表示，并将两种不同的元数据保存在两个不同的文件中。这两个文件分别为:

- Meta Data Dedup File

  该文件用于存储进行重复数据删除的原始文件的文件属性元数据，包括文件大小、atime、ctime、acls 以及指向关联映射文件的链接等。

- Mapping File

  该文件包含用于重复数据删除的文件的数据块记录列表，其存储了文件中每个数据块在原始文件中的位置、数据块是否具有唯一性、数据块的哈希值以及该数据块是否存储在远程服务器中。

DedupFile

该文件用于存储将要进行重复数据删除的文件的记录，其中的内容是挂载到 SDFS 文件系统的券中各个文件的哈希与完整文件路径与文件名的映射关系表。

WritableCacheBuffer

本数据结构包含一系列正在读取或写入的数据信息。在 I/O 活动期间，DedupFile 会缓存其中的许多内容。在这部分中，SDFS 使用 SparseDedupFile 类来处理 writableCacheBuffer 中的数据（即进行文件分块以获得获取逻辑数据块）。

SDFS 支持两种数据块分块方法：固定大小的分块方法和 Rabin 可变大小数据块分块方法。

HCService Proxy (HC == HashChunk)

HCServiceProxy 根据哈希的第一个字节（如果设置了多个 Dedup Storage Engine）为当前数据块查找到适当的重复数据删除存储引擎。HCServiceProxy 通过查询相应的 Dedup File Engine 以确定当前哈希所对应的数据块是否已存入存储系统（是否是重复的数据块），如果该数据块是重复的，则不进行任何操作，如果当前数据块是具有唯一性的，则将其加入存储系统中进行持久化存储。

6.2.1.3 Dedup Storage Engine

Dedup Storage Engine（DSE）存储、检索和删除所有 SDFS 分割形成的数据块。重复数据删除存储引擎可以作为 SDFS 卷（单节点模式）的一部分运行，或在云端单独运行（多节点模式）。Dedup Storage Engine 的运作流程如下所示：

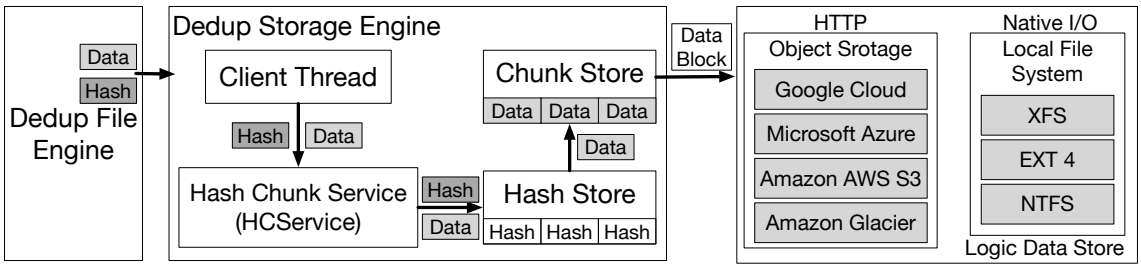

图 6-3 SDFS 文件系统的 Dedup Storage Engine 运作流程

1. 客户端线程接收 *WRITE_HASH_CMD* 或 *WRITE_COMPRESSED_CMD* 信号并读取哈希和数据。
2. 客户端线程将哈希传递给 HashChunkService。
3. HashChunkService 将哈希和数据传递给适当的 HashStore。
4. HashStore 存储哈希并将数据发送到指定的 ChunkStore。
5. ChunkStore 存储数据。

DES 由 2 个基本组件组成：

HashStore

HashStore 用于存储所有数据块的哈希与其对应的逻辑数据的存储位置。

ChunkStore

ChunkStore 负责存储和检索与特定哈希相匹配的数据块原始逻辑数据。

- 本地数据块存储
  - 单文件数据块存储：将所有数据块写入一个文件中（文件大小与数据块总量有关）。

  - 基于文件的数据块存储：将所有数据块依次写入固定大小的文件中。
- 远程数据块存储
  - Amazon S3
  - Microsoft Azure
  - Google Cloud

6.2.1.4 SDFS 文件系统调研总结

SDFS 文件系统提供了商业使用级别的（加密）重复数据删除，但其中重复数据删除工作均先进行文件内的重复数据删除，再进行文件间的重复数据删除。其中的数据块处理顺序随机，不会发生数据块逻辑顺序信息的泄漏，因此可以防范本文提出的第一种攻击方法（基于分布的频率分析攻击方法）。但本文提出的第二种攻击手段在 SDFS 中仍然可以发挥作用。

6.2.2 Destor

Destor[77] 是重复数据删除的评估平台，其包含了大量重复数据删除中常用的方法，可以用于常用重复数据删除的方法测试与应用效果模拟评估。系统原型中包含的主要特征有：

- 基于容器的数据块存储。
- 数据块级的管道通信。
- 三种重复数据删除方案：
  - 基于固定大小的数据块分块的重复数据删除。
  - 基于内容定义数据块分块方法的重复数据删除。
  - 近似的文件级重复数据删除。
- 多种文件重写算法：CFL、CBR、CAP 和 HAR 等。
- 多种文件还原算法：LRU、最优替换算法和前滚组装。

6.2.2.1 Destor 系统原型调研总结

Destor 不同与商业化的 SDFS 文件系统，该系统原型是重复数据删除领域中著名的重复数据删除工具集合与样例。系统中的数据块处理是按照原有逻辑明文数据块的逻辑顺序进行的，因此会受到本文提出的两种攻击方法的威胁。

6.3 本章小结

本章介绍了本文用于评估攻击效果的三种真实世界数据集的来源和基本信息，以及两种现有重复数据删除系统的相关调研工作。

第七章 实验测试与分析

在本章中，将给出基于真实世界数据集的评估结果，以证明本文针对加密重复数据删除提出的频率分析攻击方法的有效性（对加密重复数据删除的威胁的严重性）。

7.1 实验方法

本文所使用的真实世界数据集不包数据的实际内容，因此基于数据块哈希模拟攻击者所拥有的的对抗性知识。具体来说，本文将一些快照中的有序的数据块哈希列表作为辅助信息 **A** 和原始明文信息 **M**。为了模拟加密过程，在 **M** 中的每个原始数据块哈希（表示明文）上应用额外的哈希函数，并将结果截断为当前使用的数据集中约定的哈希值的长度。截断的结果用于模仿 **C** 中的密文。对于每个推理的密文数据块-明文数据块对 (C, M)，本文通过在 M 上应用相同的模拟加密并将结果与 C 进行比较来验证其正确性。特别的，基于聚类的频率分析攻击可以在数据段级别进行操作，断出密文数据段-明文数据段对 $(S_{\mathbf{C}}, S_{\mathbf{A}})$。在这种情况下，本文通过检查 $S_{\mathbf{C}}$ 中的每个密文是否完全映射到 $S_{\mathbf{A}}$ 中的每个明文来评估 $(S_{\mathbf{C}}, S_{\mathbf{A}})$ 的正确性。

本文通过推理率和推理精度两个标准来衡量攻击的有效性（参见第三章）。

7.2 基于分布的攻击的实验结果

7.2.1 实验 1（参数的影响）

本节将评估参数 (u, r, t) 的变化在基于分布的频率分析攻击中带来的影响。实验使用 FSL 数据集（参见6.1.1）来进行评估。在实验中使用每个用户的第 12 周备份作为辅助信息来推理该用户相应的第 14 周备份中的原始明文数据。出于控制变量的需求，实验将在固定另两个参数的前提下对三个参数中的一个的变化进行测试和分析。

7.2.1.1 参数 u 的影响

首先，配置参数 $t \to \infty$ 和 $r = 0$ 来评估参数 u 变化的影响（在这种情况下，基于分发的攻击将退化到基于数据块局部性的的攻击[44]）。

图7-1显示了将参数 u 从 2 变为 512 时对推理率和推理精度带来的影响。对于推理率的结果，本实验的观察结果与已有的工作[44]相同。推理率首先随着 u 的增

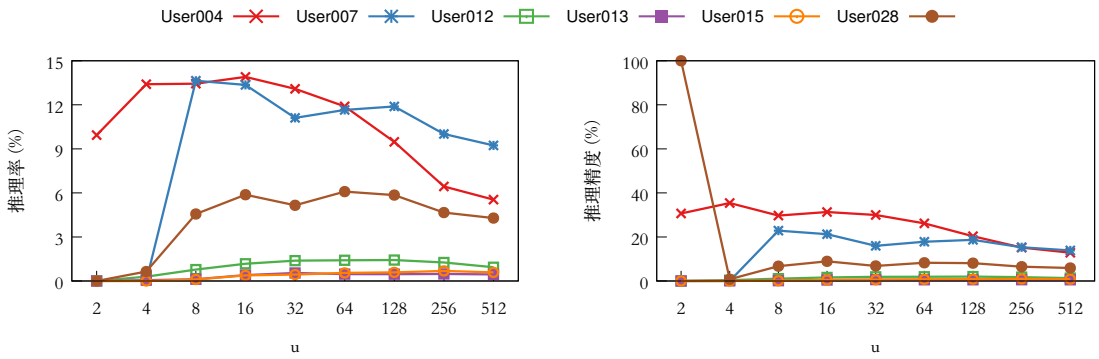

图 7-1 基于分布的攻击实验 1：参数 u 的变化对推理结果的影响（$r = 0$；$t \to \infty$）

大而提高，这是因为频率分析攻击可以推理出更多的密文数据块-明文数据块对。推理率在达到最大值之后（例如，User004 为 13.9%，User007 为 13.6%，User012 为 1.4%，User013 为 0.5%，User015 为 0.7%，User028 为 6.1%）开始下降。其原因在于频率分析引入了大量的误报，这些误报会继续影响基于分布的攻击对每个数据块相邻邻居的推理。同时，部分特殊情况下的推理率约为 0.0001%，这意味着攻击只能推理出几个密文数据块-明文数据块对。

已有的工作[44]不提供基于数据块局部性的攻击的推理精度信息，因此本文提出的两种推理攻击方法的实验的推理精度的部分不与其进行对比。实验中观察到除 User028 的数据在 $u = 2$ 的条件下没有引入任何误报外，针对所有用户备份数据的推理精度都处于相当低的水平（小于 40%）。随着 u 的增大，推理精度会略有下降。例如，当 u 从 2 增加到 512 时，User004 的推理精度从 30.7% 下降到 12.8%。

