软件学报 ISSN 1000-9825, CODEN RUXUEW
2024,35(12):5582−5608 [doi: 10.13328/j.cnki.jos.007089] [CSTR: 32375.14.jos.007089]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于异构多核心 GPU 的高性能密码计算技术研究进展*

董建阔[1], 黄跃花[1], 付宇笙[1], 肖 甫[1], 郑昉昱[2], 林璟锵[3], 董振江[1]

[1](南京邮电大学 计算机学院, 江苏 南京 210023)

[2](中国科学院大学, 北京 100093)

[3](中国科学技术大学 网络空间安全学院, 安徽 合肥 230026)

通信作者: 肖甫, E-mail: xiaof@njupt.edu.cn

摘 要: 密码学是保障网络安全的核心基础, 其在数据保护、身份验证、加密通信等方面发挥着至关重要的作用. 随着 5G 和物联网技术的迅速普及, 网络安全面临着空前的挑战, 对密码学性能的要求呈现出爆发式增长. GPU 能够利用数以千计的计算核心并行化加速复杂计算问题, 这种并行化特性非常适用于密码学算法的计算密集型特性. 鉴于此, 研究人员广泛探索了在 GPU 平台上加速各种密码算法的方法, 与 CPU、FPGA 等平台相比, GPU 展现出明显的性能优势. 论述各类密码算法的分类与 GPU 平台架构, 对各类密码在 GPU 异构平台上的研究现状进行详细分析, 总结当前基于 GPU 平台高性能密码学面临的技术难题, 并对未来技术发展进行展望. 通过深入研究和总结, 旨在为密码工程研究从业者提供有关基于 GPU 的高性能密码计算的最新研究进展和应用实践的综合参考.

关键词: 公钥密码; 后量子密码; 同态密码; 并行计算; GPU 加速

中图法分类号: TP306

中文引用格式: 董建阔, 黄跃花, 付宇笙, 肖甫, 郑昉昱, 林璟锵, 董振江. 基于异构多核心GPU的高性能密码计算技术研究进展. 软件学报, 2024, 35(12): 5582–5608. http://www.jos.org.cn/1000-9825/7089.htm

英文引用格式: Dong JK, Huang YH, Fu YS, Xiao F, Zheng FY, Lin JQ, Dong ZJ. Research Progress in High-performance Cryptographic Computing Technology Based on Heterogeneous Multicore GPUs. Ruan Jian Xue Bao/Journal of Software, 2024, 35(12): 5582–5608 (in Chinese). http://www.jos.org.cn/1000-9825/7089.htm

Research Progress in High-performance Cryptographic Computing Technology Based on Heterogeneous Multicore GPUs

DONG Jian-Kuo[1], HUANG Yue-Hua[1], FU Yu-Sheng[1], XIAO Fu[1], ZHENG Fang-Yu[2], LIN Jing-Qiang[3], DONG Zhen-Jiang[1]

[1](School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China)

[2](University of Chinese Academy of Sciences, Beijing 100093, China)

[3](School of Cyber Science and Technology, University of Science and Technology of China, Hefei 230026, China)

Abstract: As the core foundation for ensuring network security, cryptography plays a crucial role in data protection, identity verification, encrypted communication, and other aspects. With the rapid popularization of 5G and the Internet of Things technology, network security is facing unprecedented challenges, and the demand for cryptographic performance is showing explosive growth. GPU can utilize thousands of parallel computing cores to accelerate complex computing problems, which is very suitable for the computationally intensive nature of cryptographic algorithms. Therefore, researchers have extensively explored methods to accelerate various cryptographic algorithms on GPU platforms. Compared with platforms such as CPU and FPGA, GPU has significant performance advantages. This study discusses the classification of various cryptographic algorithms and GPU platform architecture, and provides a detailed analysis of current research

* 基金项目: 江苏省重点研发计划 (BE2022798); 国家自然科学基金 (62302238); 江苏省自然科学基金 (BK20220388); 江苏省高等学校基础科学 (自然科学) 研究面上项目 (22KJB520004); 中国博士后科学基金 (2022M711689); 公安部技术研究计划 (201JSYJD03)
收稿时间: 2023-05-11; 修改时间: 2023-09-08; 采用时间: 2023-11-13; jos 在线出版时间: 2024-03-13
CNKI 网络首发时间: 2024-03-15

on various ciphers on GPU heterogeneous platforms. Additionally, it summarizes the current technical challenges confronted by high-performance cryptography based on GPU platforms and provides prospects for future technological development. Finally, comprehensive references can be provided for practitioners in cryptography engineering research on the latest research progress and application practices of high-performance cryptography based on GPU by in-depth studies and summaries.

Key words: public key cryptography; post quantum cryptography; homomorphic cryptography; parallel computing; GPU acceleration

随着数字化、网络空间的不断扩展, 云计算、物联网、移动互联网、线上交易、电子政务等需要密码技术保证数据安全的应用领域不断发展. 应用服务的有序开展, 既要保证服务的高效可用, 又要保证隐私数据信息的机密性、完整性、不可抵赖性等信息安全目标, 密码技术是实现信息安全目标的有力支撑. 基于密码技术实现的安全协议已应用到许多领域, 超文本传输协议 (hyper text transfer protocol secure, HTTPS)[1]基于安全套接层 (secure sockets layer, SSL)[2]协议或传输层安全 (transport layer secure, TLS)[3]协议和数字证书来保证传输数据的安全, 广泛应用到银行、电子商务、电子政务等涉及敏感信息的领域中, 以实现端到端的通信安全. 不同类型的信息都需要加密来保护其机密性, 例如个人信息、商业机密、国家机密等. 因此, 密码技术的重要性不言而喻.

现代密码学是研究如何保证信息的安全性, 防止信息在传输或存储过程中被攻击者窃取或篡改的一门科学, 可以分为对称密码、杂凑密码、传统公钥密码、后量子密码和同态密码等一些前沿密码技术. 对称密码加密和解密使用相同密钥, 适合对数据进行加密和解密. 在对称密码中, 加密和解密过程都是快速的, 但是密钥管理困难, 如果密钥被泄露, 加密的安全性将被破坏. 因此, 对称密码在安全性和密钥管理方面存在着一定的矛盾. 公钥密码使用不同密钥进行加密和解密. 与对称密码相比, 公钥密码具有更高的安全性, 但加密和解密的速度较慢. 由于公钥密码的特殊性质, 它能够应对对称密码无法解决的问题, 例如密钥分发和数字签名等. 杂凑密码用于实现数据的认证性和完整性验证. 由于量子计算机的强大计算能力, 传统的对称密码和公钥密码都存在被破解的风险. 后量子密码是基于量子计算机无法破解的密码算法, 其研究将是未来密码学领域的重点方向, 目的是在量子计算机出现后保障信息安全. 同态密码是一种特殊的加密技术, 可以在密文的基础上执行计算并生成新的密文, 最后将其解密得到与原始明文相同的结果. 同态密码技术可以保护数据隐私, 同时在云计算和大数据处理中具有广泛的应用前景. 同态密码技术的发展和应用将为数据安全和隐私保护提供更多的选择和保障. 除了以上分类外, 现代密码学还包括密码分析、密码协议和密码标准等方面. 密码分析是指攻击者利用密码算法的潜在弱点进行密码破解和攻击的过程, 包括攻击者获取密钥的技术和手段. 密码协议是指在安全通信和身份认证中使用的一系列协议. 现代密码学在数据保护、通信安全和隐私保护等方面都发挥着至关重要的作用.

