中国科学 : 信息科学 2025 年 第 55 卷 第 3 期: 516–527
SCIENTIA SINICA Informationis
《中国科学》杂志社
SCIENCE CHINA PRESS
先进计算与新兴软件专题 · 论文

云边融合安全存储架构及技术挑战

李经纬[1], 孙嘉[1], 杨劲远[1], 沈志荣[2], 杨浩淼[1], 方华[3], 陈厅[1*], 张小松[4]

1. 电子科技大学计算机科学与工程学院 (网络空间安全学院), 成都 611731
2. 厦门大学信息学院 (特色化示范性软件学院), 厦门 361005
3. 广东魅视科技股份有限公司, 广州 510440
4. 电子科技大学信息与软件工程学院, 成都 611731
* 通信作者. E-mail: brokendragon@uestc.edu.cn

收稿日期: 2024–11–01; 修回日期: 2025–01–10; 接受日期: 2025–02–17; 网络出版日期: 2025–03–04

国家重点研发计划青年科学家 (批准号: 2022YFB4501200)、国家自然科学基金 (批准号: 62072381, 62072081)、四川省国际科技创新合作/港澳台科技创新合作 (批准号: 2024YFHZ0339)、中央高校科研业务费 (批准号: ZYGX2021J018) 和四川省科技计划 (批准号: 2024NSFTD0031) 资助项目

摘要 物联网的迅猛发展对数据管理提出了诸多挑战, 为应对这些挑战并满足管理日益增长数据的切实需求, 国家制订了 “协同发展云计算与边缘计算” 的远景规划. 在此背景下, 本文提出了一种云边融合安全存储架构, 旨在探索物联网数据在云边融合场景下的有效管理途径. 该架构利用机密计算、数据缩减、跨域同步等关键技术, 支撑数据的高效管理和安全保障. 在此基础上, 本文进一步提出了一套适应于上述安全存储架构的系统核心设计方案, 解决实施过程中的效率问题. 此外, 本文还探讨了该架构下的未来研究方向, 以期为新场景下物联网数据管理提供新视角和解决思路.

关键词 云边融合, 数据缩减, 可信执行环境, 机密计算, 增量同步

1 引言

物联网 (Internet of Things, IoTs) 作为支柱新兴产业的重要组成部分, 已成为衡量国家综合实力的关键指标之一. 我国在 “十二五” 期间已将物联网列为国家发展规划的重点领域, 并予以大力推动. 随着 5G 技术的快速发展, 数据传输能力显著提升, 物联网应用得以更为广泛地部署. 根据国际数据公司的预测 [1], 到 2025 年, 全球物联网设备的连接数量将达到 246 亿, 总数据量预计超过 79.4 ZB (1 ZB = 10^{12} GB).

为了提高海量数据的管理效率, 我国 “十四五” 规划明确提出要 “协同发展云服务与边缘计算服务”. 该方案通过在云计算的基础上, 部署靠近终端的轻量级软硬件实例 (称为边缘节点), 实现关键数据资源的近本地管理, 有效降低网络负载和服务响应时延, 提供更加灵活、实时的计算模式. 这种将云端与边缘相结合的全新体系架构 (即云边融合), 被广泛认为是实现分布式资源灵活调度、全域数据高

引用格式: 李经纬, 孙嘉, 杨劲远, 等. 云边融合安全存储架构及技术挑战. 中国科学: 信息科学, 2025, 55: 516–527, doi: 10.1360/SSI-2024-0325
Li J W, Sun J, Yang J Y, et al. Cloud-edge integrated secure storage architecture and technical challenges. Sci Sin Inform, 2025, 55: 516–527, doi: 10.1360/SSI-2024-0325

© 2025《中国科学》杂志社

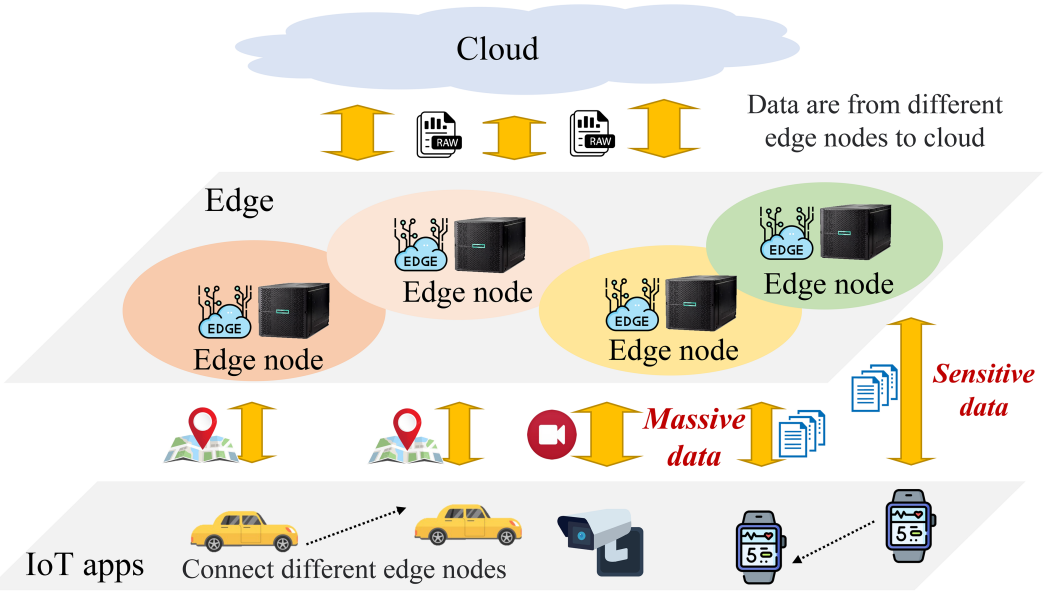

图 1　(网络版彩图) 面向物联网数据管理的云边存储场景.

Figure 1　(Color online) Cloud-edge storage scenarios for IoT data management.

速互联以及智能应用渗透边缘的重要路径. 云边融合代表了我国在数字基础设施领域的发展方向, 并为未来智能化应用和服务的深入发展奠定了坚实的基础.

本文聚焦于云边融合的存储架构 (如图 1 所示). 与传统云计算相比, 云边融合场景下的存储系统面临三大关键需求与挑战.