结论（1）：相对较大的 u 提高了推理率，但降低了推理精度（即，引入了更多的误报）

7.2.1.2 参数 r 的影响

然后，通过 u 变化的影响分析，本文选择增加推理的密文数据块-明文数据块对的覆盖范围，同时使用 r 和 t 来过滤可能的误报。因此，本文将 User004，User013 和 User015 的 u 设置为 128，User007，User012 和 User028 的设置分别为 256，以评估 r 和 t 变化的影响。

本实验中首先令 $t \to \infty$，然后评估 r 变化带来的影响。图7-2展示了参数 r 变化的实验结果。实验观察到大多数用户的推理率随着 r 而增加。例如，当将 r 从 0 变为 16 时，User004 推理率从 9.5% 增加到 16.0%，User007 从 10.0% 增加到 13.0%，User013 从 0.5% 增加到 0.7%，User028 从 4.7% 增加到 5.6%。发生这种现象的原因是基于分布的攻击减少了直接频率排序的干扰，并推理出更多正确的密文数据

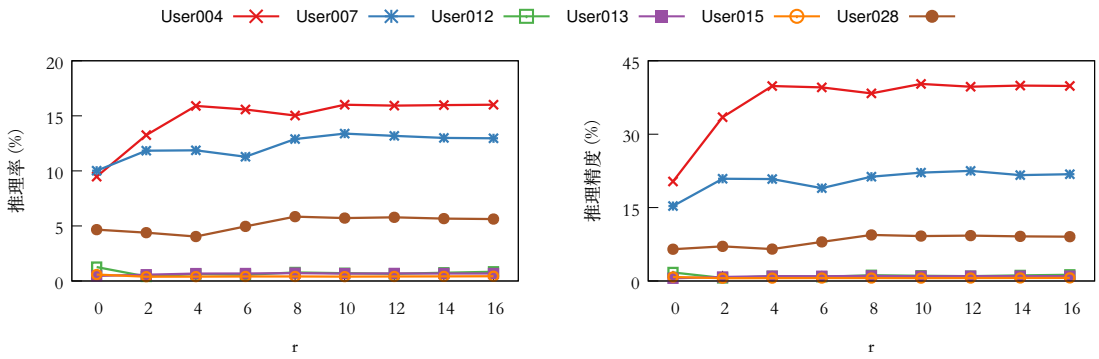

图 7-2 基于分布的攻击实验 1：参数 r 的变化对推理结果的影响（User004、User013 和 User015 中 $u = 128$，User007、User012 和 User028 中 $u = 256$；$t \to \infty$）

块-明文数据块对。另一方面，对于 User012 和 User015，推理率分别从 1.3% 降至 0.8%，从 0.6% 降至 0.4%。原因是随着检查的扩大，可能引入更多的误报。同时，所有用户的推理精度均处于低水平（小于 45 %），并且具有与其相应的推理率类似的变化趋势。

结论（2）：较大的 r 提供了识别正确的密文数据块-明文数据块对的更多机会，但也增加了发生误报的概率。

7.2.1.3 参数 t 的影响

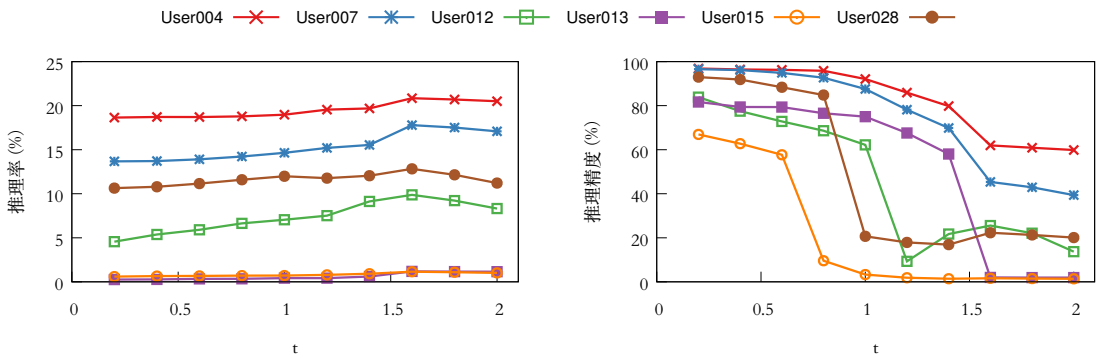

图 7-3 基于分布的攻击实验 1：参数 t 的变化对推理结果的影响（User004、User013 和 User015 中 $u = 128$，User007、User012 和 User028 中 $u = 256$；$r = 10$）

最后，设定 $r = 10$ 来评估参数 t 变化的影响。图7-3展示了参数 t 变化的实验结果果。当 t 很小（例如，小于 0.5）时，实验观察到攻击误判并过滤了大量的密文数据块-明文数据块对，即使它们是正确的（即，引入了降低推理率的假阳性结果）。随着 t 的增加，假阳性的结果数量减少。当 $t = 1.5$ 时，对于 User004，User007，User012，User013，User015 和 User028，推理率分别达到最大值 21.2%，

18.2%，10.4%，1.2%，1.2% 和 13.5%。当 t 进一步增大到 2 时，相应的推理率分别降至 20.5%，17.1%，8.3%，1.2%，1.0% 和 11.2%。其原因是如果 t 过大，攻击就无法有效地过滤误报。因此，所有用户数据的推理精度都会随着 t 的增大而下降。

结论（3）：较小的 t 过滤了大部分频率分析带来的误报，但它引入了更多的假阳性结果，导致了推理率的下降。

7.2.2 实验 2（与已有工作的对比）

本文将基于分布的攻击的有效性与基于数据块局部性的攻击的有效性进行比较[44]。除了使用实验 1（参见7.2.1）中采用的 FSL 数据集之外，本文还加入 MS 数据集（参见6.1.3）用于进行跨数据集的验证与评估。在 MS 数据集中，对于每个系统类别，一次选择两个快照，使用其中一个来推理另一个，以此评估平均的推理率和推理精度。

本文考虑采用以下攻击实例来进行比较。

- `Baseline`：

  实验根据已有工作[44]中建议的参数配置实现基于数据块局部性的攻击。具体来说，它在频率分析的第一次调用中推理出 5 个出现频率最高的的密文数据块-明文数据块对（即，初始化一组用于迭代的密文数据块-明文数据块对），并且在每次后续调用中根据已有数据块对的邻居推理 30 个新的数据块对（即，迭代地推理密文数据块-明文数据块对）。

- `Distribution` **和** $\texttt{Distribution}^{\texttt{S}}$：

  考虑两个基于分布的攻击的攻击实例，分别由 `Distribution` 和 $\texttt{Distribution}^{\texttt{S}}$ 表示，它们分别使用和不使用大小信息作为辅助信息（即上标 S 表示攻击实例使用大小信息进行操作）。实验在与 `Baseline` 相同参数的条件下配置 $\texttt{Distribution}^{\texttt{S}}$ 和 `Distribution`。此外，实验通过以下方式在 $\texttt{Distribution}^{\texttt{S}}$ 和 `Distribution` 中选择参数 r 和 t 的取值：对于 FSL 数据集，我们为所有用户设定 $r = 10$，并分别为 User004、User007、User012、User013、User015 和 User028 设置参数 t = 1.5、1.2、1、1、0.7 和 0.9；对于 MS 数据集，实验继续为所有类别设置参数 $r = 10$，为 Win7 和 Serv-08 设置参数 $t = 2$，为 Vista-U、Serv-03 和 Vista-B 设置参数 $t = 1.6$。以上数据是通过按照实验 1（参见7.2.1）的方法对数据集的最佳配置的测试得出的。

- `Distribution-o` **和** $\texttt{Distribution}^{\texttt{S}}$`-o`：

  实验考虑另外两个基于分布的攻击的实例，用 `Distribution-o` 和 $\texttt{Distribution}^{\texttt{S}}$`-o` 表示，对于 r 和 t 应用与 `Distribution` 和 $\texttt{Distribution}^{\texttt{S}}$ 相同的配置，但进

一步使用更大的参数 u 增加推理的密文数据块-明文数据块对的覆盖范围。具体来说，实验通过以下方式在 Distribution-o 和 Distribution^S-o 中配置参数 u：对于 FSL 数据集，参数 u 的选择与实验 1（参见7.2.1）相同；对于 MS 数据集，实验为 Win7 数据设置 $u = 128$，为 Serv-03、Serv-08、Vista-B 和 Vista-U 设置 $u = 30$。

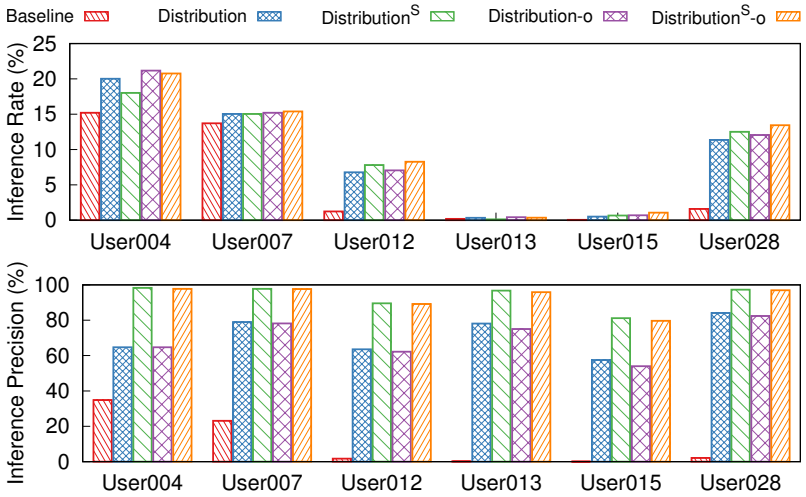

图 7-4 基于分布的攻击实验 2（与已有工作的对比）：比较基于分布的攻击和基于数据块局部性的攻击的有效性（FSL 数据集）。

图7-4给出了基于 FSL 数据集的上述 5 种实例的比较结果。实验观察到，在几乎所有情况下，基于分布的攻击的不同实例都优于基于数据块局部性的攻击。例如，对于 User028，基于分布的攻击的最低推理率是 11.4%，推理精度为 84.1%（Distribution）；此时 Baseline 相应的推理率仅为 1.2%，推理精度仅为 1.7%。这意味着在这种情况下，基于分布的攻击可以将假阳性的数量减少 82.4%。