图形处理器 (graphics processing unit, GPU) 是一种专门用于图形处理和高性能计算的处理器, 适用于并行计算和大规模数据处理. GPU 不仅用于计算机图形学, 还用于各种需要大规模数据处理的场合. 按照架构和使用场景, GPU 可以分为以下几类: (1) 专业级 GPU: 专业级 GPU 通常是针对科学计算、图形渲染和机器学习等需要高性能计算的领域而设计的, 从硬件角度来看, 专业级 GPU 拥有双精度浮点数加速模块, 并具有更大的内存与更高的主频. (2) 消费级 GPU: 消费级 GPU 常被用于进行图形渲染、机器学习和深度学习等计算密集型任务, 具有超强的定点数计算能力, 同样具备单独的人工智能加速模块 (tensor core), 缺乏双精度浮点数独立加速硬件, 但具有更高的性价比. (3) 嵌入式 GPU: 通常用于移动设备、物联网设备和其他嵌入式系统中, 具有较小的体积和功耗. 相比普通显卡, 嵌入式 GPU 做了硬件裁剪, 尤其是寄存器数量做减半处理, 嵌入式 GPU 的设计旨在提供高效的图形处理能力, 同时最小化功耗和空间占用. 随着科学技术的发展, GPU 的发展从早期的单一功能图形加速器到现在的通用并行计算设备. GPU 的应用范围不断扩大, 分类也越来越多样化. 随着新技术的出现, GPU 还将继续发挥着越来越重要的作用.

GPU 在密码学中的应用广泛, 其优势主要体现在大规模数据并行计算方面, 基于 GPU 实现密码算法能够大幅提高密码算法的计算速度. 因此, 基于 GPU 实现密码算法已成为当前密码学领域研究的一个热门方向. 一些密码学算法已经成功移植到 GPU 平台上, GPU 还可以被用于加速数字签名、消息认证等常见的密码技术. GPU 作为高性能的计算设备, 在密码学中的应用越来越广泛, 然而基于 GPU 的密码算法实现也存在一些挑战. 首先, GPU 的存储容量通常相对于 CPU 内存较小, 需要合理地设计算法以充分利用 GPU 的计算能力; 其次, 基于 GPU

的密码算法实现需要注意安全性问题, 防止密码被恶意攻击者破解. 随着技术的发展和应用需求的增长, 基于GPU 实现密码算法的研究具有越来越大的理论和应用价值. 同时, 需要在实现算法的高效性和安全性之间做出权衡, 以充分利用 GPU 的并行计算能力, 提高密码算法的计算速度和安全性.

本文第 1 节介绍密码技术的相关背景和政策. 第 2 节介绍本文所需的基础知识, 包括现代密码学方向和GPU 分类及其并行加速. 第 3 节介绍基于 GPU 的各类密码加速研究现状. 第 4 节介绍本研究方向的展望. 最后总结全文.

1 背 景

在当今数字化时代, 密码技术在网络空间安全中扮演着重要的基石作用, 可用于确保数据的机密性、完整性和可用性. 我们每天都产生和传输大量的个人隐私信息, 例如银行卡信息、各种应用程序的账户密码和医疗记录等, 密码技术可以确保个人信息的安全性. 随着电子商务、在线支付、移动互联网等互联网应用的快速发展, 在海量数据即时安全传输需求下, 需要高吞吐、低延时的高性能密码计算技术来保障交易和通信的安全性, 以保证敏感数据在传输过程中不会被窃取或篡改. 在国家安全方面, 密码技术更是不可或缺的, 密码技术为其提供可靠的保密机制. 此外, 密码技术还可用于实现数字签名和数字证书等认证机制, 以确保通信双方都是合法的. 随着数字化时代的不断发展, 密码技术的重要性也将变得越来越不可替代.

随着《网络安全法》《密码法》《数据安全法》和《个人信息保护法》的相继颁布, 网络空间安全和数据安全已经上升到一个新的高度. 这些法规的制定和实施, 为网络空间安全和数据信息保护提供了坚实的法律保障和指导, 促进了网络空间安全规范化和健康发展.《数据安全法》的出台旨在促进数据的开发和利用, 对于数据的收集、存储、使用、加工、传输、提供和公开等相关概念进行了明确的定义, 要求企业在使用数据的过程中遵守相关规定, 保护数据的合法性和安全性, 以有效遏制数据信息泄露和滥用现象的发生. 据报道, 2019 年, 数据隐私安全相关事件达到 7098 起, 涉及 51 亿条数据, 比 2018 年增长了 284%. 这说明了数据安全的严峻形势, 数据安全事件的影响不仅局限于个人信息泄露, 还可能导致公司的商业机密泄露, 进而损害企业的声誉和利益. 根据数据显示, 2020 年全球数据泄露的平均损失为 1 145 万美元, 数据安全事件对个人、企业和国家安全带来的影响是深远而长久的. 2023 年 4 月, 国务院常务会议审议通过《商用密码管理条例修订草案》[4], 表明商用密码在保障网络和信息安全、保障公民和企业利益等方面的重要性日益凸显. 近年来, 我国的密码产业市场规模不断扩大, 平均增速高于全球增速, 预计在 2023 年, 我国商用密码市场总体规模将达到 985.85 亿元, 同比增长 39.32%. 密码技术是保障网络信息安全和数据安全的核心技术和基础支撑, 是网络空间安全防护的基石. 密码技术不仅可以保障数据信息的机密性、完整性、认证性、可用性和不可否认性, 同时还能有效地防范黑客攻击、网络钓鱼、恶意软件等安全威胁, 为网络空间安全的发展提供了坚实的技术保障.

随着云计算、大数据、移动互联网、物联网、人工智能、车联网、边缘学习、分布式计算、隐私计算等技术的快速发展, 世界正逐步迈入万物互联的泛在智能数字化时代. 在云-边-端三侧协同进行边缘学习模式中, 例如车联网智能化场景, 面临计算迁移、隐私保护、访问控制等诸多问题. 当计算数据需要从物联网终端设备迁移到边缘服务器或云服务器时, 存在隐私数据泄露风险、身份认证、访问控制等问题. 在这方面, 密码算法和密码协议发挥着不可或缺的作用. 基于数字签名、实体认证、数字证书等技术, 建立公钥基础设施 (public key infrastructure, PKI)[5], 通过双向的认证机制, 在云、边、端三侧实现身份认证和访问控制, 避免遭受未授权访问或越权访问攻击; 基于哈希算法、公钥密码、对称密码等技术的结合使用, 对信道中传输的数据包进行加密和签名, 防止数据在传输过程中被窃取或篡改, 保证数据的安全可信; 基于同态密码等隐私计算技术, 实现数据计算过程中的隐私保护, 打破数据孤岛鸿沟, 达到“数据可用不可见”; 基于后量子密码算法, 以抵抗未来随着量子计算技术发展而可能产生的量子攻击. 在上述的车联网智能化场景中, 物联网终端设备通常设计为小型化、低功耗、低成本, 以满足其特定场景的要求. 因此, 与普通计算机相比, 物联网终端设备具有资源受限的特点——其处理能力和存储容量通常较低. 云服务器、边缘服务器、物联网终端设备之间面临海量数据传输, 存在巨大的身份认证、访问控制及数据机

密性、完整性等需求. 针对物联网设备资源受限的特点及云-边-端三侧存在的安全需求, 高性能密码算法实现的重要性逐渐凸显.

目前, 高性能密码算法的实现主要有两种方式, 一是基于硬件加速器, 如 GPU、现场可编程门阵列 (field-programmable gate array, FPGA)、专用集成电路 (application specific integrated circuit, ASIC)、数字信号处理 (digital signal processing, DSP) 处理器等; 二是优化密码算法的实现, 针对特定的密码算法或密码协议模块, 设计更优的算法实现, 也可以通过并行化技术、矢量化技术等方式提高算法的效率. 硬件实现比软件实现的速度更快, 但学习成本和开发成本相对较高. 基于 GPU 的高性能密码算法实现在研究中受到广泛关注, 具有较高的并行性和灵活性, 能够显著提高密码算法的执行速度. 因此, 基于 GPU 的高性能密码算法实现已经成为当前密码技术研究领域的热点之一.

