華中科技大學

HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

数据中心技术文献综述

学院	武汉光电国家研究中心
班级	硕 **2104** 班
小组	无
姓名	**Sheng**
学号	**M202173484**

2021 年 1 月 7 日

非易失内存完整性维护方法综述

Sheng

摘　要　非易失性存储器（NVM）由于其可扩展性、非易失性、超低空闲功率和低延迟，被认为有希望取代动态随机存储器（DRAM）。然而，伴随非易失性而来的，是数据安全与完整性保证的挑战。就数据安全性方面而言，目前主流采用的计数器加密模式已经在理论上验证具有较优的性能，可满足绝大部分安全场景的性能需求。然而，现有的数据完整性方案大多停留在理论研究阶段，没有考虑实现时的性能和成本问题，因而不具有实际应用价值。在本综述中，我们以 Anubis、Phoenix、ARES 为例，介绍近年来在数据完整性验证方案在软件和硬件层面所做性能优化的进展。实验结果显示，作为第一个专注于加速故障后恢复的内存完整性方案，Anubis 通过优化体系结构，以小于 20%运行时间的性能代价，将基于 SGX 树的数据验证时间从数小时缩短至秒级别；Phoenix 在 Anubis 的基础上作了进一步优化，在维持秒级恢复时间的同时将 NVM 写入量缩小了 30%以上；ARES 则从硬件角度出发，使用现场可编程门阵列（FPGA）优化基于 Merkle 树的完整性验证流程，同样达到了秒级数据验证时间，相比于对照方案还有近 40%的运行速度提升。

关键词　安全；非易失内存；数据持久化；持久化安全；崩溃一致性；安全数据恢复

A Survey on Integrity Maintenance of Non-Volatile Memory

Sheng

Abstract　Emerging non-volatile memory (NVM) is considered due to its scalability, non-volatility, ultra-low idle power and low latency, and is promising to replace dynamic random-access memory (DRAM). However, non-volatility put stress on the importance of data security and integrity. Concerning data security, the state-of-the-art counter-mode encryption has been theoretically verified to satisfy the performance requirements of most security scenarios. However, most of the existing data integrity schemes still stay in a theoretical stage, and do not consider the performance and cost in implementation, therefore is completely impractical. In this survey, we take Anubis, Phoenix, and ARES as examples to comprehensively introduce works on performance optimization of the data integrity schemes in recent years, both hardware and software. Experiment results show that as the first memory integrity scheme focused on post-failure recovery speedup, Anubis, an SGX tree-based scheme, take optimization in architecture and reduce data verification time from hours to a magnitude of seconds at a performance cost of less than 20% of execution time; Phoenix is further optimized on the basis of Anubis, maintaining the recovery time with a few seconds while reducing the NVM data writes by more than 30%; ARES employs a hardware approach by leveraging field-programmable gate array (FPGA) to speedup Merkle tree-based integrity scheme. ARES also reached a second-level data verification time, plus a 40% execution time speedup compared to baseline.

Key words　security; non-volatile memory; persistence; persistence security; crash consistency; secure recovery

1 引言

非易失性存储器（NVM）由于其可扩展性、非易失性、超低空闲功率和低延迟，被认为有希望取代动态随机存储器（DRAM）。另一方面，新兴的NVM 为内存管理和安全带来了一系列新的挑战：由于 NVM 即使在断电后也能保留数据，因此必须在系统的整个生命周期内以及在系统重启/崩溃时采取必要的持久性安全措施，而这要求设计者在基于 NVM 的存储系统的设计时就考虑持久性和安全性问题。为了在保证数据安全性的同时尽可能减小额外开销，通常选择采用安全元数据的方式确保安全和功能恢复[1,2]。

目前最为广泛采用的安全元数据方案为计数器加密模式（counter-mode encryption）[1-3]。在计数器加密模式中，每个内存块都与一个特定的计数器相关联，内存的写入/读取操作需要使用基于该计数器生成的密钥，通过加解密算法操作对应的数据块。虽然在设计上允许计数器以明文形式存在的同时保证加密算法不被破解，但是该算法无法应对重放攻击：将计数器值回退到旧版本可以在不破解加密算法的情况下破坏计数器模式加密的安全性，因而保护计数器值的完整性对保护数据安全性十分重要。目前主要使用树形结构保护计数器数据的完整性。