结论（4）：基于分布的攻击显着提高了推理精度，同时实现了比基于数据块局部性的攻击更高的推理率。

Distribution^S 和 Distribution^S-o 相较于 Distribution 和 Distribution-o 拥有更高的推理精度，这因为它们进一步通过数据块大小信息过滤了误报。例如，对于 User004，Distribution^S 和 Distribution^S-o 相较于 Distribution 和 Distribution-o 将误报分别从 35.2% 减少到 1.7%，从 35.3% 减少到 2.2%。然而，在同样的情况下，实验观察到 Distribution and Distribution-o 的推理率分别为 20.0% 和 21.2%，相较于 Distribution^S 和 Distribution^S-o 分别高出 2.0% 和 0.4%。其原因是 Distribution 和 Distribution-o 推理出了来自不正确的密文数据块-明文数据块对的邻居中的少量正确结果。换句话说，虽然 (C, M) 是不正确的密文数据块-明文数据块对，但 C 的邻居可能以很小的概率对应于 M 的邻居。即使

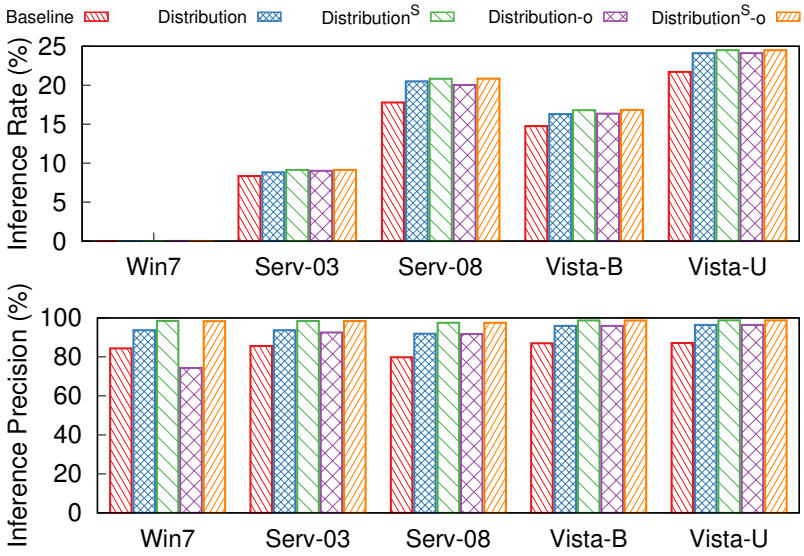

图 7-5 基于分布的攻击实验 2（与已有工作的对比）：比较基于分布的攻击和基于数据块局部性的攻击的有效性（MS 数据集）。

在这种情况下，所有基于分布的攻击实例都比基于数据块局部性的攻击实例效果更佳。具体来说，Baseline 的推理率仅为 15.2%，比基于分布的攻击的最佳情况低 6.0%，最差情况低 2.8%。

结论（5）：过滤不正确的推理结果可以提高推理精度，但会降低推理的密文数据块-明文数据块对的覆盖范围，并可能降低推理率。

我们进一步观察到，虽然 Distribution-o 和 Distribution^S-o 构建在更大的参数 u 上，但它们的推理率仅略高于 Distribution 和 Distribution^S（分别高为 0.4% 和 0.9%）。原因是基于分布的攻击仅通过已推理出数据块对的邻居进行迭代推理。参数 u 的进一步增大只会在结果中添加少量新的正确密文数据块-明文数据块对。

图7-5给出了基于 MS 数据集的上述 5 种实例的比较结果。基于数据块局部性的攻击和基于分布的攻击在大多数 MS 数据集的类别（Win7 除外）中具有高推理率和精度。可能的原因是 MS 数据集中的快照高度相关（例如，数据块总数的方差很小，如表6-3所示）。实验观察到基于分布的攻击仍然优于基于数据块局部性的攻击。例如，在 Vista-U 中，基于分布的攻击的所有推理的推理率和推理精度分别高于 24.1% 和 96.4%，而 Baseline 的推理率和分辨率分别为 21.7% 和 87.1%。

实验中对于数据集中 Win7 类别，基于分布的攻击和基于数据块局部性的攻击的推理率都很低（小于 0.01 %）。其原因是 Win7 包含大量具有唯一性的数据块（大约超过 98.8%，如表6-3所示），导致了攻击效果表现不佳。

7.2.3 实验 3（攻击效果）

实验考虑长期备份模式，并使用 FSL 数据集检查基于分布的攻击的有效性。具体来说，实验选择每个用户的第 i 个 FSL 每周备份作为辅助信息，以推理相应的第 $(i+w)$ 个 FSL 每周备份中的原始明文。显然，w 越小，辅助信息和目标备份之间的相关性就越高。与实验 2（参见7.2.2）配置两个基于分布的攻击实例 `Distribution-o` 和 $\texttt{Distribution}^{\texttt{S}}\texttt{-o}$，并评估它们的推理率和推理精度（针对每个用户的所有可用的 i）。

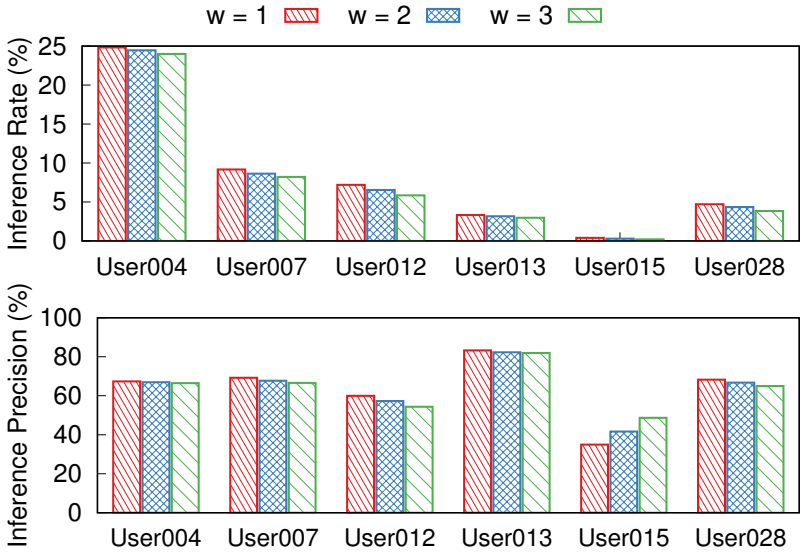

图 7-6 实验 3（攻击效果）：没有大小信息辅助的基于分布的攻击在 FSL 数据集中的有效性。

图7-6给出了 `Distribution-o` 的实验结果。基于分布的攻击在用户之间具有不同的推理率和推理精度。例如，在类似 User004 这样的有利情况下，它实现了推理率 24.8%、推理精度 67.3%（$w=1$）；推理率 24.5%、推理精度 67.0%（$w=2$）；推理率 24.0%、推理精度 66.5%（$w=3$）。在类似 User015 这样的非有利情况下，基于分布的攻击的推理率仅有 0.3%。可能的原因是 User015 的备份数据具有较低的数据块局部性。

此外，实验观察到辅助信息的相关性（即 w）对基于分布的攻击的有效性影响较低。例如，当 w 从 1 增加到 3 时，它仅导致推理率（下降小于 1.4%）和精度（下降小于 5.6%）的有限下降。其原因是基于分布的攻击解决了频率排名中的干扰并保留了攻击的有效性。

结论（6）：基于分布的攻击可以在存在松散相关的辅助信息的帮助下有效控制攻击有效性的下降程度。

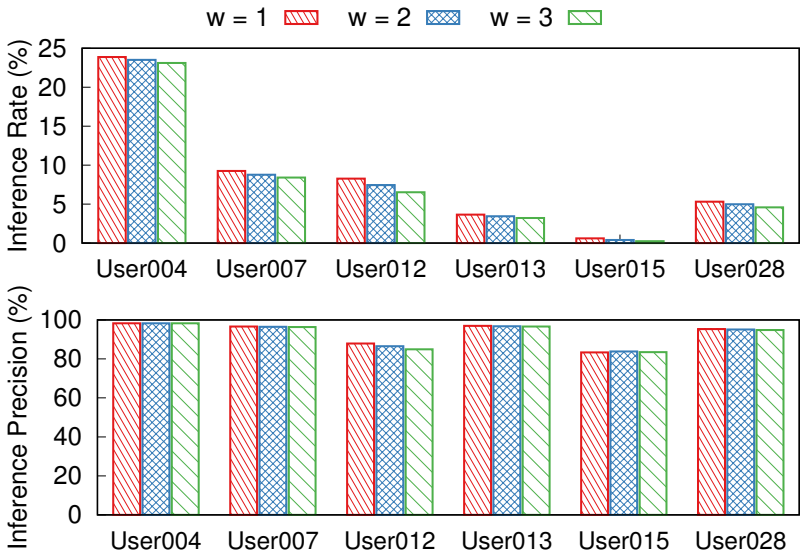

图 7-7 实验 3（攻击效果）：有大小信息辅助的基于分布的攻击在 FSL 数据集中的有效性。

图7-7给出了 $\texttt{Distribution}^{\texttt{s}}$-o 的实验结果。实验观察到它具有与 Distribution-o 相似的推理率，同时拥有更高的推理精度。例如，对于所有用户的平均推理精度分别为 93.1%（$w = 1$）、92.8%（$w = 2$）和 92.4%（$w = 3$）。

7.3 基于聚类的攻击的实验结果

7.3.1 实验 4（参数的影响）

实验首先评估参数 k 的影响，该参数定义了凝聚法层次聚类中组合两个最近聚类的距离上限。本实验使用 FSL 和 VM 数据集来研究 k 如何影响攻击中的基础聚类方案。具体来说，分别对每个选出的 FSL 和 VM 用户数据的最后一次备份应用分段方法，并生成固定大小为 4MB 的数据段。