GPU 是一种专门用于高效执行数据处理任务的专用处理器, 也是显卡的核心芯片, 在深度学习、科学计算、游戏等领域被广泛应用. 与 CPU 不同的是, GPU 拥有大量的速度较慢计算核心, 其晶体管主要用于算术逻辑单元而不是数据缓存和流程控制. 因此, GPU 比 CPU 具有更强的并行计算能力, 能够同时处理大量数据. 在特定的应用场景下, GPU 可以提供比 CPU 更高的运算速度和性能. GPU 编程是指使用 GPU 进行并行计算的编程技术, 通常使用的编程语言包括 CUDA (compute unified device architecture, 统一计算设备架构)[6]、OpenCL (open compute language, 开放计算语言) 等. 其中, CUDA 是 NVIDIA 公司推出的通用并行运算架构, 用于在 NVIDIA GPU 芯片上进行 GPU 编程. 在深度学习、图像处理、科学计算等领域, GPU 编程已经成为必不可少的技术, 基于 GPU-CPU 的异构计算逐渐发展为高性能计算 (high performance computing, HPC) 领域的主流模式. 近年来, 基于 GPU 的高性能密码算法实现研究已经取得了重要进展. 通过使用 CUDA 技术, 将密码算法的部分模块实现映射到 GPU 上, 可以实现高效的并行化运算, 相比于 CPU 软件实现, 具有巨大的性能突破.

2 基础知识

本文主要介绍基于异构多核心 GPU 的高性能密码算法实现的研究进展, 下面就相关概念和基本知识予以介绍.

2.1 现代密码学方向

1949 年, Shanon 发表文献 [7], 标志着现代密码学的开端, 密码学正式成为一门科学. 1976 年, Diffie 和 Hellman 发表文献 [8], 为现代密码学的发展指引了新方向. 在保密通信系统模型中, 包含六元组: 明文 m、密文 c、加密算法 E_{k1}、解密算法 D_{k2}、加密密钥 k_1、解密密钥 k_2. 图 1 是现代密码学的基本方向, 具体包括加解密密钥相同的对称密码; 生成固定长度散列值的杂凑密码; 基于整数分解和离散对数困难问题的传统公钥密码; 为应对量子计算攻击的后量子密码; 同态密码等服务隐私计算的新型密码.

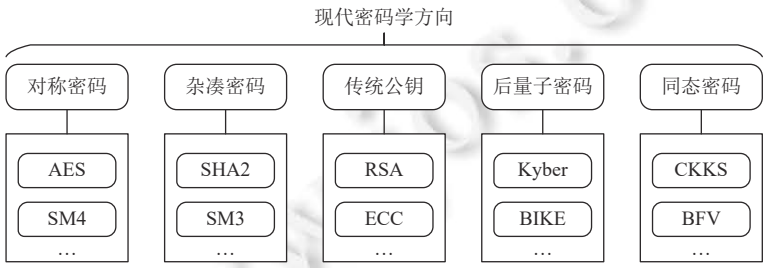

图 1 现代密码学方向

(1) 对称密码

Shannon 在文献 [7] 中, 首次将信息论引入到密码学的研究中, 建立了保密通信系统的数学理论, 为现代密码学的发展奠定了数学基础, 并提出了对称密码体制, 这是现代密码学的第 1 次飞跃. 对称密码体制也被称为共享密钥密码体制. 它使用同一个秘密密钥对明文进行加密和密文进行解密, 这个密钥必须在发送者和接收者之间事先

共享, 因此称为“共享密钥”. 对称密码算法通常分为两种类型: 流密码 (序列密码) 和块密码 (分组密码). 流密码是将明文流和密钥流通过某种算法一起处理成密文流, 其基本思想是利用密钥 k 产生密钥流 z, 并使用特定规则对明文串 m 进行加密得到密文串 c, 解密时也同步产生的同样的密钥流 z 实现解密; 块密码则是将明文分成固定长度的块, 每块分别在密钥的控制下变换成等长的输出序列.

常见的对称密码算法有 DES[9]、AES[10]、SM4[11]、RC4[12]、ZUC[13]等, 这些算法都是经过广泛研究和分析的, 具有较高的安全性和可靠性. 对称密码算法加密速度快, 加密和解密效率高等优点, 但是对称密码的缺点是密钥需要安全地分发, 且在多个参与者之间共享密钥时存在安全隐患. 因此, 通常事先通过非对称密码算法在多个实体之间安全地协商出共享密钥, 然后再用于对海量数据的加密操作.

(2) 杂凑密码

杂凑密码是一种密码学中常用的密码算法, 用于将任意长度的消息转换为固定长度的摘要, 设计要满足单向性、抗碰撞性、抗篡改性等安全性要求, 输出的长度通常是 128 位、160 位、256 位或更长, 具体取决于所采用的算法. 杂凑密码的概念最早由 Merkle[14]于 1979 年提出, 但 Rivest 在 1990 年才提出了第 1 个杂凑密码算法 MD4[15], 之后, MD5[16]、SHA-1、SHA-2 等算法相继问世.

杂凑密码在密码学领域中得到了广泛应用, 可以用于数字签名、消息认证、消息摘要等方面. 随着计算机硬件和算法的不断发展, 杂凑密码也在不断演进. 其中, MD5 和 SHA-1 等算法由于安全性缺陷而逐渐被淘汰, SHA-2、SHA-3 等新一代算法逐渐成为主流. 除了常规的杂凑密码算法外, 还有一些特殊的杂凑密码算法, 如 HMAC (基于哈希的消息验证码)[17]、RIPEMD[18]和 Whirlpool[19]等. 这些密码算法在不同的场景下具有不同的优势和适用性. 杂凑密码是保护信息安全的基础之一, 在未来信息时代中仍将扮演重要角色.

(3) 传统公钥密码

传统的对称密码体制由于采用了相同的密钥进行加解密, 导致其在密钥分发以及密钥管理等方面存在一些问题. 鉴于此, 公钥密码这一思想被提出, 并且被广泛地应用于数据保护、数字签名、密钥交换等各种领域, 在各个领域都有着十分广泛的应用.

1976 年, 美国斯坦福大学的 Diffie 与 Hellman 发表文献 [8], 正式提出了公钥密码思想. 公钥密码体制大致分为 3 类, 分别是基于大整数分解困难问题的公钥密码, 代表密码算法有 RSA 算法[20]和 Rabin 算法[21]; 基于有限域离散对数困难问题的公钥密码[22], 代表密码算法有 ElGamal 算法[23]及其变种数字签名算法 (digital signature algorithm, DSA)[24]; 基于椭圆曲线离散对数困难问题的公钥密码, 代表密码算法有 ECDSA[24]、EC-KCDSA[25]、Curve25519[26]、国产 SM2 算法[27]等. 椭圆曲线密码 (elliptic curve cryptography, ECC) 利用椭圆曲线上的点来完成加密和解密操作. 椭圆曲线密码的加密过程是将明文转换为椭圆曲线上的点, 再进行数学运算得到密文; 解密过程则是利用私钥来逆向数学运算得到明文. 椭圆曲线密码算法具有很多优点, 例如密钥尺寸相对较小、加解密速度快、安全性高等, 特别适用于资源受限的物联网设备、移动通信设备等, 因此在许多领域得到了广泛的应用, 是目前被广泛认可的安全性最高的公钥密码学算法之一.

(4) 后量子密码

随着量子计算机技术的不断进步和发展, 传统的加密算法已经无法保证数据的安全性, 基于 Shor[28]和 Grover[29]的量子计算能在多项式时间内破解目前广泛使用的公钥密码学 (如 RSA 和 ECC). 因此后量子 (post quantum cryptography, PQC) 密码逐渐成为研究的热点. 国内外有很多科研机构和企业着手研究量子后密码学, 并建立了相关平台. 其中, 中国科学技术大学成立了量子安全与信息中心, 研究量子密码学及其在信息安全中的应用; 北京大学也成立了量子安全中心, 旨在推动国内量子安全领域的研究和发展; 中国科学院信息工程研究所也在这一领域投入了大量的人力和物力资源.