完整性树是一类树形结构的统称，特点是其每层节点数据是其子节点数据相关联，而叶子节点的数据与计数器值关联。除树根保存在安全处理器内部外，其余部分全部位于不安全的内存中。由于完整性树的连锁特性，每个计数器的更新都会引起树中整条路径的数据更新（直至树根），因此任何数据篡改都可以通过树根匹配检查发现。完整性树已应用于部分商业产品，例如英特尔的 SGX 指令（软件防护扩展）。此外，完整性树被认为是迄今为止最安全的方案，同时因为只有根部需要保存在安全处理器芯片内，具有较低的片上存储开销。

然而，将完整性树应用于 NVM 系统是一项具有挑战性的工作：首先，NVM 的掉电数据持久特性，使得在写入操作未完成时掉电可能导致数据损坏或明文数据泄露；其次，NVM 具有有限的写入性能和写寿命，每次内存访问时都更新完整路径会带来严重的写入延迟和写放大问题。第三，在一些完整性树（如 SGX 树）允许并发更新，因而依靠恢复计数器值来重建树的开销过大，是不现实的。最后，重建 Merkle 树需要在内存上线（对外提供访问）前完整以确保安全性。实际上，服务器上每处理器的 NVM 安装量可能达到 TB 级，如英特尔的 Xeon 服务器预计将配备 6TB 的 NVM 内存[4]。而基于现有 NVM 完整性方案对 TB 级的 NVM 进行完整性树重建需要数小时的时间，与 99.999%的可用性要求相差甚远（每年仅允许 5.25 分钟的故障时间）。

作为面向商用场景的 NVM 内存系统方案，必须要同时考虑功能（加密）、性能（访存速度）、完整性（容错或崩溃恢复）三大特点。目前在功能上和完整性上都已提出了较为完善的方案，如[1,2]，但在性能上离实际需求还有较大的差距。本综述将从数据完整性的角度，介绍近年来在 NVM 内存系统方案设计中关注性能优化的工作。

2 原理和优势

2.1 攻击模型

内存攻击大致包括两类：内存保密性攻击和内存完整性攻击。我们假设攻击者具有物理接触目标机器的能力。

内存保密性攻击属于被动攻击，一般通过嗅探等手段观察内存总线上传输的或存储在内存中的物理信息。

相反，内存完整性攻击属于主动攻击，包括：

(1) 欺骗攻击：试图任意修改现有的内存块；

(2) 拼接攻击：交换内存块；

(3) 重放攻击：重新发送过去的合法数据事务。

2.2 安全处理器

安全处理器是一种具有执行密码学操作的专用微处理器，嵌入在具有多种物理安全措施的包装中，具有一定程度的防篡改能力。与一般中央处理器（CPU）不同之处在于，安全处理器除内置加解密硬件电路外，还包含持久存储介质，可存储密钥等安全元数据。由于理论上一般认为芯片内部电路内部信息无法被物理方法窃取，即安全处理器的数据无法被篡改或嗅探，因而可在系统设计时将安全处理器及其内部组件视为绝对安全区域，在处理器内安全地加解密数据。