聚类方案旨在将类似的密文数据段聚合到同一个聚类中，这一过程不会损害每个密文数据段中的数据块的机密性。本文使用聚类接近度量化其影响，将聚类的结果与实际分类方法的结果进行比较（实际分类方法通过最小数据块哈希直接对分段进行分类）。假设我们分别通过实际分类方法生成 m 个数据段的集合，并且通过聚类方案生成 $\tilde{m}$ 个数据段的集合。通过 $\frac{\mathsf{abs}(m-\tilde{m})}{m}$（其中 $\mathsf{abs}(m-\tilde{m})$ 返回 $m-\tilde{m}$ 的绝对值）衡量聚类形成的集合与实际分类形成的集合的相似程度。记集合所有数据段都相同（具有相同的最小数据块哈希）的聚类的数量为 $\hat{m}$。此外，实验还考虑了聚类正确性，通过 $\frac{\hat{m}}{\tilde{m}}$ 进行评估，它量化了对相似数据段的聚类方案的精确程度。

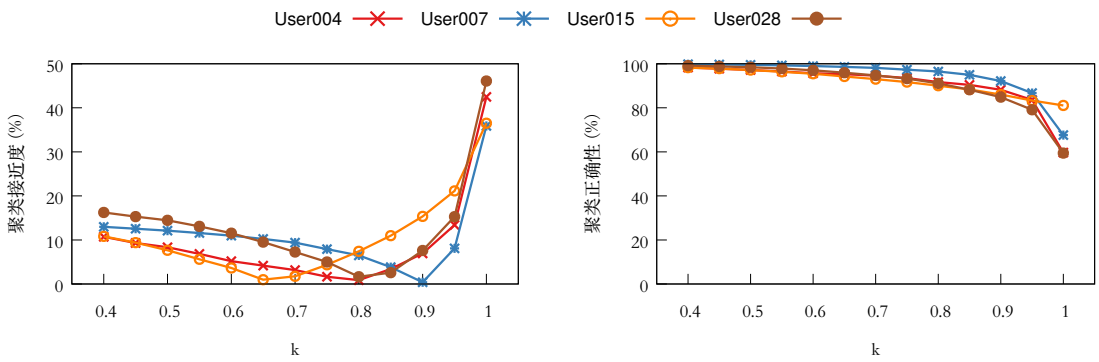

图 7-8 实验 4（参数的影响）：FSL 数据集中参数 k 对基于聚类的攻击的影响（聚类接近度越小越好；聚类正确性越大越好）。

图7-8给出了针对 FSL 数据集的实验结果，其中考虑了四个 FSL 用户的数据（User004，User007，User015 和 User028）以节省评估时间。聚类接近度首先随 k 的增加而增加，这是因为聚类的数量（即 $\tilde{m}$）减少并接近 m。当 k 进一步增加时，聚类数量会从 m 开始下降，并导致聚类间接近度的增加。此外，实验观察到聚类正确性随 k 的增大逐渐减小，这是因为一些非相似的数据段（即，它们的最小块散列不同）被聚合到同一聚类中。这两个结果都表明攻击者可以通过配置一个合适的 k 来平衡聚类的接近程度和正确性的关系。例如，当我们为 User015 设置 $k = 0.65$ 时，相应的聚类接近度和正确性分别为 1.0% 和 94.2%。这意味着聚类方案的结果高度接近实际分类的结果。

结论（7）：通过为聚类方法配置适当的参数 k，攻击可以在未知数据段的最小数据块哈希的前提下得到近似的分类效果。

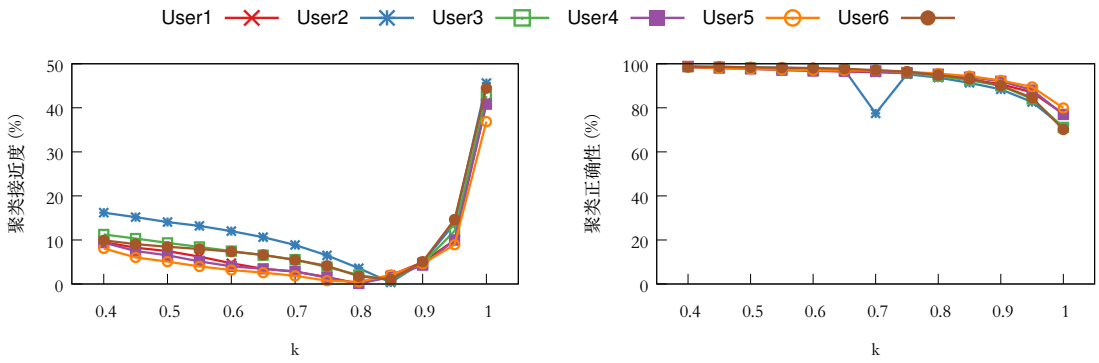

图 7-9 实验 4（参数的影响）：VM 数据集中参数 k 对基于聚类的攻击的影响（聚类接近度越小越好；聚类正确性越大越好）。

图7-9给出了针对 VM 数据集的实验结果。实验观察到所有 VM 数据集中的用

户数据的聚类接近度和正确性与 FSL 数据集中的用户数据具有相似的趋势。当配置参数 $k = 0.8$ 时，所有 VM 数据集中的用户数据的平均聚类接近度为 3.0%，相应的聚类正确性高达 93.1%。

7.3.2 实验 5（攻击效果）

本实验研究基于聚类的攻击的有效性。由于固定大小的数据段的边界移位问题，它对 FSL 数据集的有效性很低（在本实验的测试中大约仅有 1% 的推理率）。因此，本实验使用 VM 数据集来评估其有效性。在基于聚类的攻击中，为聚类方法设置参数 $k = 0.8$，并配置 $(u, r, t) = (5000, 100, 0.5$ 用于将密文聚类与相应的明文聚类相关联。

在本实验的基准测试中，实验发现基于聚类的攻击中的数据块级推理仅正确地推理出数千个数据块（推理率过低，小于 0.01%）。因此，本文重点研究数据段级的推理，它给出了基于聚类的攻击的有效性的下限。为了与数据块级推理效果的测量保持一致，实验基于每个正确推理的数据段中的唯一性的数据块个数来计算推理率和推理精度。

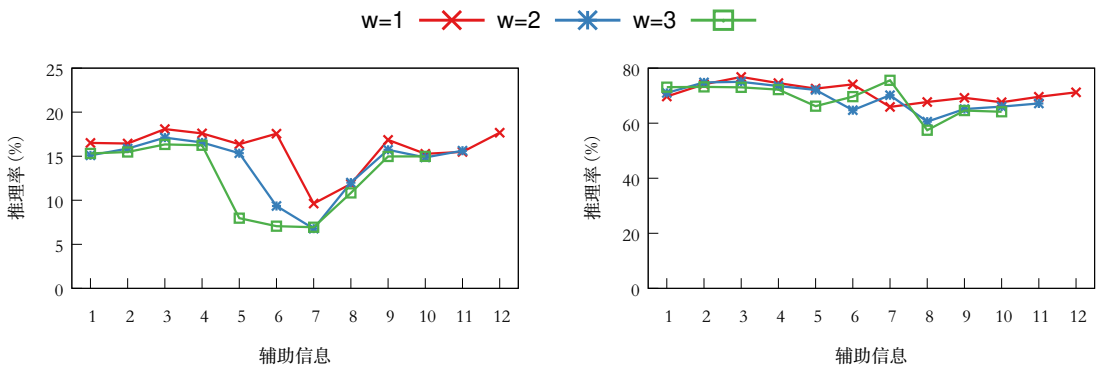

图 7-10 实验 5（攻击有效性）：VM 数据集中基于聚类的攻击的有效性。

本实验采取与实验 3（参见7.2.3）相同的评估方法，实验结果如图7-10所示。具体来说，X 轴描述了 VM 备份数据集中的第 i（其中 $1 \le i \le 12$）个备份用作攻击的辅助信息，而 y 轴表示该辅助信息对 VM 数据集中第 $(i + w)$（其中 $w = 1, 2, 3$）个备份的平均推理效果。实验观察到推理率和推理精度都有显著波动。例如，当使用第 3 个备份作为辅助信息时，攻击的推理率为 18.1%，推理精度为 76.8%（$w = 1$）；推理率为 17.1%，推理精度为 75.0%（$w = 2$）；推理率为 16.3%，推理精度为 73.0%（$w = 3$）。但当使用第 7 个备份作为辅助信息时，相应的推理率和推理精度分别下降到 9.6% 和 65.9%；6.8% 和 71.2%；以及 6.9% 和 75.6%。原因是 VM 数据集中的备份在第 7 周之后有大量更新，显著减少了相邻备份间的相关性。平均而言，对

于 $w = 1, 2, 3$ 时，基于聚类的攻击推理出 15.8%，14.0% 和 12.6% 密文数据块-明文数据块对，对应的推理精度为 71.1%，69.1%，和 68.9%。

7.4 攻击对安全影响的结果

到目前为止，本文通过量化正确推理的密文数据块-明文数据块对来检查推理攻击的有效性。然而，这些结果所带来的安全隐患以及频率分析攻击如何带来实际损害仍然是一个悬而未决的问题。在下面的实验中，本文根据原始推理率评估攻击的安全隐患，原始推理率定义为受正确推理的数据块影响的原始数据块的百分比。

7.4.1 实验 6（安全隐患分析）

实验首先考虑基于分布的攻击，并针对不同类型的文件评估其原始推理率。这一部分使用 FSL 数据集来进行评估，因为只有 FSL 数据集包含数据块所属的文件的文件元数据（包含文件的扩展名）。具体来说，本实验关注具有特定扩展名的五种类型的文件（参见表7-1）：office 文件，图片文件，程序源代码文件，数据库文件和磁盘映像文件。这些文件占 FSL 数据集原始内容的 60% 以上。

表 7-1 实验 6（安全隐患分析）：基于分布的攻击

文件类型	扩展名	文件大小范围	原始推理率	
			Distribution-o	Distribution^{S}-o
Office 文件	doc(x), ppt(x), xls(x)	10KB-1MB	4.9%	5.3%
图片文件	jpg, png	10-100KB	7.7%	6.7%
程序源代码	c, h, cpp, java, py	10-20KB	17.1%	15.0%
数据库文件	db, po	20-700KB	2.6%	2.4%
磁盘映像文件	vmdk, img	200MB-1GB	15.8%	16.7%