- 防不住. 由于边缘节点部署在靠近终端的区域, 分布范围较广, 原有的云安全防护体系需扩展至边缘节点, 以应对其更为复杂和分散的安全威胁.
- 存不下. 随着终端设备连接数的迅速增长, 边缘节点须具备高效的数据存储能力, 以应对来自不同终端的海量数据, 尽可能减少数据访问时延, 提升服务响应速度.
- 找不到. 终端设备 (如智能可穿戴设备) 在不同区域间移动时, 会动态连接到不同的边缘节点. 这种情况下, 不同区域的边缘节点可能存在数据不一致问题, 难以确保终端设备能够从任意边缘节点访问到相同数据.

为了应对上述挑战, 本文提出了一种基于可信执行环境 (trusted execution environment, TEE) 的云边融合安全存储架构, 旨在充分利用机密计算 (confidential computing) 技术 [2,3], 将安全存储基础设施从传统的云计算扩展至云边融合场景. 具体而言, 针对 “防不住”, 在各边缘节点部署可信执行环境, 用以处理数据, 确保敏感信息不被泄露; 针对 “存不下”, 通过在可信执行环境内删除来自不同终端的冗余数据, 减少实际存储的占用空间, 此外, 边缘节点仅缓存部分热数据, 即使用频率较高的数据, 以降低边缘存储负载; 针对 “找不到”, 通过实现跨边缘节点的数据同步机制, 确保相同终端设备在不同边缘节点之间能够访问一致的数据, 解决终端设备在不同区域移动时的数据一致性问题.

依托上述架构, 本文进一步提出了一套安全存储系统的核心设计方案, 旨在弥补从理论架构到系统实践之间的差距. 该方案包含三项核心设计: (1) 面向可信执行环境的数据高效缩减, 通过优化数据处理流程, 减少跨可信执行环境与内存和外存的数据交换次数, 解决增量压缩的 I/O 效率瓶颈问题; (2) 异构边缘缓存和自适应演化, 根据不同终端的访问特性动态调整缓存策略, 确保缓存资源的高效利用和持续优化, 解决异构设备和动态访问模式带来的缓存挑战; (3) 支持加密数据增量同步的密码设计, 通过优化密钥生成和加密过程, 确保数据隐私的同时减少传输带宽, 提升跨边缘节点的数据同步效率. 上述设计方案融合了本文作者前期的研究工作 [4~7], 在避免增量压缩 I/O 开销 (4.2 小节) 和有

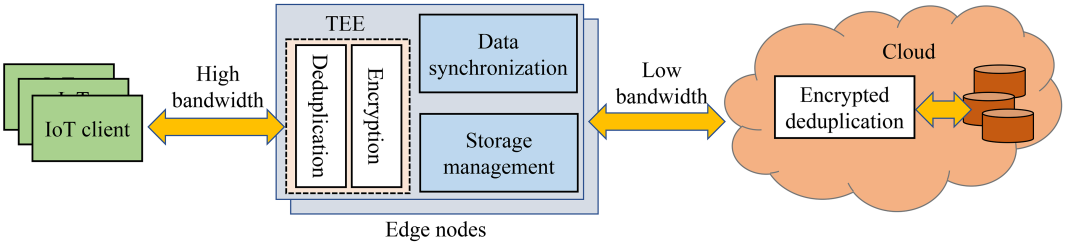

图 2　(网络版彩图) 基于可信执行环境的云边融合安全存储架构.

Figure 2　(Color online) Cloud-edge integrated secure storage architecture based on the trusted execution environment.

效管理边缘缓存 (4.3 小节) 方面形成了方法创新. 最后, 本文探讨了依托云边融合安全存储架构进一步解决数据容错、高效恢复、同步预测、安全计算、系统研制的潜在思路与技术挑战 (第 5 节).

2　相关工作

目前针对云边并存场景下的存储系统研究较少, 主要工作集中于优化边缘侧存储的数据分布, 以提高性能. SEND [8] 通过考虑数据流行度、数据处理的目标位置等因素, 智能调整跨边缘节点的数据存储布局, 从而提升边缘侧对用户请求的响应效率. 但是, 该工作未考虑如何降低边缘存储负载. 华中科技大学金海教授团队 [9] 采用纠删码技术降低基于多副本的边缘容错存储的存储负载, 并通过建模和求解纠删码编码块的跨边缘分布优化问题, 确保在降低存储需求的同时保持响应效率. EF-dedup [10] 通过构建互不相交的边缘集群, 并在集群内部执行去中心化重复数据删除, 旨在实现云边环境下传输与存储成本的优化平衡. HotDedup [11] 基于数据流行度与相似性, 设计了一种高效的边缘存储方案, 以提高边缘服务效率和存储空间利用率. LOFS [12] 采用三层哈希映射技术, 将相似文件映射至相近位置, 并结合边缘节点的存储容量约束, 增强了重复数据删除操作效率. 然而, 上述研究主要聚焦于云边场景下的存储策略优化, 尚未有效解决数据隐私保护的问题.

在安全性方面, 南京大学仲盛教授团队 [13] 设计了一种面向边缘存储的数据完整性审计协议, 允许第三方验证者在不破坏用户数据和查询模式隐私的前提下检查边缘数据完整性. 然而, 该协议依赖于复杂的公钥密码运算, 难以满足处理大规模物联网数据的需求. 本文提出一种云边融合安全存储架构, 旨在高效地解决安全性、存储效率和跨边缘数据一致性的问题.

3　安全存储架构概述

图 2 描述了本文提出的云边融合安全存储架构. 为了确保数据处理过程中的安全隐私, 在各边缘节点部署可信执行环境. 可信执行环境是一种由硬件支持的隔离计算环境 (如 Intel SGX [14] 或 ARM TrustZone [15]), 能够确保在其内部执行的代码和处理的数据免受外部环境 (包括操作系统、虚拟机监控器等) 的干扰或窃听. 考虑到在适当配置下, 可信执行环境 (如 SGX [14,16]) 能够提供较高的性能, 因此可以在其隔离区域内执行基于明文数据的敏感任务, 以确保数据处理的安全性. 同时, 该架构避免了计算密集型的安全操作 (如复杂的密码学运算), 从而维持了系统的整体效率.