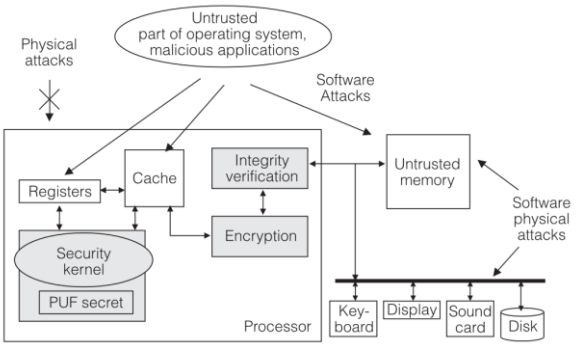

图 2.1　一种安全处理器架构及其可应对的安全性攻击示意

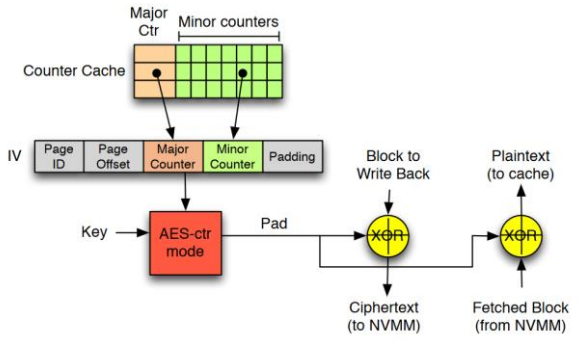

图 2.2　两级计数器加密模式

图 2.1 是一种安全处理器的架构设计。由图可见，除数据进出安全处理器需经过处理器内的安全逻辑处理外，其余架构与常见的计算机架构一致。

2.3　数据加密方案

2.3.1　对称加密模式

对称加密模式是最为原始的数据加密方式。在对称加密模式下，数据进出安全处理器时，直接使用 DES 或 AES 等对称加密算法对数据作相应的处理。由于 DES、AES 等加密算法具有数十周期的延迟导致的性能问题，以及对于相同的明文加密结果为相同的密文导致的安全问题，目前已不再使用该方法加密内存数据。

2.3.2　计数器加密模式

计数器加密模式分别通过将计数器加入密钥和使用 Pad 异或加密两种方法来改进对称加密模式的安全性和性能。首先，分配一个全局的计数器，每当内存块数据被修改一次后，计数器执行自增操作，以确保不同修订版数据的加密密钥相异；其次，相比于对称加密，计数器加密模式不直接使用密钥对数据加密，而是使用密钥对固定的密文加密（包含于安全处理器中），生成与数据大小一致的 Pad，再使用 Pad 与数据按位一一对应执行异或操作，得到加密密文，可将加密周期缩减至个位数。由于密钥不同，生成的 Pad 是不同且随机的。

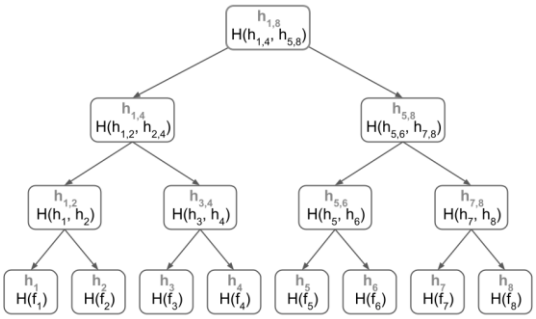

图 2.3　Merkle 树示意图

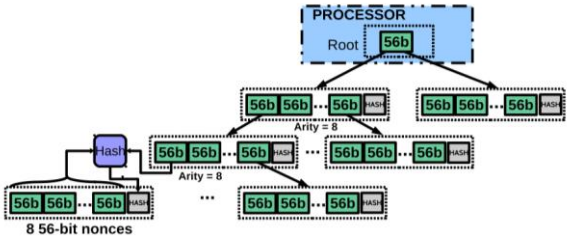

图 2.4　SGX 树示意图

图 2.2 是当前计数器模式中广泛采用的分级计数器模式。分级计数器可以有效减缓计数器的溢出频率，减少全局内存重加密带来的额外性能开销。

2.4　数据完整性方案

2.4.1　Merkle 树

Merkle 树的每个叶节点均以数据块的 Hash 作为标签，而除了叶节点以外的节点则以其子节点标签的加密 Hash 作为标签，如图 2.3 所示。Merkle 树能够高效、安全地验证大型数据结构的内容，是 Hash 链的推广形式。

Merkle 树可用于验证不可信来源的数据完整性：通常先从可信的来源获取 Merkle 树根的 Hash 值，在从不可信来源获取数据后，使用数据构建 Merkle 树，此时只需要比对两颗树树根的 Hash 值是否一致，即可得知数据是否被篡改。

2.4.2　SGX 树

图 2.4 展示了一棵 SGX 树的结构。与 Merkle 树不同的是，SGX 树的节点标签由计数器和 Hash 值共同组成，当前节点的 Hash 值为当前节点的计数器值的“签名”，数据的更新首先导致对应的计数器自增，接着触发当前节点签名的更新与上一级节点的计数器值更新。相比于 Merkle 树，SGX 树有两大优点：一是树的开度增大（每个中间节点有 8 个子节点，相比于 Merkle 树常用的二叉树），能有效降低树的高度，减少数据更新需要计算的 Hash 值总量；二是使用计数器保存节点之间的版本关系，从而使得路径更新的值可以预测，提供了原生的并行更新功能。