美国国家标准与技术研究院 (national institute of standards and technology, NIST) 于 2016 年便启动了全球范围内的后量子密码标准征集, 以抵抗量子计算攻击. 2022 年, NIST 公布了后量子密码学标准化流程的第 3 轮结果, 包括 4 种选定的算法 (CRYSTALS-Kyber[30]、CRYSTALS-Dilithium[31]、Falcon[32]、SPHINCS+[33]) 和 4 种候选算法 (BIKE、Classic McEliece、HQC、SIKE). 这些算法都是基于比传统公钥密码学更复杂的数学难题, 并且没有已

知的经典算法和量子算法可以有效攻克这些问题. 目前主要有 4 种实现方案: 基于格 (Lattice-based cryptography, LBC)、基于编码 (code-based cryptography, CBC)、基于哈希 (hash-based cryptography, HBC) 和基于多变量 (multivariate-based cryptography, MBC) 的后量子公钥密码方案. 在选定的算法中, 大部分的算法 (CRYSTALS-Kyber、CRYSTALS-Dilithium、Falcon) 都是基于格的密码方案, 只有一个算法 (SPHINCS+) 是基于哈希的密码方案. 其中, CRYSTALS-Kyber 是唯一的公钥加密和密钥建立算法.

(5) 同态密码

随着大数据、云计算、人工智能等新兴技术的发展, 传统防御技术无法有效保障信息安全、避免关键数据泄露. 同态加密技术作为密码学的“圣杯”, 为安全隐私保护提供了新的思路: 应用服务直接作用于加密数据之上, 在这一技术路线下攻击者即使成功突破防御边界入侵系统也无法获取原始关键数据, 从根源上降低了信息泄露的风险. 相较于其他隐私保护方法, 如联邦学习、安全多方计算、机密计算、差分隐私等, 基于同态加密技术的隐私保护方法更具有普适性, 可以适配多种不同的应用方向, 在云计算、人工智能、区块链等领域有极大的发展潜力.

Rivest、Adleman 和 Dertouzos 这 3 位科学家在 1978 年首次提出同态加密[34]的概念: 对数据密文进行运算并解密得到的结果与明文运算结果一致. 此后相关学者就提高计算速度、缩短密文长度、扩展数据类型、扩大支持操作等方面进行研究并取得了大量的研究成果. 2009 年 Gentry[35]基于理想格首次构建了全同态加密方案, 并在次年实现了在 IBM x3500 服务器下的加法与乘法运算[36], 但性能远远无法达到实际落地的需求. 近年来, BGV[37]、BFV[38]、CKKS[39]等一系列具有代表性的全同态加密方案在工业界和学术界都得到了广泛应用, 但巨大的性能开销仍然是制约同态加密技术进一步发展的主要瓶颈. 当前, 对提高同态加密技术运算速度的迫切需求, 引起了业内的广泛关注.

以往同态加密技术的应用与研究大多依赖运行在 CPU 平台上的库, 例如著名的 SEAL[40]和 HElib[41], 这些库更多的关注可用性与兼容性. 根据最新的研究[42], 基于 SEAL 和 HElib 的数据密文计算性能相较于明文下降了 10^5–10^7 倍. 而一些利用平台特性来进行性能优化的方法, 受限于 CPU 平台性能的限制, 无法得到有效的提升. 近年来, GPU 的快速发展为同态加密的性能困境带来了新的解决思路. 2006 年 NVIDIA 在 GPU 中引入了 SIMT 架构, 具有多个独立的线程使用一条指令并发执行的线程级并行特性, 为需要大量算术运算的同态加密性能优化带来了巨大的发展空间.

2.2 GPU 分类与并行加速

随着高性能计算与人工智能等数据密集型领域的需求, GPU 已经从早期的图形处理器演化成为拥有强大并行计算能力和存储器带宽的多核处理器. 与 CPU 相比, GPU 专注于计算密集型的大规模数据并行计算, 其更多的晶体管作为算术逻辑单元, 而不是用于数据缓存或流程控制. 因此, GPU 特别适用于数据并行计算问题, 即同一段程序指令在多条数据上执行. 这类操作对不同的数据执行相同的操作, 对复杂流程控制的要求低, 符合 GPU 芯片硬件设计的特点. 此外, GPU 能够提供更高的指令吞吐量和带宽, 在价格和功耗相近的情况下, 比 CPU 更加优越.

GPU 设计之初, 主要用于计算机的图形处理, 如在 3D 渲染中通过大量线程对同一帧画面中的像素点并行运算. 此后, 逐渐发展成为通用计算和人工智能加速的重要组成部分, 许多处理大规模数据的应用也适合利用 GPU 并行计算优势提升计算速度, 从图形处理、视频渲染, 到科学计算、机器学习、自动驾驶、高性能计算等新兴场景, 都可通过数据并行运算实现加速, 这种计算技术称为 GPGPU (general-purpose computing on GPU)[43].

利用 GPU 并行计算优势来满足高性能运算的需求越来越大. 2006 年, NVIDIA 公司推出了 CUDA, 以实现 GPU 上的并行计算程序的开发. CUDA 是通用并行运算设备架构和编程模型, 基于 CUDA 编程可以利用 GPU 的并行计算引擎来更加高效地解决比较复杂的计算难题, 研究人员可以非常方便地利用 NVIDIA GPU 来完成各类并行计算程序的开发, 大大降低了利用 GPU 强大计算能力开发程序的门槛. CUDA 工具包提供了利用 GPU 加速优势开发高性能并行计算程序的开发环境, 包括 GPU 加速库、调试、优化、C/C++编译器、CUDA 运行时 API、CUDA 驱动 API 等. 目前, CUDA 支持利用 C/C++、Fortran、Java、Python 等编程语言来完成 GPU 并行计算程序的开发. CUDA 指令集包括了大量的并行计算指令和内存操作指令, 其中最常用的指令包括: 矩阵乘法指令, 向量

加法指令, 卷积运算指令, 归约运算指令, 分支和循环指令, 除此之外, CUDA 还提供了大量的内存操作指令, 例如内存读写、共享内存操作等. 这些接口方便了编程人员对 GPU 实现高效利用, 完成复杂的大规模运算任务的计算[44]. 在基于 CUDA 的 CPU-GPU 异构计算编程模型中, CPU 被称为主机端 (host), GPU 被称为设备端 (device), 将 GPU 看作 CPU 的协处理器, 两者通过总线传输数据. 主机端以串行的方式执行程序, 调用 CUDA 内核函数, 将需要执行并行计算的数据从主机端拷贝到设备端. 在设备端, 多个线程以并行的方式执行代码, 处理相关数据, 最后将计算结果送回主机端. 同时执行 CUDA 程序时还需注意线程同步问题. 通常, CUDA C/C++程序的执行流程如下: 1) 在主机端调用 CUDA API 申请设备内存 (cudaMalloc()); 2) 将待处理数据从主机内存拷贝到设备内存 (cudaMemcopy()); 3) 调用 CUDA 核函数并行处理数据 (kernel<<<blockSize, threadSize…>>>()); 4) 将计算结果从设备内存拷贝到主机内存 (cudaMemcopy()); 5) 释放申请的设备内存 (cudaFree()). 在 CUDA 线程执行期间, 可能需要从多种存储空间中访问数据. 每个线程都拥有自己的本地内存 (local memory); 每个线程块都具有共享内存, 该共享内存对于同一块内的所有线程可见, 且具有与该块相同的生命周期. 线程块簇较为集中的不同线程块可以对彼此的共享内存执行读取、写入和原子操作. 此外, 所有线程都可以访问相同的全局内存 (global memory). 还存在两个额外的只读内存空间, 供所有线程访问: 常量内存 (constant memory) 和纹理内存 (texture memory). 全局、常量和纹理内存空间都经过了针对不同内存使用情况的优化, 纹理内存还为某些特定数据格式提供不同的寻址模式和数据过滤功能. 在同一应用程序的内核启动过程中, 局、常量和纹理内存空间是持久存在的.