受物联网数据高度可压缩特性的启发 [17], 每个边缘节点的可信执行环境负责执行数据缩减处理, 以有效减轻边缘侧存储管理压力, 同时降低从边缘到云端的数据传输负载. 具体而言, 数据缩减过程分为以下 3 个阶段: (1) 重复数据删除 [18], 以数据块为单位, 删除重复出现的冗余数据块; (2) 增量压缩 [19,20], 针对一组内容相似但并不完全相同的数据块, 仅保存其中一个完整数据块 (称为基础块), 并记录基础块与其他数据块之间的差异内容 (称为增量块), 进一步减少数据冗余; (3) 局部压缩, 使用通用的无损压缩算法 (如 LZ4 等) 对基础块和增量块进行压缩, 以最大化冗余数据缩减效果. 在完成上述

数据缩减后, 可信执行环境通过消息锁加密等机制 [21~23] 对处理后的基础块和增量块进行加密, 确保数据在离开可信执行环境后仍保持安全. 每个边缘节点仅管理热数据, 为终端提供实时访问响应, 并将冷数据加密后卸载至云存储管理. 云存储可以基于接收到的来自不同边缘节点的加密数据, 进一步执行跨边缘节点的冗余数据缩减操作 (注: 消息锁加密允许将相同原始明文数据块加密为相同密文数据块, 从而使云存储能够基于加密后的数据块检测重复的明文数据块, 进而支持重复数据删除, 详见文献 [21]), 仅持久性存储缩减冗余后的剩余内容.

为了满足终端访问请求的实时响应需求, 边缘节点应具备高效的存储管理功能, 以对热数据的生命周期进行智能控制. 热数据是指频繁被访问或处理且对时效性要求较高的数据. 通过在边缘节点缓存这些数据, 可以有效减少数据传输时的延迟, 提升系统的响应速度. 当热数据的访问频率降低时, 边缘节点会将其从缓存中移除, 或将其迁移到云端的持久性存储中, 从而释放宝贵的边缘存储资源, 为新的高频数据腾出空间. 为了进一步优化存储和访问效率, 边缘节点可以结合预测算法 (4.3 小节), 提前评估哪些数据在未来可能会被频繁访问. 通过这种预测机制, 边缘节点能够在数据需求激增之前, 主动将可能成为热数据的内容缓存在本地, 从而减少数据获取的时间延迟.

此外, 物联网终端通常会跨域移动, 在不同的边缘节点上存储不同时间段的终端数据 [24]. 这种情况下, 同一终端 (称为 “同源”) 在跨越多个边缘节点时, 可能会出现数据不一致的问题. 因此, 边缘节点必须具备跨边缘数据同步功能, 以确保终端设备在不同边缘节点之间的数据一致性. 具体而言, 跨边缘数据同步功能允许终端设备发生移动时, 自动协调和同步其在不同边缘节点上生成或更新的数据, 确保终端无论连接到哪个边缘节点, 均可访问最新且完整的数据副本. 这种同步机制不仅能够有效解决数据不一致的问题, 还能提升终端设备在不同区域间移动时的用户体验, 保障数据访问的连续性和可靠性.

4　系统核心设计

依托基于可信执行环境的云边融合安全存储架构 (第 3 节), 本文提出一套系统设计方案, 旨在弥补从理论架构到系统实践之间的差距. 首先讨论系统设计面临的挑战 (4.1 小节), 进一步提出三项核心设计元素 (4.2~4.4 小节), 以提升系统实用性和处理效率. 需要指出的是, 系统设计方案建立在作者前期研究 [4~7] 的基础上, 本文的主要创新体现在如何降低增量压缩 I/O 开销 (4.2 小节) 和有效管理边缘缓存 (4.3 小节).

4.1　设计挑战

依托上述架构设计实用的安全存储系统面临如下三大技术挑战.

挑战一. 在增量压缩过程中, 边缘节点需要频繁从本地存储或云存储中读取已存储的基础块并将其载入可信执行环境, 这会带来巨大的 I/O 开销. 该开销不仅包括将基础块从外部存储 (例如边缘节点的外存或云端) 读取到内存的外部访问, 还涉及从内存调入可信执行环境的环境切换 (context switching) [16, 25]. 因此, 基于可信执行环境的增量压缩进一步加剧了 (增量压缩的) I/O 开销问题. 一种解决方案是在可信执行环境内部管理所有的基础块, 从而避免频繁的 I/O 操作. 然而, 可信执行环境的大小受到宿主边缘节点物理内存的限制. 例如, 在阿里云的 g7t 虚拟机实例中, 可信执行环境只能使用宿主实例总内存的 50% [26]. GB 级别的可信内存空间不足以缓存所有的基础块. 当缓存的基础块达到可信执行环境的大小限制时, 系统会将加密的页面与不可信内存进行交换, 这将导致巨大的分页开销 (paging overhead) [14, 16, 25].

挑战二. 边缘节点通常会缓存多个终端设备的热数据, 以减少数据访问时的延迟并提升系统的响应效率 (第 3 节). 然而, 由于物联网场景的复杂性, 不同终端设备的数据访问模式在访问频率、数据大小与访问时间等方面存在显著差异. 即使针对同一终端设备, 这些访问模式并非是固定不变的, 而

是随着时间的推移不断演化. 例如, 某个终端可能在特定时间段 (如高峰时段) 频繁访问数据, 而在其他时间段访问频率较低. 进一步复杂化的问题是, 部分终端设备, 如智能可穿戴设备、智能汽车等, 具有跨域移动的特性. 这些终端设备在不同区域之间频繁移动, 导致其连接的边缘节点不断变化 [24]. 在这种情况下, 终端设备移动前连接的边缘节点中缓存的数据可能会失效, 无法被后续使用. 因此, 边缘缓存机制不仅需要识别和适应不同设备的异构访问模式, 还必须能够应对终端设备跨域移动带来的缓存失效问题.

挑战三. 在边缘节点之间进行文件同步时, 增量同步技术只传输文件的变动部分, 使目标节点能够基于增量数据和原始文件恢复最新版本 [27,28]. 由于增量数据通常远小于整个文件, 该方法能够有效减少网络传输的开销. 但是, 在云边存储架构中, 增量同步面临着安全性与效率难以兼顾的挑战. 为了保障同步数据隐私, 须以数据块为单位加密同步文件 (第 3 节), 然而加密机制破坏了数据块的语义, 使得修改前后的相似明文数据块被映射为完全不同的随机密文数据块. 在这种情况下, 即使明文数据块之间的差异很小, 加密后的密文数据块也将截然不同, 丢失了原有的相似性, 导致边缘节点难以生成和应用字节级增量, 将旧版本密文数据块直接更新为新版本密文数据块.