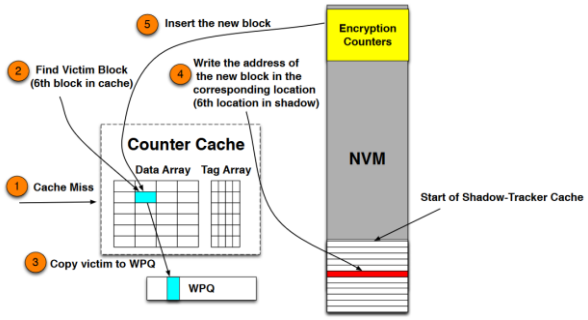

图 3.1 “影子跟踪”基本流程

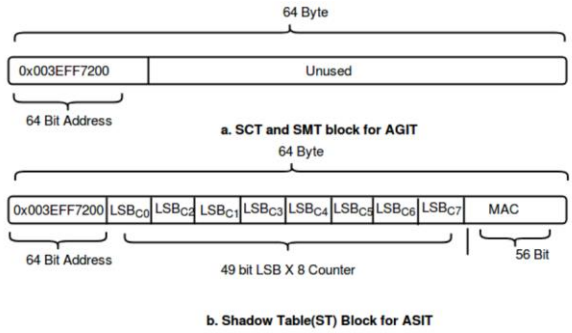

图 3.2 AGIT 跟踪表与 ASIT 跟踪表项的数据结构

3 研究现状分析 5000

3.1 Anubis[5]

Anubis 是一种新颖的纯硬件解决方案，可实现超低恢复时间并额外确保可并行化树（如 SGX 树）的可恢复性。Anubis 基于一个关键观察：持续跟踪计数器和 Merkle 树缓存中的块地址可用于显着减少恢复时间。但是，此类缓存中的地址不会像内容那样频繁更改；地址仅在缓存未命中时更改，因此在内存中跟踪地址的开销很小。通过跟踪这些地址，在崩溃后，系统只需要重建完整性树受影响的部分。此外，对于 SGX 树及其变种，Anubis 使用透明的元数据缓存并提供其完整性保障逻辑（“影子版本”）。因此，在 SGX 树恢复阶段只需要将影子版本的数据恢复回树中即可。Anubis 是第一个解决一般树的恢复时间和 SGX 树的可恢复性的方案，通过跟踪可能丢失的元数据和在持久内存中保留缓存的完整性保护阴影副本实现故障后的快速恢复。

3.1.1 方案设计

注意到 NVM 掉电事件中的一个关键现象是，只有被读入缓存的块有可能被更新而没有被持久化；因而，持续跟踪完整性树和计数器缓存中的块的地址就足以大幅减少恢复时间。基于这个思想，