这一部分应用实验 2（参见7.2.2）提出的方法的方法，用 Distribution^{S}-o 和 Distribution-o 来评估机遇分布的攻击的原始推理率。表7-1给出了实验结果。两种攻击实例针对磁盘映像文件都具有较高的原始推理率（例如，Distribution-o 为 15.8%，Distribution^{S}-o 为 16.7%），这因为每个磁盘文件中包含大量很少进行更新的数据块，这些数据块在同一文件中具有很强的数据块局部性。有趣的是，实验观察到程序源代码文件中，尽管每个文件都很小，但基于分布的攻击也具有很高的原始推理率（例如，Distribution-o 为 17.1%，Distribution^{S}-o 为 15.0%）。其原因是程序源代码文件通常一起存储在文件系统中（例如，属于同一项目的源文件位于相同的目录中）并形成大范围的相关数据块，在文件之间具有很强的数据块局

部性。对于小型或分散的文件（例如，Office 文件，图片文件和数据库文件），基于分布的攻击的原始推理率相对较低。

结论（8）：基于分布的攻击的有效性取决于目标文件的更新频率，文件大小和存储结构。

这一部分使用 VM 数据集评估基于聚类的攻击带来的安全隐患。具体来说，本实验使用每个 VM 用户数据的第 11 个备份来推理该用户相应的第 13 个中的原始数据内容。由于 VM 数据集不包含任何元数据，因此本实验根据每个 VM 快照中的所有数据块计算原始推理率。请注意，本实验在原始推理率的计数中过滤所有全零数据块，因为它们在 VM 磁盘映像中占有很大比例[53]，且会对评估的结果产生误导（导致原始推理率结果虚假偏高）。

表 7-2 实验 6（安全隐患分析）：基于聚类的攻击

用户 ID	原始推理率	用户 ID	原始推理率
1	13.2%	2	22.4%
3	25.7%	4	28.4%
5	18.5%	6	30.8%

这一部分使用与实验 5（参见7.3.2）相同的配置，并在通过数据段级推理来评估原始推理率。表7-2给出了不同用户数据的原始推理率结果。实验观察到基于聚类的攻击对 VM 数据集实现了很高的安全威胁。例如，它可以推理出 User6 的 VM 备份中高达 30.8% 的原始内容。所有用户的平均原始推理率高达 23.2%。

结论（9）：基于聚类的攻击严重威胁到 VM 磁盘映像的机密性。

7.5 本章小结

本章通过 6 个实验对基于分布的攻击和基于聚类的攻击的相关参数的影响、攻击的有效性以及两种攻击手段对真实世界数据集的安全威胁进行了验证与分析，提出了 9 个根据实验观察得出的结论，证明了两种攻击手段的作用和价值。

第八章 应对频率分析攻击的对策讨论

在本章中，将讨论应对本文攻击方案使用的三种泄漏的对策及其优缺点。但在三中泄漏中，单纯的防范数据块大小信息泄漏并不足以使本文提出的攻击手段失效（大小信息的利用只是攻击用于进一步提高推理攻击准确性的可选条件）。

8.1 防止频率泄漏

MinHash 加密[28,44]使用从一组相邻数据块上的最小块哈希导出的密钥加密该组内每个明文数据块，因此可能将相同的明文数据块加密映射到不同的密文数据块。防御频率信息泄漏的基本原理是通过 MinHash 加密改变各个密文数据块的频率，并由此扰乱不同数据块的频率排名。

与此同时，MinHash 加密还可以用于防止本文中的频率分析攻击。MinHash 加密是非确定性的加密方法，使用该方法会改变数据块的频率分布，因此本文的的攻击目标是确定性加密（例如，MLE[10]）。但 MinHash 加密因为打破了明文数据块-密文数据块之间的一一映射关系，所以在加密重复数据删除中导致了存储效率的下降（重复数据删除所能节省的存储空间变小）。此外，因为 MinHash 加密的随机性主要取决于目标工作负载中的最小块哈希，所以其不是一种扰乱数据块频率的主动对策，在不同的工作负载中能实现的扰乱效果具有较大的差异。

最近的一项工作[78]提出通过有计划的添加冗余（重复）的数据块来防止针对客户端加密重复数据删除的流量分析攻击。该方法[78]在运作中会改变数据块的频率，因此可以用于防止频率分析攻击。与 MinHash 加密相比，该方法[78]只是添加了重复的数据块，因此不会降低存储效率。但另一方面，它需要使用在特定的假设之下。首先它要求特定的数据块是重复的，其次它只适用于客户端重复数据删除。因此使用这种方法可能会引入额外的泄漏通道（参见第九章）。

现有工作提出了几个扩展的 MLE 实例基于强加密原语构建，以防止加密重复数据删除的频率泄漏，例如支持等式测试的随机加密[29]，混合加密[30]，以及与完全同态加密的交互[31]。它们都提供了可证明的安全性，但它们如何在实践中实施和部署仍未有进一步的研究。

8.2 防止顺序泄漏

一个简单的对策是干扰数据块在重复数据删除处理中顺序。例如，加密重复数据删除可以在加密之前对明文数据块流进行额外的顺序扰动添加处理，以隐藏

每个明文数据块的真实逻辑顺序。这种方法可以有效的防止基于分布的频率分析攻击（参见第四章），因为攻击者无法正确识别到数据块邻居信息。但它对于基于聚类的频率分析攻击（参见第五章）并不是完全有效的。如果添加扰动仅在很小的范围内进行（例如，每个数据段中的明文数据块的顺序被添加扰动），则基于聚类的频率分析攻击方法仍然有效。

顺序扰乱策略的最大缺陷在于它破坏了数据块的局部性并且在大规模的重复数据删除中会导致性能的严重下降[45–47]。

8.3 防止大小泄漏

正如已有的工作[50]所建议的那样，加密重复数据删除可以在每个明文数据块种填充额外的数据来混淆这个明文数据块的实际大小。但是，这种填充方案的实现存在很大的难度，因为它需要在相同的明文数据块中添加相同的冗余数据；否则加密重复数据删除存储系统无法正常检测数据块是否重复。一种可能的解决方案建立在 MLE[10,20] 的范例之上，考虑将每个明文数据块的加密哈希作为计算种子，并使用它来生成可变大小的伪随机数据用于明文数据块的填充。与 MLE 一样，这种解决方案的代价是需要使用服务器辅助方法[20]来抵御暴力攻击（参见第九章）。

另一种思路是使用固定大小的数据块分块方法来取代可变大小的数据块分块方法。在这种情况下，由于所有的数据块都具有相同的大小，因此攻击者无法利用大小信息来区分它们。虽然固定大小的分块会受到边界偏移的影响，但它在 VM 磁盘映像这一特定数据类型中实现了与可变大小数据块分块方案几乎相同的重复数据删除存储空间节省效果[53]。因此，本文建议在某些特定的加密重复数据删除工作负载（例如，VM 磁盘映像）中应用固定大小的数据块分块方法，以防止数据块大小信息泄漏。

8.4 本章小结

本章给出了应对三种类型泄漏带来的风险的方案，并比较了现有方案的优缺点。

第九章 相关工作

9.1 MLE 的应用

回顾第二章，MLE[10] 正式确立了加密重复数据删除的加密方案基础。第一个发布的 MLE 实例是收敛加密（CE）[11]，它使用明文的加密哈希及其对应的密文作为 MLE 密钥和标签。其他 CE 的变体包括：散列收敛加密（HCE）[10]，它从明文中导出标记，同时仍然使用明文的哈希作为 MLE 密钥；随机收敛加密（RCE）[10]，用新的随机密钥加密明文以形成非确定性密文，通过从明文的哈希中导出的 MLE 密钥保护随机密钥，并附加来自明文的确定性标签以实现数据块重复性的检查；收敛扩散加密（CD）[21]，它通过使用明文的加密哈希作为秘密共享算法的随机种子，将 CE 扩展到秘密共享。由于所有上述实例仅从明文中获取 MLE 密钥和（或）标签，如果明文是可预测的（即，所有可能的明文的总数量有限），它们很容易受到离线暴力攻击[20] 的影响，因为攻击者可以从所有可能的明文中穷举地导出 MLE 密钥和标签，并检查是否存在某种明文被加密到目标密文（在 CE，HCE 和 CD 中）或映射到目标标签（在 RCE 中）。

暴力攻击已经被证明可以用来获取文件信息[79]。为了防止离线暴力攻击，DupLESS[20] 通过在独立密钥服务器中管理 MLE 密钥来实现服务器辅助 MLE，从而确保无法从离线消息中获得每个 MLE 密钥。DupLESS 采用两种机制来实现强大的 MLE 密钥管理：

1. **无关密钥生成：**
   DupLESS 中客户端总是在不向密钥服务器显示消息内容的前提下从密钥服务器获得确定性 MLE 密钥。
2. **密钥生成频率限制：**
   DupLESS 中的密钥服务器通过限制来自客户端的密钥生成请求频率，防止在线暴力攻击。

其他研究扩展了服务器辅助 MLE 的各个方面，例如可靠的密钥管理[22]，透明定价[23]，点对点密钥管理[24] 和密钥转换（所属权变更）[28]。但是服务器辅助 MLE 仍然建立在确定性加密的基础之上，现有的 MLE 实例（基于 CE 或服务器辅助 MLE）都容易受到本文研究的推理攻击方法的攻击。

9.2 对（加密）重复数据删除的攻击

除了离线暴力攻击之外，之前的研究还考虑了针对重复数据删除存储的各种攻击，此类攻击通常也适用于加密重复数据删除。例如，侧信道攻击[9,48]使攻击者可以利用重复数据删除的运作模式来推理目标用户上传文件的具体内容，或者在客户端重复数据删除中获取未经授权的访问权限；例如，2010 年针对“Dropbox”成功发起了侧通道攻击（以及其他相关攻击）[49]。重复伪造攻击[10]通过利用不一致的标签破坏了消息的完整性。Ritzdorf 等人在其工作中利用数据块大小的泄漏来推理文件的存在性[50]。基于数据块局部性的攻击[44]利用频率分析来推理密文数据块-明文数据块对。本文的工作在推理攻击的已有工作[44,50]，通过利用各种类型的泄漏给出了针对加密重复数据删除的推理攻击的更深入研究。