4.2　面向可信执行环境的数据高效缩减

本文作者在前期研究工作 [4] 中提出了基于频率的重复数据删除方法, 支撑在受限的可信执行环境内高效管理重复数据删除的哈希索引表. 为了快速检测冗余数据块, 重复数据删除需要维护哈希索引表来记录已处理数据块的哈希值 [18], 但随着存储数据的累积, 索引表逐渐增大 [18,29,30]. 由于可信执行环境的受保护内存空间有限 (4.1 小节), 当索引表超过该上限时, 分页开销会显著增大进而影响系统效率. 前期工作 [4] 发现, 存储负载中的冗余数据主要来自少量高频数据块. 因此, 提出基于数据块频率执行两个阶段的重复数据删除过程. 在可信执行环境内维护高频数据块的哈希索引表, 识别和删除重复的高频数据块 (第一阶段), 由于仅索引了少量高频数据块, 索引表所占空间较小, 避免了分页开销. 同时, 在可信执行环境外维护所有数据块的索引表, 处理剩余的低频数据块的重复删除任务 (第二阶段), 由于第一阶段已删除大量高频重复数据块, 此阶段减少了对外部索引表的访问频率, 降低了环境切换开销.

本文进一步解决增量压缩的 I/O 效率瓶颈问题 (4.1 小节). 本文考虑在可信执行环境内部缓存部分基础块, 并通过预取和更新必要的基础块, 减少增量压缩过程中频繁的 I/O 操作. 相关研究 [31,32] 表明, 同一数据来源的两次存储负载之间, 数据修改通常集中于少数数据区域. 因此, 本文认为, 如果最近存储负载中的某个数据块 M' 是由之前存储的某个数据块 M 修改而来的, 那么 M 很可能是 M' 的基础块. 同时, M 邻近的数据块也可能是 M' 邻近修改块的基础块.

具体而言, 本文考虑由边缘节点本地或云存储持久性管理基础块. 基于此, 提出了一种基于数据局部性的基础块预取方法: 当某个已存储的数据块 M 被检测为 M' 的基础块 (例如可以通过基于特征的方法 [33] 识别相似数据块) 时, 则将 M 及其邻近的基础块一并预取到边缘节点的可信执行环境内. 这样做的合理性在于, 这些邻近的基础块很可能会用于后续的数据块增量压缩, 从而减少频繁的 I/O 操作.

但是, 上述方法可能导致一些预取的基础块在短时间内并未被使用, 而这些长期累积未使用的基础块会占用宝贵的可信执行环境空间, 造成资源浪费. 为了解决这一问题, 本文基于 “当前存储负载中的绝大多数数据块所对应的基础块都源自最近存储的数据块” 的观察 [34,35], 提出了一种推迟压缩方法. 推迟压缩的核心思想是: 如果某个数据块 M' 对应的基础块 M 是很久之前存储的数据块, 则仅将 M' 临时缓存到可信执行环境中, 而不立即执行增量压缩. 当后续处理的数据块也使用 M 或 M 邻近的数据块作为基础块时, 才将 M 及其邻近的基础块预取到可信执行环境中, 并对 M' 执行增量压缩. 该方法的合理性在于, 可以避免频繁地预取很久前存储、短期内不太可能被使用的基础块.

基于上述方法, 图 3 描述了在可信执行环境内融合基础块预取和推迟压缩的增量压缩处理流程.

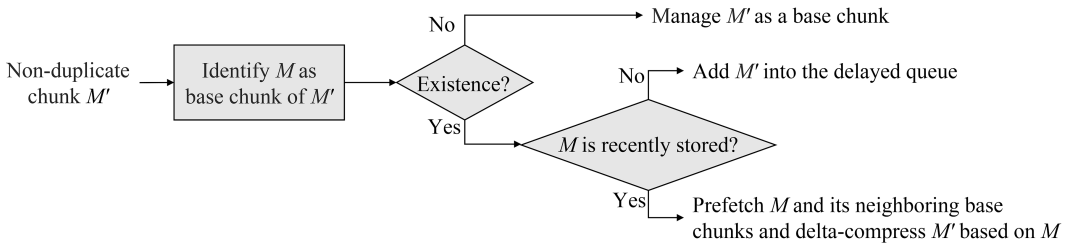

图 3　融合基础块预取和推迟压缩的增量压缩处理流程.

Figure 3　Delta compression processing flow combining base chunk prefetching and deferred compression.

需要特别指出的是, 由于可信执行环境空间有限, 难以无限推迟压缩操作. 当缓存的推迟压缩数据块数目达到上限时, 可信执行环境须逐出相应的数据块 M', 直接对 M' 使用局部压缩并加密存储.

4.3　异构边缘缓存和自适应演化

4.3.1　异构边缘缓存

为了实现边缘缓存的高效管理, 本文提出了一种异构的边缘缓存机制. 该方法基于的核心观察是, 不同的物联网终端设备可能频繁访问某些相同的数据 (如流行的音视频文件); 同时, 各终端设备也具有独立的访问模式. 因此, 提出的异构边缘缓存由静态缓存和动态缓存两部分构成: (1) 静态缓存用于存储所有终端频繁访问的公共数据块, 由于这些数据块通常不会频繁更新, 静态缓存可以采用无锁数据结构, 具备轻量、快速的优势, 减少缓存竞争导致的性能开销; (2) 动态缓存负责存储各终端设备的实时访问数据, 基于传统的缓存构建方法, 采用最近最少使用策略逐出数据, 以支持缓存内容的频繁更新, 确保最新的访问数据能够及时响应.

异构边缘缓存将遵循如下流程处理数据块访问请求. 首先, 查询静态缓存, 如果请求的目标数据块存在于静态缓存中, 则读取该数据块, 在可信执行环境中解密后通过安全信道返回到终端. 如果目标数据块不存在于静态缓存, 则查询动态缓存: 如果目标数据块存在于动态缓存, 按照如上方式解密返回并更新动态缓存的索引结构 (例如访问计数器或最近使用时间); 如果目标数据块在动态缓存中也缺失, 则从边缘节点本地或云存储中预取该数据块及其周围的相关数据块, 并将其加载至动态缓存中以备后续使用.