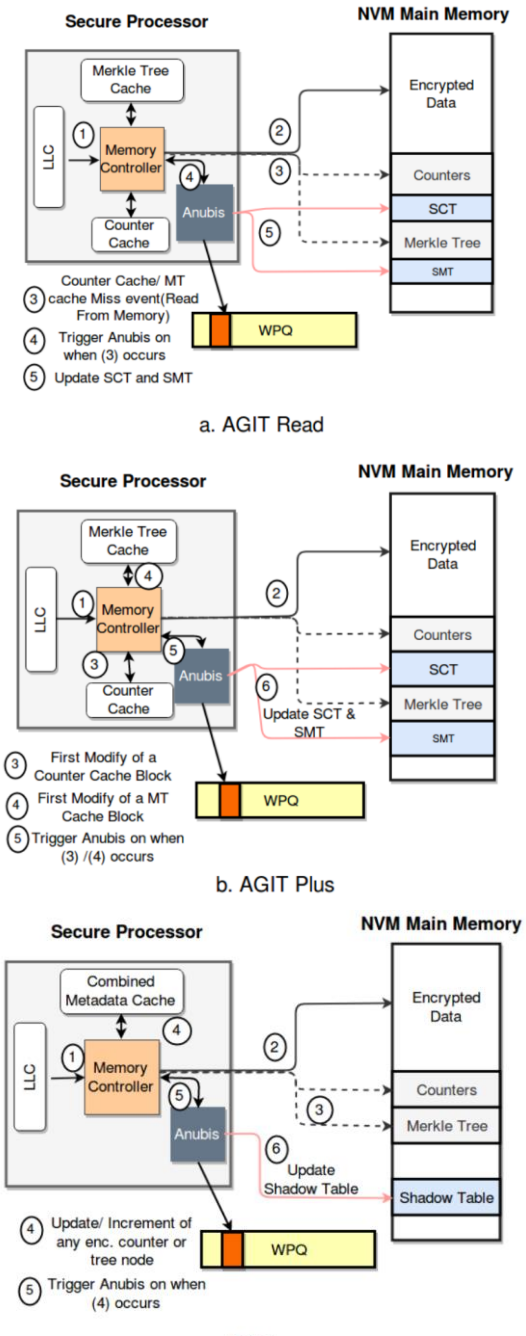

图 3.3 AGIT 跟踪表与 ASIT 跟踪具体流程

在记录了掉电时在缓存中的计数器块所对应内存块地址的情况下，只需要迭代对应的块的计数器的值即可完成对应块的恢复。同样，通过跟踪完整性树缓存中完整性树节点的地址，在修复损坏的计数器块后，就可从叶节点开始，逐层向上修复判定为损坏的计数器块，直至顶层，来实现完整性树的修复。由完整性树的特性可知，修复父节点值依靠的是子节点值的正确性。

作者将最简单的块地址跟踪方法称为“影子跟踪”，通过在 NVM 中分配一块与缓存大小相同的空

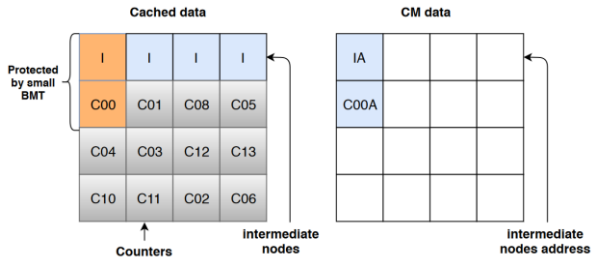

图 3.4　AGIT 跟踪表与 ASIT 跟踪具体流程

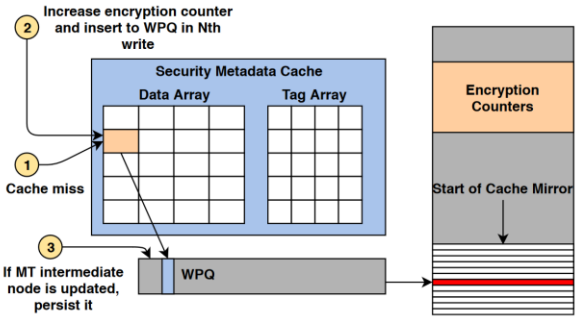

图 3.5　AGIT 跟踪表与 ASIT 跟踪具体流程

间来记录块地址以实现。在计数器缓存缺失事件中，缓存替换策略会选择一个旧缓存块会写回 NVM，并将读入的新块写至对应的缓存地址。影子跟踪算法将新区块的地址写入 NVM，其偏移量与缓存中的位置相对应，如图 3.1 所示。类似地，影子跟踪方法也可用于 Merkle 树。由于计数器缓存的失误率通常非常小，加上数据块在缓存中的一次生命周期通常不会变更其在缓存中的地址，跟踪算法导致的 NVM 写入量可以控制在较小的范围内。如前所述，Merkle 树和计数器缓存中的更新节点都必须被追踪。作者将计数器的影子跟踪区块称为影子计数器表（SCT），而 Merkle 树的影子跟踪区块称为影子 Merkle 树表（SMT）。在这两种情况下，存储开销都很小，例如，在 8TB 的内存下，跟踪表的开销只有 128KB，约占 NVM 大小的 10^{-8}。

针对 SGX 树和 Merkle 树，作者分别提出了 ASIT 和 AGIT 方案，并定义了对应的跟踪算法与跟踪表数据结构，如图 3.2、3.3 所示。其中对于 Merkle 树情景下的跟踪表结构，作者考虑到对于读取密集型应用可能会引入较大的额外写开销，进一步提出了由缓存修改而非缓存读取触发的跟踪方案，大幅减少了 NVM 写入量。