9.3 防御机制

在第八章中讨论了针对加密重复数据删除的频率，顺序和大小泄漏的应对对策。而其他防御机制旨在防范其他类型的攻击。

如上所述，服务器辅助的 MLE[20]可以抵御离线暴力攻击。服务器端重复数据删除[9,21,80]和所有权证明[48,59,60]可以抵御侧信道攻击。服务器端标签生成[11,20]和受保护的解密[10]可以抵御重复检查攻击。

9.4 推理攻击

已有工作已经提出了几种针对加密数据库[12,39–43]和关键字搜索[34–38]的推理攻击方案。它们都利用确定性加密的性质来识别不同类型的泄漏。本文的研究内容与它们的不同之处在于专门研究了针对加密重复数据删除的频率分析推理攻击。

9.5 本章小结

本章介绍了现有加密重复数据删除中的各种安全机制与攻击手段，并分析了本文的工作与已有工作的不同之处。

第十章 结束语

10.1 本文总结

加密重复数据删除应用确定性加密，并由此泄漏了明文的频率。本文重新审视了频率分析引起的安全漏洞，并证明加密重复数据删除更容易受到推理攻击。本文提出了两种新的频率分析攻击方法，它们在攻击者所具有的条件的不同假设下都能实现高推理率和高推理精度。本文用三个真实世界的数据集来验证评估这两种攻击方法，提出关于其性质的各种新观察，并进一步分析它们如何带来实际性的损害。本文还讨论了加密重复数据删除应对频率分析攻击的可能的对策及其相应的优缺点，以建议从业者安全地实现和部署加密重复数据删除存储系统。

10.2 下一步学习工作方向

本文为未来的工作提出三个方向。

首先，本文将完整的旧备份视为辅助信息，如果攻击者仅对旧备份的部分有了解（没有完整的旧备份用作辅助信息），则不研究攻击者如何发起攻击。可能的是，攻击者仍然可以应用频率分析攻击从可用的部分备份中提取特征，并通过比较特征与目标备份中的特征来推理密文数据块-明文数据块对。未来研究的第一个方向是设计更高级的推理攻击方法，并比较在前述的部分知识案例中新方法是否比直接应用本文提出的两种攻击更好。

其次，本文根据频率分析攻击的有效性调整频率分析攻击的参数，这种调整只有在攻击发生后才能学习。本文没有研究如何预先从辅助信息中推导出最佳参数。未来研究的第二个方向是在预先通过辅助信息推导最佳攻击参数方面尝试作出改进。

第三，本文不对实际的加密重复数据删除存储系统实现攻击原型。未来研究的第三个方向是在实际系统中部署本文提出的攻击设计以及报告加密重复数据删除在实践中存在的漏洞。

致 谢

本科阶段四年的求学生涯即将落下帷幕，在老师、同学的支持与帮助下，本科这四年收获颇丰。本研究及论文是在我的导师李经纬的亲切关怀和耐心的指导下完成的。在此，我谨向李经纬老师至以万分诚挚的谢意。

参考文献

[1] Gartner. Gartner forecasts 59 percent mobile data growth worldwide in 2015[EB/OL]. http://www.gartner.com/newsroom/id/3098617, Dec 12, 2018

[2] 敖莉, 舒继武, 李明强. 重复数据删除技术 [J]. 软件学报, 2010, 21(5): 916-929

[3] J. McKnight, T. Asaro, B. Babineau. Digital archiving: End-user survey and market forecast 2006–2010[J]. The Enterprise Strategy Group, 2006,

[4] 付印金, 肖侬, 刘芳. 重复数据删除关键技术研究进展 [J]. 计算机研究与发展, 2012, 49(1): 12-20

[5] EMC. Emc data domain[EB/OL]. http://www.emc.com/en-us/data-protection/data-domain.htm, Dec 12, 2018

[6] EMC. Avamar deduplication backup software and system[EB/OL]. http://www.emc.com/data-protection/avamar.htm, Dec 12, 2018

[7] veritas. Netbackup appliances[EB/OL]. https://www.veritas.com/product/backup-and-recovery/netbackup-appliances.html, Dec 12, 2018

[8] CommVault. Commvault solutions for data protection backup and recovery[EB/OL]. http://www.commvault.com/solutions/by-function/data-protection-backup-and-recovery, Dec 12, 2018

[9] D. Harnik, B. Pinkas, A. Shulman-Peleg. Side channels in cloud services: Deduplication in cloud storage[J]. IEEE Security & Privacy, 2010, 6): 40-47

[10] M. Bellare, S. Keelveedhi, T. Ristenpart. Message-locked encryption and secure deduplication[C]. Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2013, 296-312

[11] J. R. Douceur, A. Adya, W. J. Bolosky, et al. Reclaiming space from duplicate files in a serverless distributed file system[C]. Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on, 2002, 617-624

[12] M. Naveed, S. Kamara, C. V. Wright. Inference attacks on property-preserving encrypted databases[C]. Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, 644-655

[13] MEGA. Mega[EB/OL]. https://mega.nz, Dec 12, 2018

[14] ElephantDrive. Elephantdrive[EB/OL]. https://www.elephantdrive.com, Dec 12, 2018

[15] Cryptosphere. Cryptosphere[EB/OL]. https://cryptosphere.io, Dec 12, 2018

[16] Freenet. Freenet[EB/OL]. https://freenetproject.org, Dec 12, 2018

[17] GNU. Gnu's framework for secure peer-to-peer networking[EB/OL]. https://gnunet.org, Dec 12, 2018

[18] Tahoe-LAFS. Tahoe-lafs[EB/OL]. https://tahoe-lafs.org/trac/tahoe-lafs., Dec 12, 2018

[19] M. Li, C. Qin, J. Li, et al. Cdstore: Toward reliable, secure, and cost-efficient cloud storage via convergent dispersal[J]. IEEE Internet Computing, 2016, 3): 45-53

[20] S. Keelveedhi, M. Bellare, T. Ristenpart. Dupless: server-aided encryption for deduplicated storage[C]. USENIX Security 13, 2013, 179-194

[21] M. Li, C. Qin, P. P. Lee. Cdstore: Toward reliable, secure, and cost-efficient cloud storage via convergent dispersal.[C]. USENIX Annual Technical Conference, 2015, 111-124

[22] Y. Duan. Distributed key generation for encrypted deduplication: Achieving the strongest privacy[C]. Proceedings of the 6th edition of the ACM Workshop on Cloud Computing Security, 2014, 57-68

[23] F. Armknecht, J.-M. Bohli, G. O. Karame, et al. Transparent data deduplication in the cloud[C]. Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, 886-900

[24] J. Liu, N. Asokan, B. Pinkas. Secure deduplication of encrypted data without additional independent servers[C]. Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, 874-885

[25] Y. Zhou, D. Feng, W. Xia, et al. Secdep: A user-aware efficient fine-grained secure deduplication scheme with multi-level key management[C]. Mass Storage Systems and Technologies (MSST), 2015 31st Symposium on, 2015, 1-14

[26] J. Li, J. Li, D. Xie, et al. Secure auditing and deduplicating data in cloud[J]. IEEE Transactions on Computers, 2016, 65(8): 2386-2396

[27] J. Li, C. Qin, P. P. Lee, et al. Rekeying for encrypted deduplication storage[C]. Dependable Systems and Networks (DSN), 2016 46th Annual IEEE/IFIP International Conference on, 2016, 618-629

[28] C. Qin, J. Li, P. P. Lee. The design and implementation of a rekeying-aware encrypted deduplication storage system[J]. ACM Transactions on Storage (TOS), 2017, 13(1): 9

[29] M. Abadi, D. Boneh, I. Mironov, et al. Message-locked encryption for lock-dependent messages[M]. Springer, 2013, 374-391

[30] J. Stanek, A. Sorniotti, E. Androulaki, et al. A secure data deduplication scheme for cloud storage[C]. International Conference on Financial Cryptography and Data Security, 2014, 99-118

[31] M. Bellare, S. Keelveedhi. Interactive message-locked encryption and secure deduplication[C]. IACR International Workshop on Public Key Cryptography, 2015, 516-538

[32] J. Katz, A. J. Menezes, P. C. Van Oorschot, et al. Handbook of applied cryptography[M]. CRC press, 1996

[33] R. Kumar, J. Novak, B. Pang, et al. On anonymizing query logs via token-based hashing[C]. Proceedings of the 16th international conference on World Wide Web, 2007, 629-638

[34] D. Cash, P. Grubbs, J. Perry, et al. Leakage-abuse attacks against searchable encryption[C]. Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, 2015, 668-679

[35] P. Grubbs, R. McPherson, M. Naveed, et al. Breaking web applications built on top of encrypted data[C]. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016, 1353-1364

[36] M. S. Islam, M. Kuzu, M. Kantarcioglu. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation.[C]. Ndss, 2012, 12

[37] D. Pouliot, C. V. Wright. The shadow nemesis: Inference attacks on efficiently deployable, efficiently searchable encryption[C]. Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, 2016, 1341-1352

[38] Y. Zhang, J. Katz, C. Papamanthou. All your queries are belong to us: The power of file-injection attacks on searchable encryption.[C]. USENIX Security Symposium, 2016, 707-720

[39] V. Bindschaedler, P. Grubbs, D. Cash, et al. The tao of inference in privacy-protected databases[J]. Proceedings of the VLDB Endowment, 2018, 11(11): 1715-1728

[40] F. B. Durak, T. M. DuBuisson, D. Cash. What else is revealed by order-revealing encryption?[C]. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016, 1155-1166

[41] P. Grubbs, K. Sekniqi, V. Bindschaedler, et al. Leakage-abuse attacks against order-revealing encryption[C]. Security and Privacy (SP), 2017 IEEE Symposium on, 2017, 655-672

[42] G. Kellaris, G. Kollios, K. Nissim, et al. Generic attacks on secure outsourced databases[C]. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016, 1329-1340

[43] M.-S. Lacharité, B. Minaud, K. G. Paterson. Improved reconstruction attacks on encrypted data using range query leakage[C]. 2018 IEEE Symposium on Security and Privacy (SP), 2018, 297-314