上述设计的合理性在于两方面. 首先, 真实的数据访问模式具有高度倾斜特征 [36], 即只有少量数据会被频繁访问, 这种访问的 “二八法则” [36] 决定了通过无锁的静态缓存管理这些高频数据, 可以有效降低缓存管理的开销, 显著提升高频访问数据的访问速度. 并且, 静态缓存无需频繁更新, 采用无锁结构能够减少竞争带来的性能损耗. 其次, 终端设备的访问模式通常表现出局部性, 例如在一次访问中, 先后连续访问数据块 M_1 和 M_2. 因此, 当 M_1 再次被访问时, M_2 也很可能在短期内再次被访问. 基于这一局部性特征, 预取周围数据块有助于提高动态缓存的命中率, 减少从云存储获取数据的频率, 提升系统的响应效率.

4.3.2　自适应演化

针对访问模式的持续变化挑战, 本文进一步提出了一种自适应演化静态缓存和动态缓存比例的方法 (图 4), 通过动态调整两者在边缘缓存中的占比, 维持较高的访问命中率. 该方法将边缘缓存的工作周期划分为 3 个阶段: 运行阶段、学习阶段和重构阶段. 在运行阶段, 静态缓存和动态缓存按照前述设计 (4.3.1 小节) 正常运行, 系统持续监控静态缓存的访问命中率. 如果静态缓存的命中率低于预设的阈值, 表明访问模式可能发生了变化, 此时边缘缓存进入学习阶段.

在学习阶段, 根据当前访问模式, 重新评估静态缓存和动态缓存的大小比例. 具体的, 将缓存中的数据块按照静态缓存优先的方式组织, 使用链表结构管理数据块. 链表的表头部分代表静态缓存, 表尾

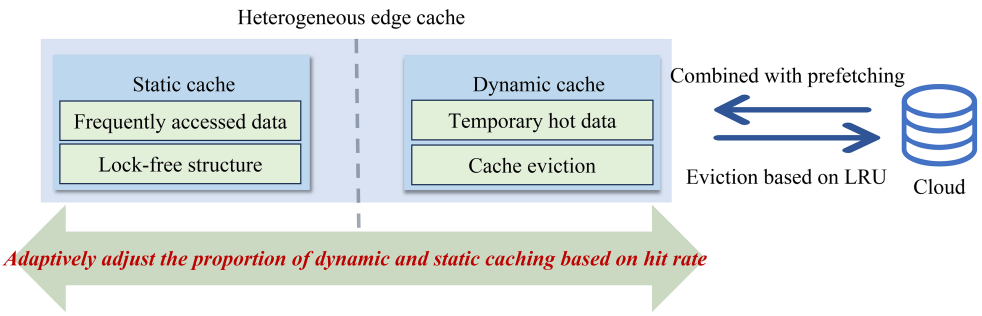

图 4　(网络版彩图) 异构边缘缓存和自适应演化设计.

Figure 4　(Color online) Design of heterogeneous edge caching and adaptive evolution.

部分代表动态缓存, 并标记上一个阶段中静态缓存和动态缓存数据块在链表中的分隔点 (临界位置). 静态与动态缓存的占比通过如下方式进行动态调整: 当静态或动态缓存中的数据块被访问命中时, 该数据块会被移动到链表表头并进入静态缓存区域, 由于临界位置不变, 此时静态缓存增大, 以适应 (不同终端之间) 同构性增强的访问模式; 当访问的目标数据块不在缓存中并需要外部载入时, 新载入缓存的数据块会被插入到链表的临界位置并进入动态缓存区域, 意味着动态缓存增大, 以适应 (不同终端之间) 异构性增强的访问模式. 通过这一过程, 边缘节点逐步感知新的访问模式信息. 当链表中静态缓存区域的数据块支持的访问命中率稳定达到阈值时, 可以得到新的静态缓存和动态缓存的大小比例.

在重构阶段, 边缘节点根据在学习阶段确定的新的静态缓存和动态缓存大小, 重新组织链表中的数据块, 并将它们写入各自的缓存区域: 静态缓存依然采用快速的无锁数据结构, 保证在高频数据访问时的效率. 完成缓存的重构后, 边缘节点返回到运行阶段, 并开启新一轮的工作周期.

上述自适应演化方法的优势在于: (1) 随着访问模式的变化, 边缘节点能够自动调整静态缓存和动态缓存的比例, 保证在不同访问模式下都能维持较高的命中率; (2) 在学习阶段, 通过链表结构中的数据块移动, 边缘节点能够快速感知当前访问模式的变化, 并根据访问命中情况做出调整.

4.4　支持加密数据增量同步的密码设计

加密数据增量同步要求加密方法能够支持密文的更新转换, 即允许目标边缘节点直接在密文文件上应用增量, 将加密的旧版本文件转换为加密的新版本文件. 为了实现这一点, 加密方法需满足以下两项要求: (1) 同一文件的不同版本必须使用相同的密钥, 以确保文件中未修改的数据块仍能被加密为相同的密文数据块 (架构采用数据块级加密, 见第 3 节), 从而避免重新传输; (2) 加密过程需保持相似性, 即当文件的某个明文数据块发生少量修改时, 反映到相应密文数据块的变化也应是有限的, 以支持字节级的增量同步.

本文作者在前期工作中 [5,6] 提出了相似性感知的密钥生成方法 (图 5), 为相似文件生成相同密钥. 该方法为每个文件选择一个代表性数据块, 例如代表性数据块可以是文件的所有数据块中具有最小哈希值的数据块 [37], 然后基于文件的代表性数据块生成密钥, 并使用该密钥加密相应文件的所有数据块. 在现实场景中, 文件的版本更新往往只发生少量数据块修改, 不太可能改变该文件的代表性数据块和密钥. 因此, 在版本更新前后, 文件的所有数据块仍然采用相同密钥加密, 从而避免在同步过程中传输未修改的数据块. 需要注意的是, 前期工作 [6] 仅满足要求 (1) 但并不足以支持安全且高效的增量同步. 其主要原因在于, 由于采用了数据块级加密, 在极端情况下, 即便仅修改某个数据块的一个比特, 仍需重新同步整个数据块, 导致显著的网络负载开销.