3.1.2　性能测试

作者基于 Gem5 模拟器搭建了一组 4 核 x86 环境、配有软件模拟的 PCM 内存，并使用来自 SPEC 2006 中的 11 个内存密集型测试集对设计的算法进行了测试。实验结果显示，Anubis 相对于不具有快速恢复设计的前置工作 Osiris，产生的平均额外性能开销小于 10%；在 8TB 的 NVM 容量下，重建时间从 7.8 小时缩减到约 0.48 秒，提升超过 10^5 倍。

3.2　Phoenix[6]

Phoenix 为 Anubis 作者在该方案上的进一步优化。Anubis 通过采用类似日志的缓存跟踪表以标记掉电前可能被修改的数据块地址；而在本文中，作者基于 Anubis，针对运行时性能提出了两种新的策略以减小运行时时间开销和 NVM 额外写入量。

3.2.1　方案设计

在本文中，作者提出了以下策略以提升基于 SGX 的 Anubis 在运行时的性能。

选择性安全元数据持久化。在系统崩溃时，缓存内容丢失。丢失缓存的安全元数据会导致完整性验证失败；因此，缓存的安全元数据需要被持久化。严格的持久化策略会在每次数据更新产生数十次额外的写入。为了避免这些不必要的额外写入，Phoenix 只持久化元数据中不可恢复的结点。由于使用 Osiris[26]可以在牺牲一定性能的情况下实现宽松的持久化，Phoenix 将遵循一个类似的方案，在每 N 次数据写入或计数器块驱逐时持久化加密计数器。另一方面，中间节点是不可恢复的；因此，Phoenix 持久化这些缓存节点以确保恢复成功。

宽松的缓存镜像。为了实现选择性的安全数据持久化，Phoenix 在 NVM 中分配了一个与安全元数据缓存相同大小的区域（约 256kB），称为缓存镜像，如图 3.4 所示。每当一个 SGX 中间节点被写入缓存时，这个更新将被直接持久化，其地址将被复制到缓存镜像，同时每写固定次数就更新一次加密计数器。图 3.5 解释了如何进行选择性的持久化；每当写发生时，如果写操作更新的是一个中间节点，更新的中间节点就会被复制到缓存镜像区域。在恢复过程中，缓存镜像的内容被用来恢复丢失的缓存内容并更新 SGX 树，以确保安全的恢复过程。缓存镜像仍处在处理器的安全边界之外，在恢复过程中使用缓存镜像之前，应该确保其完整性。因此，Phoenix 在缓存镜像上建立了一棵 4 层 8 开度的小型 Merkle 树，同时将这个树的根留在处理器中。在恢复过程中，缓存镜像区域的完整性验证通过建立缓存镜像及其对应的 Merkle 树并将产生的根与处理器保留的根进行比较以实现。通过使用缓存镜像记录更新的中间节点，并使用 Merkle 树验证其完整性，可以在 SGX 树不更新到树根的情况下仍然保证树的数据可恢复特性。