[44] J. Li, C. Qin, P. P. Lee, et al. Information leakage in encrypted deduplication via frequency analysis[C]. Dependable Systems and Networks (DSN), 2017 47th Annual IEEE/IFIP International Conference on, 2017, 1-12

[45] B. Zhu, K. Li, R. H. Patterson. Avoiding the disk bottleneck in the data domain deduplication file system.[C]. Fast, 2008, 1-14

[46] M. Lillibridge, K. Eshghi, D. Bhagwat, et al. Sparse indexing: Large scale, inline deduplication using sampling and locality.[C]. Fast, 2009, 111-123

[47] W. Xia, H. Jiang, D. Feng, et al. Silo: A similarity-locality based near-exact deduplication scheme with low ram overhead and high throughput.[C]. USENIX annual technical conference, 2011, 26-30

[48] S. Halevi, D. Harnik, B. Pinkas, et al. Proofs of ownership in remote storage systems[C]. Proceedings of the 18th ACM conference on Computer and communications security, 2011, 491-500

[49] M. Mulazzani, S. Schrittwieser, M. Leithner, et al. Dark clouds on the horizon: Using cloud storage as attack vector and online slack space.[C]. USENIX security symposium, 2011, 65-76

[50] H. Ritzdorf, G. Karame, C. Soriente, et al. On information leakage in deduplicated storage systems[C]. Proceedings of the 2016 ACM on Cloud Computing Security Workshop, 2016, 61-72

[51] J. Black. Compare-by-hash: A reasoned analysis.[C]. USENIX Annual Technical Conference, General Track, 2006, 85-90

[52] M. O. Rabin. Fingerprinting by random polynomials[J]. Technical report, 1981,

[53] K. Jin, E. L. Miller. The effectiveness of deduplication on virtual machine disk images[C]. Proceedings of SYSTOR 2009: The Israeli Experimental Systems Conference, 2009, 7

[54] A. Adya, W. J. Bolosky, M. Castro, et al. Farsite: Federated, available, and reliable storage for an incompletely trusted environment[J]. ACM SIGOPS Operating Systems Review, 2002, 36(SI): 1-14

[55] L. P. Cox, C. D. Murray, B. D. Noble. Pastiche: Making backup cheap and easy[J]. ACM SIGOPS Operating Systems Review, 2002, 36(SI): 285-298

[56] M. W. Storer, K. Greenan, D. D. Long, et al. Secure data deduplication[C]. Proceedings of the 4th ACM international workshop on Storage security and survivability, 2008, 1-10

[57] P. Anderson, L. Zhang. Fast and secure laptop backups with encrypted de-duplication.[C]. LISA, 2010,

[58] Z. Wilcox-O'Hearn, B. Warner. Tahoe: the least-authority filesystem[C]. Proceedings of the 4th ACM international workshop on Storage security and survivability, 2008, 21-26

[59] J. Xu, E.-C. Chang, J. Zhou. 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, 2013, 195-206

[60] R. Di Pietro, A. Sorniotti. Boosting efficiency and security in proof of ownership for deduplication[C]. Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, 2012, 81-82

[61] A. Juels, B. S. Kaliski Jr. Pors: Proofs of retrievability for large files[C]. Proceedings of the 14th ACM conference on Computer and communications security, 2007, 584-597

[62] G. Ateniese, R. Burns, R. Curtmola, et al. Provable data possession at untrusted stores[C]. Proceedings of the 14th ACM conference on Computer and communications security, 2007, 598-609

[63] TechCrunch. Aol proudly releases massive amounts of private data[EB/OL]. TechCrunch:http://www.techcrunch.com/2006/08/06/aol-proudly-releasesmassive-amounts-of-user-search-data, 2006

[64] www.healthitoutcomes.com. Lost devices and healthcare data leakage, two sides of the same coin?[EB/OL]. https://www.healthitoutcomes.com/doc/lost-devices-and-healthcare-data-leakage-two-sides-of-the-same-coin-0001, 2016

[65] www.eweek.com. Cloud data leak exposes information on 123 million americans[EB/OL]. http://www.eweek.com/security/cloud-data-leak-exposes-information-on-123-million-americans, 2018

[66] I. A. Al-Kadit. Origins of cryptology: The arab contributions[J]. Cryptologia, 1992, 16(2): 97-126

[67] A. Z. Broder, M. Charikar, A. M. Frieze, et al. Min-wise independent permutations[J]. Journal of Computer and System Sciences, 2000, 60(3): 630-659

[68] A. Z. Broder. On the resemblance and containment of documents[C]. Compression and complexity of sequences 1997. proceedings, 1997, 21-29

[69] D. Bhagwat, K. Eshghi, D. D. Long, et al. Extreme binning: Scalable, parallel deduplication for chunk-based file backup[C]. Modeling, Analysis & Simulation of Computer and Telecommunication Systems, 2009. MASCOTS'09. IEEE International Symposium on, 2009, 1-9

[70] S. C. Johnson. Hierarchical clustering schemes[J]. Psychometrika, 1967, 32(3): 241-254

[71] Z. Sun, G. Kuenning, S. Mandal, et al. A long-term user-centric analysis of deduplication patterns[C]. Mass Storage Systems and Technologies (MSST), 2016 32nd Symposium on, 2016, 1-7

[72] FSL. FSL traces and snapshots public archive[EB/OL]. http://tracer.filesystems.org/, Dec 12, 2018

[73] Z. Sun, G. Kuenning, S. Mandal, et al. Cluster and single-node analysis of long-term deduplication patterns[J]. ACM Transactions on Storage (TOS), 2018, 14(2): 13

[74] D. T. Meyer, W. J. Bolosky. A study of practical deduplication[J]. ACM Transactions on Storage (TOS), 2012, 7(4): 14

[75] Microsoft. Microsoft traces and snapshots public archive[EB/OL]. http://iotta.snia.org/tracetypes/6, 2018

[76] OpenDedup. SDFS: Opensource dedup to cloud and local storage[EB/OL]. http://opendedup.org/odd/documentation/, Dec 12, 2018

[77] M. Fu, D. Feng, Y. Hua, et al. Design tradeoffs for data deduplication performance in backup workloads.[C]. FAST, 2015, 331-344

[78] P. Zuo, Y. Hua, C. Wang, et al. Mitigating traffic-based side channel attacks in bandwidth-efficient cloud storage[C]. 2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2018, 1153-1162

[79] Z. Wilcox-O'Hearn. Drew perttula and attacks on convergent encryption[EB/OL]. https://tahoe-lafs.org/hacktahoelafs/drew_perttula.html, 2008

[80] F. Armknecht, C. Boyd, G. T. Davies, et al. Side channels in deduplication: trade-offs between leakage and efficiency[C]. Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, 2017, 266-274

[81] M. Fu. Destor: a platform for data deduplication evaluation[EB/OL]. https://github.com/fomy/destor, Dec 12, 2018

附　录

外文资料原文

翻译的文献内容是 2017 年 ASIA CCS 会议上由 Armknecht, F., Boyd, C., Davies, G. T., Gjsteen, K., & Toorani, M. 完成的 Side Channels in Deduplication: Trade-offs between Leakage and Efficiency 的引言部分，该部分的电子版原始内容截图如下

Side Channels in Deduplication: Trade-offs between Leakage and Efficiency

Frederik Armknecht
University of Mannheim
Mannheim, Germany
armknecht@uni-mannheim.de

Colin Boyd
Norwegian University of Science and Technology, NTNU
Trondheim, Norway
colin.boyd@item.ntnu.no

Gareth T. Davies
Norwegian University of Science and Technology, NTNU
Trondheim, Norway
gareth.davies@ntnu.no

Kristian Gjøsteen
Norwegian University of Science and Technology, NTNU
Trondheim, Norway
kristian.gjosteen@math.ntnu.no

Mohsen Toorani
University of Bergen
Bergen, Norway
mohsen.toorani@uib.no

ABSTRACT

Deduplication removes redundant copies of files or data blocks stored on the cloud. Client-side deduplication, where the client only uploads the file upon the request of the server, provides major storage and bandwidth savings, but introduces a number of security concerns. Harnik et al. (2010) showed how cross-user client-side deduplication inherently gives the adversary access to a (noisy) side-channel that may divulge whether or not a particular file is stored on the server, leading to leakage of user information. We provide formal definitions for deduplication strategies and their security in terms of adversarial advantage. Using these definitions, we provide a criterion for designing good strategies and then prove a bound characterizing the necessary trade-off between security and efficiency.

1. INTRODUCTION

Deduplication is a process used by many cloud storage providers and services to remove redundant copies of data stored in the cloud. It has been shown [12, 14] to greatly reduce storage requirements in practice because users, both individuals and corporations, often store identical or similar content. Deduplication can take place either at the server-side or at the client-side. In *server-side deduplication*, the server checks whether a file uploaded by a client has already been stored. If so, the server does not store it again but instead records the ownership by the client and allows the client to access the shared file using a suitable index. Server-side deduplication achieves the aim of reducing storage but still requires the client to upload each file it wishes to store. In *client-side deduplication*, a user wishing to upload a file first checks whether the file is already stored in the cloud, for example by sending a hash of the file to the server which checks against its list of stored file hashes. If the file is already stored then the file is not sent by the client, but the server allows the client access to the shared file as before. Thus client-side deduplication greatly reduces the bandwidth requirements in cloud storage in addition to reducing storage requirements. Since communication costs can be high in comparison with storage costs, client-side deduplication is generally preferable to server-side deduplication on economic grounds. Deduplication can take place either with respect to files or with respect to blocks, but we will not be concerned with this difference since many of the attacks and countermeasures considered in this paper can be applied to either approach.

ASIA CCS '17, April 02-06, 2017, Abu Dhabi, United Arab Emirates

DOI: http://dx.doi.org/10.1145/3052973.3053019

Secure Deduplication.

Despite the great saving in storage and bandwidth, deduplication causes at least two major security and privacy problems, and this has led to extensive recent work on *secure deduplication* [14]. The first problem is that deduplication cannot take place if semantically secure end-to-end encryption is deployed. Under ciphertext indistinguishability the cloud service provider (CSP), which does not possess the decryption key, would be unable to determine if two ciphertexts correspond to the same plaintext. Several alternative forms of encryption have been proposed in order to address this problem [1,3,5,8,9,18], deriving the encryption key from the file itself in various ways. These works often make strong assumptions regarding file unpredictability or key distribution. The second problem is that client-side deduplication can work as a side channel leaking information under different attacks [7]. This paper focuses on these side-channel attacks.