本文作者近期提出了相似性保持的加密方法 [7] (图 5), 旨在降低少量修改的密文数据块之间的增量同步开销. 受流密码中按比特位异或可以使不同比特位的加密结果互不影响的启发, 首先采用基于流密码的计数器 (counter, CTR) 模式模式加密明文数据块. 在此模式下, 明文数据块 M 将与由文件

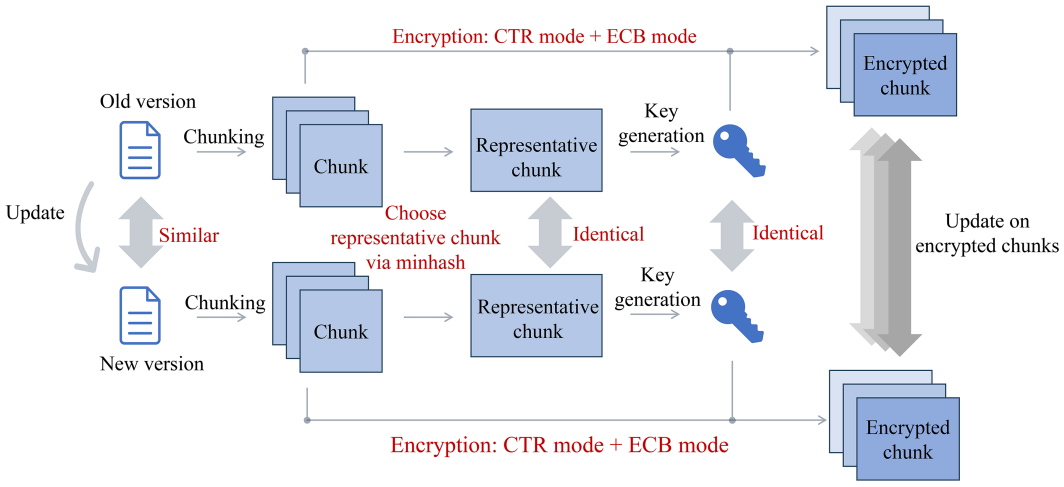

图 5　(网络版彩图) 支持加密数据增量同步的密码设计.

Figure 5　(Color online) Cryptographic design to support delta synchronization on encrypted data.

级密钥 K (K 基于代表性数据块生成) 和分组计数器 ctr 共同生成的掩码按比特位异或, 生成密文数据块 C: $C = M \oplus \boldsymbol{F}(K, \text{ctr})$, 这里 ctr 为计数器, 初始值为 0 并在每处理完 M 的一个分组 (16 字节内容) 后自增 1, $\boldsymbol{F}()$ 为伪随机函数. 由于修改前后的数据块之间具有相同的密钥和掩码, 对应的密文数据块将保留原始明文数据块之间的相同内容, 支持增量同步.

然而, 敌手可通过按位异或相同位置的密文分组移除掩码, 从而在不知道密钥情况下推测修改前后原始明文分组之间的异或结果. 为此, 在 CTR 模式的基础上, 进一步使用电码本 (electronic codebook book, ECB) 模式模式加密 CTR 模式输出的密文分组. 由于 ECB 模式基于密钥独立加密每个分组, 可将 CTR 模式输出的非重复密文分组 (敌手可按位异或这些密文分组, 获得原始分组的异或结果, 见上文) 加密为统计不可区分的密文分组, 从而避免异或操作导致的原始明文数据块内容模式泄漏问题.

该加密方法还具有如下优势. 首先, 尽管 ECB 模式具有众所周知的分组频率泄漏隐患, 但本文采用的加密方法不会泄漏分组频率, 主要原因是每个分组仍然受到第一层 CTR 加密模式的分组计数器保护; 其次, 该方法将相似数据块在相同位置上的重复明文分组映射为重复密文分组, 从而支持基于密文块的增量压缩; 最后, 由于 CTR 模式和 ECB 模式均支持独立加密各分组, 该方法也支持分组层面的并行处理以提升加密效率.

5　挑战与未来研究方向

云边融合安全存储架构的实现, 还存在诸多挑战. 本文从容错存储、高效修复、同步预测、安全计算、系统研制等方面, 提出实现上述架构的未来研究方向.

容错存储. 边缘节点的稳定性不足导致所存储的数据易于丢失, 因此提升存储可靠性迫在眉睫. 传统多副本存储通过保存数据的多份完整拷贝来提高数据的可靠性, 但会大幅增加边缘存储空间的占用. 本文作者在前期研究中, 针对云边融合场景下基于日志结构合并树 (log-structured merge-tree, LSM-Tree) 的分布式键 – 值存储可靠性问题, 通过感知日志结构合并树的管理特征, 自动将树底层键 – 值对的多备份存储转换为纠删码存储或进一步将校验键 – 值对卸载至云端 [38]. 然而, 如何构建适用于不同索引结构 (例如基于 B$^\epsilon$- 树的 SplinterDB [39]), 甚至是一般化存储结构 (如块存储、对象存储) 的云 – 边容错存储方案, 仍然是一个尚未解决的开放性问题.

高效修复. 容错存储允许在数据丢失发生后恢复目标数据, 但如何高效地进行跨边缘的数据恢复仍然是一个难题. 各边缘节点并非完全对等, 而是根据地理位置或网络拓扑呈现聚集分布的特性. 例

如, 位于同一区域的边缘节点之间通常具有较高的传输带宽和较低的网络延迟, 而不同区域的节点之间则可能面临较低的带宽和较高的延迟. 网络传输带宽的异构性为如何选择参与修复的边缘节点带来了挑战. 此外, 在基于纠删码的跨边缘容错存储系统 (例如文献 [9, 38]) 中, 参与修复的数据块需要被汇聚到目标边缘节点以进行数据重构. 然而, 由于边缘节点的计算、存储、网络资源有限, 集中处理可能导致目标节点在恢复过程中遭遇瞬时资源瓶颈, 承受巨大压力. 因此, 设计修复算法时, 必须充分考虑参与节点的选择、节点之间的带宽、实时网络负载等因素, 以优化修复路径. 为了进一步降低目标节点的修复负载, 可以在数据块传输至中间节点时, 考虑与该节点上参与修复的数据块进行联合编码 [40], 以平衡各边缘节点的负载, 避免将修复压力集中在某一个节点上.