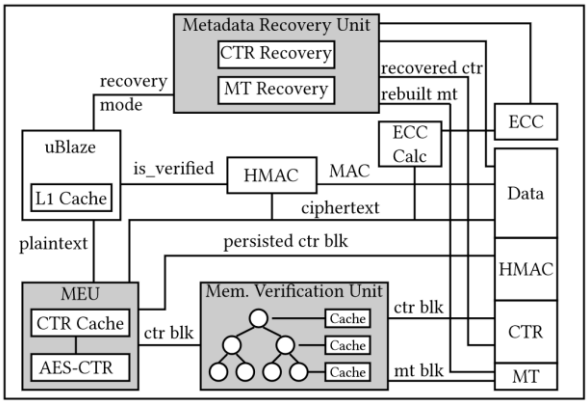

图 3.6　ARES 系统架构图

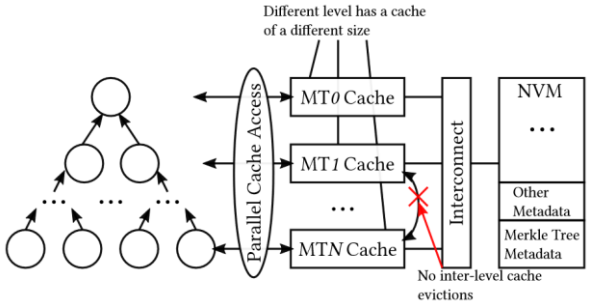

图 3.7　完整性树的分级缓存设计

3.2.2　性能测试

作者基于 Gem5 模拟器搭建了一组 4 核 x86 环境、配有软件模拟的 PCM 内存，并使用 SPEC2006 测试集对设计的算法进行了测试。实验结果显示，在额外写开销上，Phoenix 能将额外写入量控制在对照策略的 10%以内，相比于 Anubis 有平均 60%左右；在性能上，对执行时间的延缓不超过 5%，与 Anubis 的 10%性能损失相比有显著提升；在恢复时间上，Phoenix 也实现了秒级别的回复时间，且其随缓存容量的提升而增加的速度小于 Anubis。

3.3　ARES[7]

FPGA 的硬件可重构性可将性能敏感和安全相关的功能实现转移到可编程硬件，而无需微处理器参与。鉴于 CPU-FPGA 异构计算架构的日益突出，本文作者设计了一套 FPGA 辅助加速的 NVM 安全性方案，并基于 Xilinx FPGA 硬件系统实现了其原型。在系统设计过程中，作者对现有完整性树的特征做出了深入分析。

相比于传统方法，ARES 将 FPGA 逻辑作为处理器外的片上资源加入到安全边界中，采用的非入侵方法能够直接应用于现有处理器，处理器供应商无需修改内部逻辑即可应用 ARES 的安全方案，如图 3.6 所示。

根据作者分析，实现基于 FPGA 的 NVM 一致

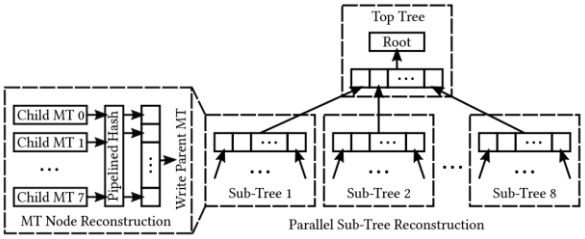

图 3.8　完整性树的子树设计

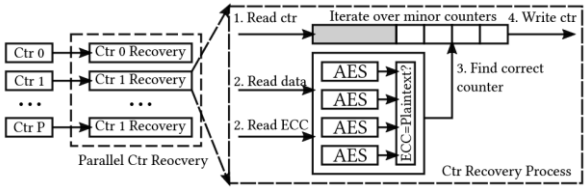

图 3.9　并行计数器回复流程

性系统具有如下三个挑战。

首先，传统的缓存策略在与 Merkle 树整合时不符合可信执行环境（TEE）的要求。所有驻留在 TEE 中的数据都必须经过验证，而传统策略下的缓存行直接来自外部存储器，没有经过验证，违反了 TEE 的要求。有两种方法可以解决这个问题：(1) 将 Merkle 树集成到缓存控制器中，(2) 设计一个具有新缓存策略的缓存。所有先前的基于仿真的工作[10, 22, 27, 30]都采用了第一种方法，而这种紧耦合的结构对于硬件设计和实现来说并不合适。

其次，传统的 Merkle 树是高度递归和串行的，而其缓存方案并未对此做出优化，对具有天生并行性的硬件实现并不友好。此外，用硬件处理潜在的无界递归调用通常是相当麻烦的。另一种改善 Merkle 树方案性能的不同方法是利用深度流水线来实现高吞吐量。然而，由于 Merkle 树的数据依赖性特征，处理时是顺序访问的，因而即使哈希计算单元每周期可开始一个计算操作，这些耗时的哈希函数也不能被流水线化。

最后，传统的元数据恢复过程是非常耗时的，因为它需要从树叶到树根恢复计数器并完全重建整个 Merkle 树。

3.3.1　方案设计

ARES 设计了三个关键策略来有效解决上述三个技术挑战。首先，ARES 采用 Merkle 树的一种变种，基于计数器构建的盆栽 Merkle 树，并设计了一种新的缓存策略。与传统缓存相比，它具有不同的读取未命中和写入未命中策略。本质上，当缓存读未命中时，读取的数据直接返回上层逻辑，而不放入缓存。当缓存写未命中时，缓存行的脏标记位根