Harnik et al. [7] identified three attacks due to side-channels in client-side deduplication. The attacks apply to the cross-user scenario where different users who upload the same file will have their data deduplicated. The basic idea of all the attacks is that one user can obtain information about an-

other user's file by receiving a signal revealing whether or not the file was previously uploaded. In one example, sometimes called the *salary attack*, the adversary attempts to learn private data of Alice (her salary) in her employment contract which she has stored with the CSP. The adversary inserts guesses on a template file and uploads it to the CSP where the corresponding file of Alice resides. Occurrence of the deduplication signal will then allow the adversary to infer correctness of the guess.

Having identified these side-channel attacks, Harnik et al. [7] proposed a countermeasure in which the signal on whether a file is already uploaded is hidden by randomization. More specifically, for each file a threshold is chosen uniformly at random and the user is only informed not to upload the file if the number of previous uploads meets or exceeds the threshold. This will obviously increase the required bandwidth compared to basic client-side deduplication. The side-channel does not occur in server-side deduplication because then the client always uploads the file and allows the server to decide whether or not to deduplicate. The random threshold countermeasure can thus be seen as a compromise between the efficiency of client-side deduplicaton and the security of server-side deduplication.

Contributions.

Although the mitigation idea of Harnik et al. [7] has been discussed and developed in the literature [10, 17], there has been no formal modeling and analysis of threshold-based solutions for defending side-channel attacks. This has prevented any opportunity to formally compare the effectiveness of different solutions. The purpose of this paper is to remedy this situation by:

- providing formal definitions for side-channel deduplication strategies, including a natural measure for effectiveness of countermeasures;
- identifying the conditions required for strategies to optimize bandwidth and security;
- characterizing the trade-off between security and efficiency necessary for all strategies;
- showing that the original proposal of Harnik et al. [7] provides an optimal defence within one natural security measure.

There are other scenarios in which similar kinds of side-channels are available to attackers. In independent and concurrent work, Ritzdorf et al. [16] consider the information leaked to a curious cloud provider in deduplicating storage systems, with particular focus on leakage caused by using content-defined chunking as the segmentation mechanism. They show empirically that under a number of strong assumptions on the target files, a cloud provider can infer the contents of low-entropy files with high probability even if the encryption key is unknown. This attack vector is tangential to the problem tackled in this paper, but it does emphasize the need for rigour in analyzing security of cloud storage in the presence of malicious clients and servers. Another closely related area is cache privacy attacks such as those considered by Ács et al. [2] in Named Data Networking; we believe that our model can also be applied to such scenarios.

The rest of this paper is organized as follows. Side-channel attacks on cloud storage and some existing countermeasures are reviewed in Section 2. Our security model and optimality of defences are discussed in Section 3. Section 4 proves our main theorems relating security and efficiency, characterizing both good dedpulication strategies and the essential trade-off between security and efficiency. In Section 5 we discuss how our work relates to other countermeasures and approaches.

2. SECURITY FOR DEDUPLICATION

This section reviews the current status of side-channel attacks on client-side deduplication and their countermeasures. For the rest of this paper, we will discuss client-side deduplication only, unless explicitly stated otherwise, and focus only on side-channel issues. We define *users* as the entities with distinct logins to a system, and *clients* as the devices that interact with the server on behalf of their owner, the user. Users and clients may be adversarially controlled: for the attacks we describe we consider an adversary that has access (i.e. login credentials) to the cloud storage service and attempts to glean information from its interactions with the server. The side-channel attacks we focus on are not the only type of attack in this scenario. If files can be retrieved using only a (deterministically-derived) index such as the hash of the file then this introduces the issue of users being able to share files with others, potentially creating copyright issues [13]. This issue can be solved by incorporating proofs of ownership (PoW) [6] into the deduplication process.

2.1 Existence-of-File Side-Channel Attack

Harnik et al. [7] identified the side channel inherent in client-side deduplication and discussed its implications in terms of three closely-related attacks performed by an adversary that follows the upload protocol correctly.

1. *Learning file contents.* An attacker can guess the contents of a file and infer its existence in the cloud.
2. *Identifying files.* The adversary can identify whether an incriminating file that should not be in the cloud, such as pirated media or a leaked document, is stored. If found, the owner could be later identified with the help of law enforcement access.
3. *Covert channels.* The existence of a unique file in the cloud can be used to signal a bit in a covert communication channel.

These are three outcomes of the same attack mechanism: an adversary wishes to learn whether or not a file has previously been uploaded to the storage of a CSP and then does something with the single bit of information it learns. We will therefore use the general term *existence-of-file attack* to incorporate any attack in which the adversary aims to learn whether or not a file has been previously uploaded. This term includes the notion of the aforementioned *salary attack* because of the following scenario.

- In order to implement client-side deduplication, the client first sends a short identifier to the CSP. The CSP instructs the client to upload the full file only if it is not already stored in the cloud.
- The adversary creates a template of an employment contract of Bob and attempts a number of uploads of files that only differ in a specific field (e.g. the salary).

外文资料译文

1 引言

重复数据删除是许多云存储提供商和服务使用的一个过程，用于删除存储在云中的冗余数据副本。已经显示 [12,14] 其在实践中大大降低了存储要求，因为个人和公司的用户通常存储相同或相似的内容。重复数据删除可以在服务器端或客户端进行。在服务器端重复数据删除中，服务器会检查客户端上载的文件是否已经存储。如果是这样，服务器不会再次存储它，而是由客户端记录文件所有权，并允许客当前户端使用合适的索引访问该共享文件。服务器端重复数据删除实现了减少存储空间占用的目的，但仍然要求客户端上载它希望存储的每个文件。在客户端重复数据删除中，希望首先上载文件的用户检查文件是否已经存储在云中，例如通过将文件的散列发送到服务器，该服务器检查其存储的文件散列列表。如果文件已经存储，则客户端不会发送文件，但服务器允许客户端像以前一样访问共享文件。因此，除了降低存储要求之外，客户端重复数据删除还可以大大降低云存储的带宽需求。由于与存储成本相比，通信成本可能很高，因此客户端重复数据删除通常优于服务器端重复数据删除，这是出于经济原因。重复数据删除可以针对文件或关于块进行，但我们不会关注这种差异，因为本文中考虑的许多攻击和对策都可以应用于任何一种方法。

1.1 加密重复数据删除

尽管存储和带宽节省了很多，但重复数据删除至少会导致两个主要的安全和隐私问题，这导致了最近在安全重复数据删除方面的广泛工作 [14]。第一个问题是，如果部署了语义安全的端到端加密，则无法进行重复数据删除。在密文不可区分的情况下，不具有解密密钥的云服务提供商（CSP）将无法确定两个密文是否对应于相同的明文。为了解决这个问题 [1,3,5,8,9,18]，已经提出了几种替代的加密形式，以各种方式从文件本身导出加密密钥。这些工作通常会对文件不可预测性或关键分布做出强有力的假设。第二个问题是客户端重复数据删除可以作为在不同攻击下泄露信息的辅助渠道 [7]。本文重点介绍这些侧通道攻击。

哈尼克等人。[7] 确定了客户端重复数据删除中由于侧通道引起的三次攻击。这些攻击适用于跨用户方案，其中上载相同文件的不同用户将对其数据进行重复数据删除。所有攻击的基本思想是，一个用户可以通过接收显示文件是否先前上载的信号来获取有关另一个用户文件的信息。在一个例子中，有时称为工资攻击，对手试图在她与 CSP 存储的雇佣合同中学习 Alice 的私人数据（她的工资）。攻击者在模板文件上插入猜测并将其上传到 Alice 的相应文件所在的 CSP。然后，重复数据删除信号的出现将允许对手推断出猜测的正确性。

确定了这些侧通道攻击后，Harnik 等人。[7] 提出了一种对策，其中关于文件是否已经上传的信号被随机化隐藏。更具体地，对于每个文件，随机均匀地选择阈值，并且如果先前上载的数量满足或超过阈值，则仅通知用户不上载文件。与基本的客户端重复数据删除相比，这显然会增加所需的带宽。在服务器端重复数据删除中不会出现旁道，因为客户端总是上传文件并允许服务器决定是否进行重复数据删除。因此，随机阈值对策可视为客户端重复数据删除的效率与服务器端重复数据删除的安全性之间的折衷。

1.2 主要贡献

虽然 Harnik 等的缓解思路。[7] 已经在文献 [10,17] 中进行了讨论和开发，没有正式建模和分析基于阈值的解决方案来保护侧通道攻击。这预示着有机会正式比较不同解决方案的有效性。本文的目的是通过以下方式纠正这种情况：

- 为旁道重复数据删除策略提供正式定义，包括对策的有效性的自然衡量标准;
- 确定优化带宽和安全性战略所需的条件;
- 描述所有战略所需的安全性和效率之间的权衡;
- 显示 Harnik 等人的原始提案。[7] 在一个自然安全措施中提供最佳防御。

在其他情况下，攻击者可以使用类似的旁道。在独立和同时的工作中，Ritzdorf 等人。[16] 考虑在重复数据删除存储系统中泄漏给好奇的云提供商的信息，特别关注使用内容定义的分块作为分段机制引起的泄漏。他们凭经验证明，在目标文件的许多强烈假设下，即使加密密钥未知，云提供商也可以高概率地推断低熵文件的内容。这个攻击向量与本文讨论的问题相关，但它确实强调了在存在恶意客户端和服务器的情况下分析云存储安全性的严格要求。另一个密切相关的领域是缓存隐私攻击，例如 A cs 等人所考虑的那些。[2] 在命名数据网络中; 我们相信我们的模型也可以应用于这种情况。

本文的其余部分安排如下。第 2 节回顾了对云存储的侧通道攻击和一些现有的对策。第 3 节讨论了我们的安全模型和防御的最优性。第 4 节证明了我们关于安全性和效率的主要定理，同时描述了良好的解除策略安全与效率之间的基本权衡。在第 5 节中，我们将讨论我们的工作如何与其他对策和方法相关。