同步预测. 密文增量同步 (4.4 小节) 能够有效解决终端设备在跨越多个边缘节点时引发的数据不一致问题. 然而, 由于终端设备 (如车载感知设备和人体可穿戴设备) 的移动具有临时性和不确定性, 如何有效选择同步的目标边缘节点成为一个关键问题. 具体而言, 云边融合安全存储系统需要具备预测终端设备移动方向的能力, 主动引导数据的预同步操作, 以确保在终端设备完成移动之前, 目标边缘节点已拥有所需的数据副本. 通过这种预测驱动的同步策略, 可以有效避免终端设备发起请求后才进行的跨边缘节点数据同步, 从而提升终端设备在跨边缘节点移动过程中与不同边缘节点之间的交互效率. 为此, 系统需考虑结合行为感知、机器学习等多种技术. 首先, 通过分析终端设备的历史移动模式和环境因素 (如位置、速度、通信延迟等), 可以预测未来的移动路径, 并在此基础上选择最合适的边缘节点进行数据预同步. 同时, 边缘节点也需要具备一定的智能决策能力, 在终端设备接近目标边缘节点之前, 主动从云端拉取相关数据并存储在本地缓存中, 确保在终端设备完成移动和发起数据访问请求时, 所需的数据副本已经准备就绪.

安全计算. 除了安全、高效、可靠的数据管理外, 如何在存储数据的基础上直接进行计算, 以实现低成本、低功耗的数据利用, 已成为研究热点. 跨边缘的联邦学习允许数据保留在边缘节点本地, 各节点通过本地训练模型, 并仅上传模型参数进行全局聚合, 从而有效保护数据隐私 [41]. 然而, 边缘场景的特殊性为跨边缘联邦学习带来了动态性和可扩展性的挑战. 首先, 边缘环境具有高度的动态性, 节点的频繁加入和离开是常见现象. 因此, 研究能够高效应对这些变化的自适应联邦学习算法显得尤为重要. 其次, 随着参与联邦学习的边缘节点数量不断增加, 如何在保持有效通信和协调的同时, 管理大量节点的联合训练而不造成性能下降, 成为一大难题. 最新研究 [42] 提出了一种面向边缘的点对点去中心化联邦学习框架, 允许任何边缘节点扮演协调器、聚合器、客户端选择器、参与节点或这些角色的任意组合. 由于去除了单点依赖, 该框架能够显著增强可扩展性和适应性, 展示了未来联邦学习发展的潜在趋势.

系统研制. 尽管现有工作从学术研究角度提出了多种存储设计方案, 但研制一套稳定可靠的云边融合安全存储系统仍然面临诸多实际问题. 首先, 除了存储故障以外, 边缘节点可能会离线或网络连接不稳定, 导致数据无法实时同步. 因此在设计同步机制时需要考虑冲突解决策略. 传统的集中式一致性保障方法难以适用于大规模的边缘节点, 而分布式一致性算法, 如 Paxos, Raft 等, 在资源受限的边缘设备上实现存在困难. 其次, 边缘设备类型繁多, 包括服务器、传感器、智能终端、工业控制设备等, 硬件性能、操作系统和通信协议各不相同. 设备的异构性增加了系统实现的复杂度. 要实现云边融合, 需要制定统一的通信协议和数据格式, 或采用适配器模式, 实现对不同设备的兼容. 推动行业标准化也是长期的工作. 再者, 云边融合涉及云端和大量边缘节点, 日常管理和维护难度大. 设备的部署、更新、监控、故障处理等都需要高效的管理方案. 需要研发自动化的运维工具, 支持远程管理和批量操作, 以提升管理效率.

6　结论

本文提出了一种新型的云边融合安全存储架构, 该架构通过融合多项关键技术来提升系统的整体

安全性和功能性: (1) 采用机密计算技术增强边缘节点的安全防护能力; (2) 利用数据缩减技术提升边缘节点的存储效率; (3) 通过数据同步机制保障终端设备跨区域移动时的数据一致性. 基于此, 本文进一步提出了适用于该架构的核心系统设计方案, 侧重于解决增量压缩中的 I/O 效率瓶颈、异构设备和动态访问模式带来的边缘缓存挑战, 以及基于加密数据的跨边缘增量同步问题. 最后, 本文还深入探讨了该架构面临的研究挑战, 并提出了未来的研究方向.

参考文献

1 IDC. How you contribute to today's growing datasphere and its enterprise impact. 2019. https://blogs.idc.com/2019/11/04/how-you-contribute-to-todays-growing-datasphere-and-its-enterprise-impact/
2 Popa R A. Confidential computing or cryptographic computing? Tradeoffs between cryptography and hardware enclaves. Queue, 2024, 22: 108–132
3 Rao A. Rising to the challenge—data security with intel confidential computing. 2022. https://community.intel.com/t5/Blogs/Products-and-Solutions/Security/Rising-to-the-Challenge-Data-Security-with-Intel-Confidential/post/1353141
4 Yang Z, Li J, Lee P. Secure and lightweight deduplicated storage via shielded deduplication-before-encryption. In: Proceedings of USENIX ATC, 2022
5 Qin C, Li J, Lee P. The design and implementation of a rekeying-aware encrypted deduplication storage system. ACM Trans Storage, 2017, 13: 1–30
6 Wu S, Tu Z, Wang Z, et al. When delta sync meets message-locked encryption: a feature-based delta sync scheme for encrypted cloud storage. In: Proceedings of IEEE ICDCS, 2021
7 Zhao J, Yang Z, Li J, et al. Encrypted data reduction: removing redundancy from encrypted data in outsourced storage. ACM Trans Storage, 2024, 20: 1–30
8 Nicolaescu A C, Mastorakis S, Psaras I. Store edge networked data (SEND): a data and performance driven edge storage framework. In: Proceedings of IEEE INFOCOM, 2021
9 Jin H, Luo R, He Q, et al. Cost-effective data placement in edge storage systems with erasure code. IEEE Trans Serv Comput, 2023, 16: 1039–1050
10 Li S, Lan T, Balasubramanian B, et al. EF-dedup: enabling collaborative data deduplication at the network edge. In: Proceedings of IEEE ICDCS, 2019
11 Li S, Lan T. HotDedup: managing hot data storage at network edge through optimal distributed deduplication. In: Proceedings of IEEE INFOCOM, 2020
12 Cheng G, Guo D, Luo L, et al. LOFS: a lightweight online file storage strategy for effective data deduplication at network edge. IEEE Trans Parallel Distrib Syst, 2022, 33: 2263–2276
13 Tong W, Chen W, Jiang B, et al. Privacy-preserving data integrity verification for secure mobile edge storage. IEEE Trans Mobile Comput, 2023, 22: 5463–5478
14 El-Hindi M, Ziegler T, Heinrich M, et al. Benchmarking the second generation of Intel SGX hardware. In: Proceedings of Data Management on New Hardware, 2022
15 Pinto S, Santos N. Demystifying arm TrustZone: a comprehensive survey. ACM Comput Surv, 2019, 51: 1–36
16 Harnik D, Tsfadia E, Chen D, et al. Securing the storage data path with SGX enclaves. 2018. https://arxiv.org/abs/1806.10883
17 Lu T, Xia W, Zou X, et al. Adaptively compressing IOT data on the resource-constrained edge. In: Proceedings of USENIX HotEdge, 2020
18 Zhu B, Li K, Patterson H. Avoiding the disk bottleneck in the data domain deduplication file system. In: Proceedings of USENIX FAST, 2008
19 Shilane P, Huang M, Wallace G, et al. WAN optimized replication of backup datasets using stream-informed delta compression. In: Proceedings of USENIX FAST, 2012
20 Shilane P, Wallace G, Huang M, et al. Delta compressed and deduplicated storage using stream-informed locality. In: Proceedings of USENIX HotStorage, 2012
21 Bellare M, Keelveedhi S, Ristenpart T. DupLESS: server-aided encryption for deduplicated storage. In: Proceedings of USENIX Security, 2013
22 Bellare M, Keelveedhi S, Ristenpart T. Message-locked encryption and secure deduplication. In: Proceedings of EuroCrypto, 2013
23 Li J, Yang Z, Ren Y, et al. Balancing storage efficiency and data confidentiality with tunable encrypted deduplication.