据 Merkle 树操作设置，即验证或更新，而不是像传统缓存那样在任何情况下都设置为脏。

其次，ARES 提出了一种拆分缓存方案，将每个 Merkle 树级别与一个单独的缓存相关联，以并行化缓存访问，如图 3.7 所示。此外，利用拆分缓存架构，ARES 提出了一种新的硬件友好的 ARES-Merkle 树方案，通过重新组织计算和流水线化耗时的哈希操作来避免递归函数之间的控制依赖，同时不违反 TEE 的可信数据验证原则。

最后，作者创新地将 Merkle 树分解为多个子树，以最大限度地利用任务级并行性，加速 Merkle 树的重建，如图 3.8 所示。同样，计数器恢复也可以通过并行恢复模块来加速，图 3.9 展示了这一流程。此外，为了利用现代 FPGA 逻辑的可重构计算能力，ARES 专门为每个元数据恢复设计了一个完全流水线化的数据路径。

3.3.2 性能测试

本文的实验环境为实机搭建，作者使用 Xilinx U200 作为 FPGA 设备，用 DDR4 内存模拟 NVM 内存设备进行试验，采用 STREAM、DSPStone、BEEBS 工作集作为工作负载。实验结果显示，在保护 128MB 数据的情况下，大约产生 15%左右的额外内存开销；相比于原始的 Merkle 树，执行时间有 20%左右的提升，同时提供秒级别的掉电数据恢复时间。

4 总结及展望

由于数据完整性树层级间数据依赖的特性，对完整性树的节点恢复必须严格遵从自下而上的重建顺序以确保数据的完整性；由此引入的数据恢复时序也将恢复时间维持在了一个过高的水平。

Osiris 提出的关键数据跟踪是数据完整性性能优化中的一个里程碑，通过仅跟踪可能被修改过的数据，可观地减少了恢复过程中要操作的节点数量，缩短了恢复用时。本文所介绍的 Anubis、Phoenix、ARES 均直接或间接地采用了 Osiris 的思想与设计，并对在此基础上添加少量元数据，或对完整性树结构做优化，进一步优化了性能，在保障低运行开销的同时，使得快速错误恢复成为可能。

安全一直是计算机和互联网领域的一大重要需求，随着拥抱互联网的行业越来越多，计算机所需要承载的安全场景也越来越多、越来越复杂。因而，针对特定场景设计的 NVM 安全方案势必会在专业领域具有更大的优势。因而，从实际场景与条件出发，设计和优化特定的安全方案将会是一个能够权衡成本、复杂度与效率的关键方法。

参 考 文 献

[1] Ye M, Hughes C, Awad A. Osiris: A Low-Cost Mechanism to Enable Restoration of Secure Non-Volatile Memories[R]. Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2018.

[2] Liu S, Kolli A, Ren J, et al. Crash consistency in encrypted non-volatile main memory systems[C]//2018 IEEE International Symposium on High Performance Computer Architecture (HPCA). IEEE, 2018: 310-323.

[3] Rogers B, Chhabra S, Prvulovic M, et al. Using address independent seed encryption and bonsai merkle trees to make secure processors os-and performance-friendly[C]//40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007). IEEE, 2007: 183-196.

[4] Enhancing High-Performance Computing with Persistent Memory T echnology., https://software.intel.com/en-us/articles/enhancing-high-performance-computing-with-persistent-memory-technology. 2018, 12, 07

[5] Zubair K A, Awad A. Anubis: ultra-low overhead and recovery time for secure non-volatile memories[C]//Proceedings of the 46th International Symposium on Computer Architecture. 2019: 157-168.

[6] Alwadi M, Zubair K, Mohaisen D, et al. Phoenix: Towards ultra-low overhead, recoverable, and persistently secure nvm[J]. IEEE Transactions on Dependable and Secure Computing, 2020.

[7] ZOU Y U, ZUBAIR K A B U, ALWADI M, et al. ARES: Persistently Secure Non-Volatile Memory with Processor-Transparent And Hardware-Friendly Integrity Verification And Metadata Recovery[J]. ACM Transactions on Embedded Computing Systems, 2021, 1(1).