GPU 在密码算法中的应用主要是由于 GPU 具有大量的并行计算单元, 可以同时处理大量的数据, 这对于密码学中的加密和解密操作非常有用. GPU 还可以通过 CUDA 等 API 模型来进行并行计算, 这使得 GPU 在密码学中的应用更加广泛. GPU 的并行性体现在它的架构上. GPU 的架构是由大量的处理单元组成, 这些处理单元可以同时处理多个数据. GPU 的架构被组织成多个可扩展的多线程阵列, 被称为多线程流多处理器 (streaming multiprocessors, SMs). 每个 SM 包含多个 CUDA 核心, 每个 CUDA 核心都可以执行一个线程[45]. 在 CUDA 编程模型中, 多个线程 (thread) 构成一个线程块 (block), 多个线程块构成一个网格 (grid), 以 grid-block-thread 的层次结构管理线程. CUDA 内核 (kernal) 函数使用关键字__global__标识, 使用配置语法<<<blockSize, threadSize>>> 指定执行内核函数调用的线程块数量和线程块中的线程数量. 与普通的函数不同, 当调用 CUDA 内核函数时, 将被 N 个线程并行执行一次. 当主机执行 CUDA 程序调用核函数时, 计算网格核心中的线程块被分配到 SM, 同一个线程块中的线程并发运行在同一个 SM 上, 多个线程块中的线程也可以运行在同一个 SM 上. 当一个线程块完成执行后, 新的线程块将在空闲的 SM 上启动. 每个 SM 以单指令多线程 (single instruction, multiple thread, SIMT) 的独特架构同时执行数百个线程, 指令流水线式执行, 利用单个线程内的指令级并行性, 并通过硬件多线程来实现线程级并行性. SM 以 32 个并行线程为一组 (称为 warps) 执行线程相关操作. 当一个或多个线程块被分配给 SM 执行时, SM 将线程分割成最小执行单元, 称为线程束 (warp), 然后由线程束调度器进行调度. 线程束内的每个线程具有连续递增的线程 ID, 其中第一个线程束包括线程 0. 线程束内的各个线程从相同的程序地址开始执行, 但它们拥有各自的指令地址计数器和寄存器状态, 因此可以独立地分支和执行. 线程束是 GPU 中最小的并行执行单位, 当线程束内的 32 个线程都执行相同的路径时, 可以实现最大化的并行执行效率. 当线程束内的线程遇到条件分支时, 即线程间的执行指令不同时, 线程束会分别执行线程的分支路径. 不同的线程束独立执行, 不会出现分叉情况. 在线程束的整个生命周期内, 每个线程束的执行上下文 (程序计数器、寄存器等) 都在片上内存中维护, 因此, 从一个执行线程束切换到另一个执行线程束不需要额外的开销. SIMT 架构与单指令多数据 (single instruction, multiple data, SIMD) 向量组织类似, 使用单一指令控制多线程以并行处理多个数据. 其中一个区别在于 SIMD 向量组织向软件公开 SIMD 宽度, 而 SIMT 指令则指定单个线程的执行和分支行为. 与 SIMD 向量机相比, SIMT 允许程序员编写面向独立标量线程的线程级并行代码, 以及协调线程的数据并行代码.

GPU 架构的演变历史包括 Fermi 架构、Kepler 架构、Maxwell 架构、Pascal 架构、Volta 架构和 Turing 架构. 现在 GPU 的架构包括 Ampere 架构和 Turing 架构. Ampere 架构是 NVIDIA 的第 2 代 RTX GPU 架构, 它是 Turing 架构的后继者, 其主要特点是更高的性能和更低的功耗. NVIDIA 根据不同应用场景来区分不同类别的 GPU, 大体上可以分为: (1) 专业图形可视化: NVIDIA RTX 和 NVIDIA Quadro 系列专业 GPU. (2) 数据中心: NVIDIA

A30 tensor core GPU 以及过往的 Tesla 系列 GPU 等. (3) 消费级娱乐游戏: NVIDIA GeForce 全系列. 除此之外, 不同的 GPU 还有不同的架构和规格, 例如核心数量、显存大小、显存带宽等, 这些规格的不同也会影响 GPU 的性能和适用场景. 目前, 全球 GPU 市场主要由 NVIDIA、AMD、Intel 等几家大型企业垄断. 其中 NVIDIA 是全球最大的独立显卡厂商之一, 其产品广泛应用于游戏、人工智能等领域; AMD 是一家美国半导体公司, 其显卡产品也广泛应用于游戏、人工智能等领域; Intel 则是全球最大的半导体生产商之一, 其产品涵盖了 CPU、GPU 等多个领域. 国产 GPU 的发展现状和市场潜力已经得到了一定的探索和发展, 但是相对于 NVIDIA 和 AMD 等国外巨头来说还有一定的差距.

3 基于 GPU 的密码加速

GPU 性能优化主要涉及 4 个基本策略: (1) 最大化并行执行以提高利用率. (2) 优化内存使用以提高内存吞吐量. (3) 优化指令使用以提高指令吞吐量. (4) 尽量减少内存抖动. 选择哪种基本策略来优化应用程序的性能取决于特定程序模块的性能受限瓶颈. 例如, 对于主要受限于内存访问的内核, 优化指令使用可能不会显著提升性能. 因此, 在进行性能优化工作时, 需要通过测量和监控性能限制因素 (例如使用 CUDA 分析器) 不断调整优化策略. 还需要将特定内核的浮点运算吞吐量或内存吞吐量与设备的相应峰值理论吞吐量进行比较, 确定内核性能的改进潜力.

最新的 GPU 架构旨在满足机器学习和图像处理等应用的原始需求, 其在程序的并行执行、指令类型和内存结构等方面实现了快速迭代. 然而, 现有的密码计算方法在软硬结合计算技术方面存在并行映射不合理、指令选型不适配、访存技术落后等问题, 这些由于 GPU 计算架构更新带来的技术隔阂使得传统加速方法难以充分利用 GPU 的高效计算能力. 从并行计算的角度来看, 最新的 GPU 增加了流处理器 SM 的数量, 但主频方面并没有得到有效改进, 这对现有单线程处理密码计算的方案非常不利. 从指令执行的角度来看, 最新的 GPU 架构提升了平台的矩阵运算和浮点数计算能力, 但降低了定点数计算效率, 并且由于浮点数在大整数运算、比特位操作等方面存在固有缺陷, 尤其是大多数密码学参数在设计过程中通常将有限域确定为 32 比特的整数倍, 这不仅影响现有研究的性能, 还会加剧基于最新 GPU 架构的密码计算方案的设计复杂度. 从内存访问的角度来看, 最新 GPU 架构显著增加内存空间大小, 并提供了最新的线程通信技术和缓存复用技术, 然而, 在标量乘法窗口选型、并行线程数据交互、共享内存访问架构等方面, 传统实现与最新架构之间不能高效适配, 甚至可能出现指令不兼容的情况. 要解决 GPU 架构与密码优化计算之间的技术隔阂, 关键在于重新分析设计密码算法的 GPU 优化模块, 以适应 GPU 的并行性能; 优化指令选型和浮点数计算, 引入特定的优化技术或计算库; 改进内存访问和数据结构, 以减少内存抖动和提高内存访问局部性.