In: Proceedings of ACM EuroSys, 2020
24 Noor J, Srivastava M, Netravali R. Portkey: adaptive key-value placement over dynamic edge networks. In: Proceedings of ACM SoCC, 2021
25 Ngoc T D, Bui B, Bitchebe S, et al. Everything you should know about Intel SGX performance on virtualized systems. In: Proceedings of ACM SIGMETRICS, 2019
26 Alibaba Cloud. g7t, security-enhanced general-purpose instance family. 2021. https://www.alibabacloud.com/help/en/ecs/user-guide/general-purpose-instance-families#section-bew-6jv-c0k
27 Xiao H, Li Z, Zhai E, et al. Towards web-based delta synchronization for cloud storage services. In: Proceedings of USENIX FAST, 2018
28 Cui Y, Lai Z, Wang X, et al. QuickSync: improving synchronization efficiency for mobile cloud storage services. In: Proceedings of ACM MobiCom, 2015
29 Lillibridge M, Eshghi K, Bhagwat D, et al. Sparse indexing: large scale, inline deduplication using sampling and locality. In: Proceedings of USENIX FAST, 2009
30 Bhagwat D, Eshghi K, Long D D, et al. Extreme binning: scalable, parallel deduplication for chunk-based file backup. In: Proceedings of IEEE MASCOTS, 2009
31 Meister D, Brinkmann A, Süß T. File recipe compression in data deduplication systems. In: Proceedings of USENIX FAST, 2013
32 Kruus E, Ungureanu C, Dubnicki C. Bimodal content defined chunking for backup streams. In: Proceedings of USENIX FAST, 2010
33 Zhang Y, Xia W, Feng D, et al. Finesse: fine-grained feature locality based fast resemblance detection for post-deduplication delta compression. In: Proceedings of USENIX FAST, 2019
34 Zou X, Yuan J, Shilane P, et al. The dilemma between deduplication and locality: can both be achieved? In: Proceedings of USENIX FAST, 2021
35 Zou X, Xia W, Shilane P, et al. Building a high-performance fine-grained deduplication framework for backup storage with high deduplication ratio. In: Proceedings of USENIX ATC, 2022
36 Tian L, Feng D, Jiang H, et al. PRO: a popularity-based multi-threaded reconstruction optimization for RAID-structured storage systems. In: Proceedings of USENIX FAST, 2007
37 Broder A. On the resemblance and containment of documents. In: Proceedings of Compression and Complexity of Sequences, 1997
38 Ren Y, Ren Y, Li X, et al. ELECT: enabling erasure coding tiering for LSM-tree-based storage. In: Proceedings of USENIX FAST, 2024
39 Conway A, Gupta A, Chidambaram V, et al. SplinterDB: closing the bandwidth gap for NVMe key-value stores. In: Proceedings of USENIX ATC, 2020
40 Li R, Li X, Lee P P C, et al. Repair pipelining for erasure-coded storage. In: Proceedings of USENIX ATC, 2017
41 Lim W Y B, Luong N C, Hoang D T, et al. Federated learning in mobile edge networks: a comprehensive survey. IEEE Commun Surv Tutorials, 2020, 22: 2031–2063
42 Ching C W, Chen X, Kim T, et al. Totoro: a scalable federated learning engine for the edge. In: Proceedings of ACM Eurosys, 2024

Cloud-edge integrated secure storage architecture and technical challenges

Jingwei LI[1], Jia SUN[1], Jingyuan YANG[1], Zhirong SHEN[2], Haomiao YANG[1], Hua FANG[3], Ting CHEN[1*] & Xiaosong ZHANG[4]

1. *School of Computer Science and Engineering (School of Cyber Security), University of Electronic Science and Technology of China, Chengdu 611731, China*
2. *School of Informatics (National Characteristic Demonstration Software School), Xiamen University, Xiamen 361005, China*
3. *Guangdong AVCiT Technology Holding Co., Ltd., Guangzhou 510440, China*
4. *School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China*
* Corresponding author. E-mail: brokendragon@uestc.edu.cn

Abstract The rapid development of the Internet of Things (IoTs) has posed numerous challenges to data management. To address these challenges and meet the growing demand for managing ever-increasing amounts of data, the nation has formulated a strategic plan for the "coordinated development of cloud computing and edge computing." This paper proposes a cloud-edge integrated secure storage architecture that aims at exploring effective management approaches for IoT data in cloud-edge fusion scenarios. This architecture leverages key technologies such as confidential computing, data reduction, and cross-domain synchronization to support efficient data management and security assurance. On this basis, the paper further presents a core system design solution tailored to the proposed secure storage architecture, addressing efficiency bottlenecks during implementation. Additionally, the paper discusses future research directions under this architecture, aiming to provide new perspectives and solutions for IoT data management in emerging scenarios.

Keywords cloud-edge integration, data reduction, trusted execution environment, confidential computing, delta synchronization