如图 2 所示, 简要概括了基于 GPU 加速密码算法领域的发展简史. 本节主要按照密码类别进行分类, 相比于公钥密码, 对称密码与杂凑密码计算结构更加简单, 在 GPU 平台优化过程中, 一般采用单线程实现完整算法的并行方式, 优化的角度一般为内存与指令层面, 在本文介绍过程中归将对称密码与杂凑为一类. 下面按照密码算法类型, 对该领域的发展现状逐类作详细介绍.

如图 3 所示, 我们统计了 2015 年以来基于 GPU 平台密码加速技术的代表性文章, 文章数量呈现上升趋势, 受到产业应用需求的推动, 传统公钥密码 (ECC、RSA) 一直以来受到广泛关注, 且文章数量保持稳定. 近年来, 同样出现新的研究热点, 尤其是集中在同态密码与后量子密码算法的优化, 我们将在后续的章节中详细展开介绍.

3.1 对称密码与杂凑密码性能加速

2012 年, 德国亚琛工业大学团队[46]通过在 CUDA 和 OpenCL 上以 OpenSSL 加密引擎的形式实现了多个分组密码 (AES、DES、Blowfish、Camellia、CAST5、IDEA), 并进行了基准测试, 提供了一个关于如何执行这些密码和类似 GPU 算法的可复制基准的指南, 为对称密钥 GPU 加密领域做出了贡献. 2014 年, 马来西亚拉曼大学团队在文献 [47] 中在 NVIDIA GTX680 中实现了 Camellia、CAST5 和 SEED, 并介绍了实现技术的细节以及针对现有

解决方案的基准测试结果. 根据评估结果, 在不考虑 CPU 和 GPU 之间的数据传输的情况下, 能够分别为 Camellia、CAST5 和 SEED 实现 61.1 Gb/s、45.5 Gb/s 和 47.4 Gb/s 的吞吐量. 通过考虑数据传输, Camellia、CAST5 和 SEED 的吞吐量分别降至 44.9 Gb/s、40.5 Gb/s 和 38.6 Gb/s. 2016 年, 马来西亚拉曼大学团队[48]提出了在具有 Maxwell 架构的 NVIDIA GTX 980 中加速对称分组密码 (AES-128、CAST-128、Camellia、SEED、IDEA、Blowfish 和 Threefish) 的技术, 在计数器模式 (CTR) 下运行, 能够实现 149 Gb/s (AES-128)、143 Gb/s (CAST-128)、124 Gb/s (Camelia)、112 Gb/s (SEED)、149 Gb/s (IDEA)、111 Gb/s (Blowfish) 和 197 Gb/s (Threefish) 的加密速度. 此外, 当分组密码在计数器模式 (CTR) 下操作时, 它可以用作伪随机数生成器 (PRNG), 但与使用较轻操作的其他 PRNG 相比, 速度通常较慢. 因此, 该团队试图修改 IDEA 和 Blowfish, 以实现更快的 PRNG 生成. 修改后的 IDEA 和 Blowfish 成功通过了所有 NIST 统计测试和 TestU01 SmallCrush, 但 TestU01 (Crush 和 BigCrush) 中更严格的测试除外.

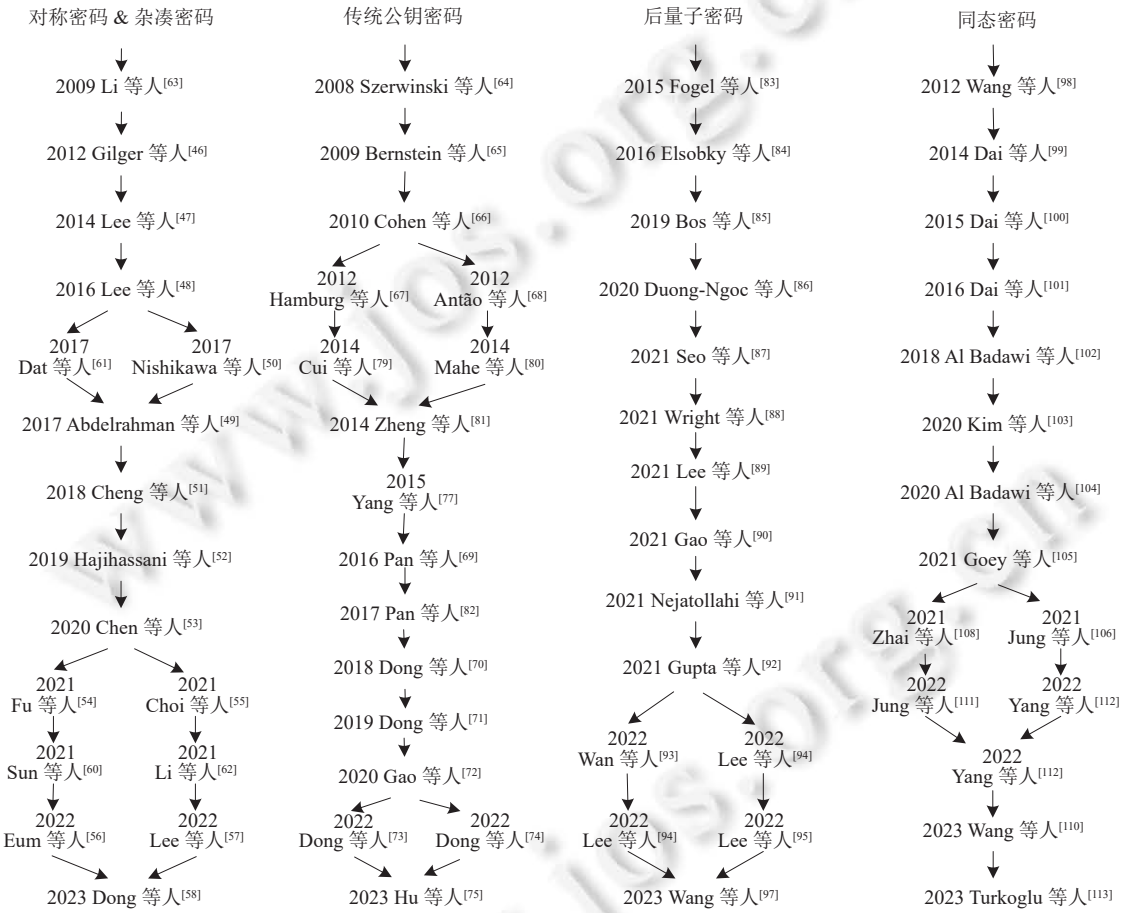

图 2　相关技术研究演进路线

2017 年, 埃及军事技术学院团队在文献 [49] 中提出了在 3 种不同的 GPU 架构 (Kepler, Maxwell 以及 Pascal) 上实现 AES-128 ECB 加密. 结果表明, 通过使用 32 字节/线程粒度执行新的优化技术, 在 NVIDIA GTX Titan X (Maxwell) 上实现了 207 Gb/s 的加密速度, 在 NVIDIA GTX 1080 (Pascal) 上实现 280 Gb/s 的加密速率. 2017 年, 日本庆应义塾大学团队在文献 [50] 中提出了一种在启用 CUDA 的 GPU 上实现 Bitsliced AES 加密的方法, 该方法具有多个参数, 特别关注 3 种并行处理粒度. 根据实验结果, 在 NVIDIA Tesla P100 PCIe 上, Bs64 粒度的位片 AES-ECB 加密的吞吐量达到 605.9 Gb/s, 与基于表的实现相比, 提高了 8.0%. 2018 年, 中国科学院信息工程研究所团队

在文献 [51] 提出了一个高性能的对称密码服务器, 利用指令级实现和可变位置改进, 仔细调度了对称算法 SM4, 并提供了优化方法来加快 CPU 和 GPU 之间低效的数据传输, 可通过网络提供 15.96 Gb/s 的数据加密, 是当时最快对称加密服务器的 1.23 倍. 此外, 对于 IPSec VPN 网关等长期密钥应用, 使用高速预计算技术可以将服务器提升 2.0 倍.

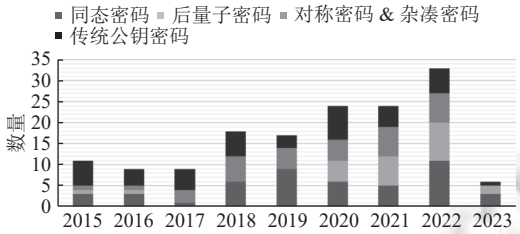

图 3　2015–2023 年 GPU 密码加速代表性文章分布

2019 年, 加拿大阿尔伯塔大学团队在文献 [52] 中提出了一种高吞吐量 Bitsliced AES 实现, 该实现建立在一种新的数据表示方案的基础上, 利用了现代多核心平台的并行化能力. 此外, 通过使用 S 盒逻辑电路, 消除了替换字节阶段使用的基于查找表的 I/O 操作的需要. S 盒逻辑电路被优化为同时处理 32 个 128 位输入数据块. 在 6 个支持 CUDA 的 GPU 上开发了高吞吐量 CTR 和 ECB AES 加密/解密, 在 Tesla V100 GPU 上分别实现了 1.47 Tb/s 和 1.38 Tb/s 的加密/解密吞吐量. 2020 年, 华中师范大学团队在文献 [53] 中利用现代 GPU 架构的并行计算能力, 研究了轻量级分组密码结构的安全粒度, 特别是 SPN 设计, 并展示了如何加速统计鉴别器的计算. 在时间复杂性方面, 该团队提出的方法与传统的 CPU 架构相比具有显著的优势, 并且对其他分组密码具有可扩展性. 2021 年, 长春理工大学团队在文献 [54] 中设计并实现了一种基于 OpenCL 的 GPU 并行 SM4 算法, 并对原有的串行 SM4 算法和基于 OpenCL 的并行 SM4 算法的性能进行了比较和验证. 在最佳情况下, 单 GPU 可以带来约 180 倍的加速效率, 多 GPU 设备可以实现 760 倍的加速效能提升, 实验表明, 使用 GPU 的 SM4 计算可以满足万兆以太网下的实时加密和解密要求. 2021 年, 韩国国民大学团队在文献 [55] 提出了一种基于 GPU 优化的 SHA-3 软件实现方案, 通过 SHA-3 内部进程的优化、内联 PTX 优化、内存的优化使用以及异步 CUDA 流的应用等优化方法, 该方案在 RTX 2080Ti GPU 上的 SHA-3(512) 和 SHA-3(256) 实现提供的最大吞吐量分别为 88.51 Gb/s 和 171.62 Gb/s. 此外, 相较于先前 NVIDIA GTX 1080 上的最佳工作, 在没有使用 CUDA 流的情况下, 基于 NVIDIA GTX 1070 实现的 SHA-3(512) 获得了约 49.73% 的吞吐量提升, 其提出的 GPU 优化的 SHA-3 软件可高效地用于区块链和多种后量子密码方案中.

2022 年, 韩国汉城大学团队在文献 [56] 中对国内常用的 SM4 分组密码进行了 GPU 并行实现. SM4 分组密码具有使用 8 位 Sbox 表的实现和使用 32 位 T 表的实现. 通过测量两个表实现的性能, T 表实现的性能比 Sbox 表实现差大约 0.75 倍. 此外, 使用共享内存实现了 SM4 来获得更好的性能, 结果显示, 在 Sbox 表实现中使用共享内存时, 性能提高了约 1.06–1.19 倍. 2022 年, 韩国嘉泉大学团队在文献 [57] 中创下了 AES 实现的速度记录, 相比于当时最先进的比特切片实现高出 9% (CTR) 和 7% (ECB) 的吞吐量. 此外, 所提出的技术不需要在编译期间将圆形密钥嵌入到代码中. AES 用于在 NIST 后量子密钥封装机制 (KEM) 中生成随机样本, 在 NVIDIA V100、T4 和 RTX3080 GPU 上分别实现每秒 3 350 次、1 503 次和 7 716 次密钥交换. 这使得所提出的 FrodoKEM 实现比最先进的性能快 2.99 倍. 所提出的 AES 实现也被用于详尽的密钥搜索应用程序, 在 NVIDIA V100、T4 和 RTX3080 GPU 上分别实现了每秒 11428×10^6、3969×10^6 和 9998×10^6 次加密. 2023 年, 南京邮电大学团队在文献 [58] 提出了一种基于 GPU 的高效并行加速框架——G-SM3, 通过并行化、内存访问和指令优化这 3 个方面对 SM3 算法进行了优化. 在 NVIDIA Titan V 桌面 GPU 上, G-SM3 的峰值性能达到 23 GB/s, 比顶级服务器 CPU (E5-2699V3) 上 OpenSSL 的性能高出 7.5 倍. 在功耗低于 40 W 的嵌入式 GPU 上, SM3 吞吐量达到 3.8 GB/s, 甚至比服务器级 CPU 的性能还要好. 与其他平台相比, G-SM3 具有巨大优势, 为区块链应用和密码学安全领域提供了高效的加速方法.

如表 1 所示, 详细列出基于 GPU 的对称/杂凑密码在 GPU 的性能优化现状. 受到算法结构的限制, 该类算法

并行策略一般采用单线程单请求的计算方案; 研究人员通过大量的技术手段, 提高上述算法在 GPU 平台的访存效率, 相比 CPU 硬件实现的密码模块, 计算性能取得显著优势.

表 1 基于 GPU 的对称密码与杂凑密码加速性能对比

实现工作	密码类型	实现平台	明文长度 (B)	吞吐性能 (Gb/s)
OpenSSL[59]	SM3	Intel(R) Xeon(R) CPU E5-2699 v3 @ 2.30 GHz	64 8 192	9.52 23.76
Sun等人[60]	SM3	GTX 1080	8 192	84.24
Dong等人[58]	SM3	Titan V	8 192	180.56
Dat等人[61]	SHA-3	GTX 1080	65 536	51.86
Choi等人[55]	SHA-3	GTX 1070	65 536	75.53
Nishikawa等人[50]	AES	Tesla P100	—	605.9
Li等人[62]	SM4	Titan X	—	31.41
Li等人[63]	MD5-RC4	GeForce 9800 GTX	32	0.55
Cheng等人[51]	SM4	GeForce GTX 1080	4 096	15.96
Gilger等人[46]	Camellia-128 AES-128 Blowfish DES IDEA	GeForce GTX 295	8 192	205.48 231.35 236.32 120.78 285.25

3.2 传统公钥密码并行加速

目前, RSA 和椭圆曲线密码 (ECC) 是公钥密码算法中应用最为广泛的两种. RSA 基于大数分解的数学难题, 在数字签名、数据加密等领域得到了广泛应用. 椭圆曲线密码则基于椭圆曲线离散对数问题, 其密钥单位比特提供的安全性相对于 RSA 更高, 尤其适用于移动设备和物联网等领域. 2008 年, 德国波鸿鲁尔大学团队[64]提出了使用 GPU 作为 RSA 和 DSA 密码系统以及椭圆曲线加密加速器的改进. 该团队使用当时最新的 NVIDIA 8800 GTS 显卡, 通过应用剩余数系统能够为基于 1 024 位整数的 RSA 或 DSA 的系统每秒计算 813 次幂指数操作. 除此之外, 该团队还使用雅各比坐标表示的方法, 在基于素数域 P-224 上的椭圆曲线实现了每秒 1 412 次点乘的吞吐量. 2009 年, 美国伊利诺大学芝加哥分校团队[65]使用 Edwards 曲线, 依赖于新的并行加法公式针对高度并行的 GPU 架构进行了精心调整. 通过使用八路模块化运算单元与整数的蒙哥马利形式结合生成预计算表的方式设计出了高效的流水线乘法实现. 实验结果显示在单个 NVIDA GTX 295 上, 对于通用的 280 位模数, 该实现每秒执行 4 188 万次模乘. 2010 年, 美国明尼苏达大学团队[66]比较了两种预计算方法即使用仿射坐标的方法和使用 Lopez-Dahab 投影坐标的方法. 选用 Lopez-Dahab 投影坐标方法使用二进制椭圆曲线的 LSB 不变标量点乘法的 GPU 来对椭圆曲线的标量点乘进行加速. 通过基于 GF(2^{163}) 域的椭圆曲线上进行的实验结果表明该方案能明显减小椭圆曲线点倍加操作的耗时. 2012 年, 美国 Mike Hamburg 团队[67]使用 Tegra-2 作为 GPU 平台实现 Ed25519 的验证签名, 在实现过程中该团队提出了一种新的点压缩算法即将椭圆曲线中的 q 扭转点压缩为域中的单个元素, 与此同时, 在验证签名方面使用 WNAF 线性组合算法结合预计算表的方式使得验签的时间相较于原有方案减少了 60% 以上.

2012 年, 葡萄牙里斯本理工大学团队[68]提出了一种在 GPU 上实现的椭圆曲线 (ECC) 点乘法的并行算法, 该团队提出的方法依靠剩余数系统 (RNS) 在高精度整数运算上提取并行度. 结果表明, 在商用 NVIDIA 285 GTX GPU 中, 底层域基于 224 位的 ECC 最大吞吐量为每秒 9 827 次标量乘法, 最小延迟为 29.2 ms. 该团队还经由分析得出, 对于由较小尺寸的底层有限域支持的 ECC 曲线, 在通用多核上的实现, 可以从所提出的 RNS 方法中获得进一步的优势. 2016 年, 中国科学院信息工程研究所团队[69]提出了一种使用 GPU 加速的通用椭圆曲线签名服务器, 简称 Guess. 该服务器使用一个 NVIDIA GeForce GTX 780Ti 作为计算平台, 可以支持各种 ECC 的签名方案包括 ECDSA 签名方案、韩国密码 EC-KCDSA 方案、国产 SM2 签名方案等. 在设计过程中, 上层该团队使用了自行设计的

PM 实现, 在下层中优化寄存器的使用, 使得每个 GPU 线程消耗最少数量的寄存器. 2018 年, 该团队针对比特位更长的 RSA 公钥密码算法, 提出了利用双精度浮点数实现 Montgomery 乘法的设计方案[70], RSA4096 吞吐性能提升超过 20%. 2019 年, 中国科学院信息工程研究所团队[71]使用 GPU 加速了 X25519/448 的实现. 通过使用大整数高低进位乘法、快速约减以及 Montgomery 算法对其进行加速, 其实验结果显示, 在 GeForce GTX 1080 上, X25519/448 的结果分别达到每秒 286 万次和 35.8 万次操作, 大大超过了之前最快的工作.

2020 年, 中国科学院信息工程研究所团队[72]利用 GPU 的浮点计算能力实现了 Curve25519 和 Edwards25519, 该团队为目标平台定制了各种性能优化方法, 包括与新的基于浮点的计算算法相结合的新颖的大数表示、高效的合并归约策略和曲线级加速. 在该文献中报告了椭圆曲线方法的创纪录性能: 在 Titan V 上, 该团队分别实现了 Edwards-25519 的每秒 721 万次和 7730 万次未知和已知点点乘法运算. 经由实验结果表明, 在 GPU 上使用浮点数进行 ECC 计算要快于使用整数进行 ECC 计算. 2022 年, 南京邮电大学团队[73]面向比特位更高 Curve448 曲线, 基于资源受限的嵌入式 GPU (Jetson TX2) 设计了一种多线程并行的大整数乘法实现方案, 具体如图 4 所示, 实现过程中严格按照单指令多线程的并行计算思想, 最终双线程的性能相比单线程实现, 吞吐性能提升 28%, 计算延时降低 48%, 该大整数乘法方案对其他平台与曲线都具有通用性. 同年, 该团队[74]提出了一种基于嵌入式 GPU 的 FourQ (EG-FourQ) 椭圆曲线公钥密码加速方案. 该团队的实现是第 1 个在 GPU 平台上完整的 FourQ 实现, 包括有限域运算、点运算和标量乘法. 仅依靠 36 W 的功耗, 其标量乘法性能就达到了 1717 kops/s, 延迟为 2.38 ms. 实验结果显示, 就能效比而言, EG-FourQ 与 ARM CPU、Intel CPU、FPGA 和台式 GPU 等其他平台相比具有显著优势. 2023 年, 武汉大学团队[75]在 NVIDIA RTX 3060 平台上实现了 IEEE P1363 标准中基于身份的签名方案. 该团队将签名验证中的配对计算转换为具有固定参数的配对的乘积, 从而避免了签名验证中 $\mathbb{G}_2$ 域的标量乘法操作. 除此之外, 该团队还使用预计算技术来改进椭圆曲线标量乘法、$\mathrm{F}_{\mathrm{p}}^{12}$ 中的幂运算和配对计算, 整体提高了基于身份的签名方案的速度.

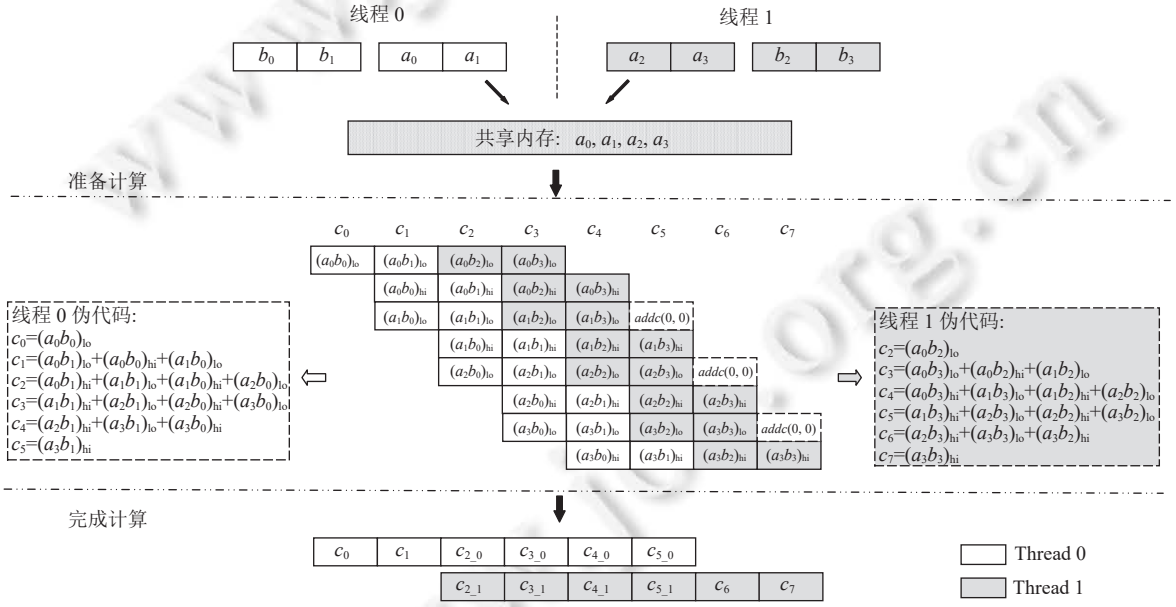

图 4 多线程实现大整数乘法案例

如表 2 所示, 目前针对公钥密码优化的加速方案主要围绕 RSA 与椭圆曲线密码, 基于大整数分解难题的 RSA 密码比特位较长, 多线程并行的性能相比单线程实现具有显著的性能优势, 当前最优的实现相比 CPU 性能提升超过 50 倍; 椭圆曲线密码算法受到 ECC 参数的影响, 存在种类较多, 不同类型的椭圆曲线性能同样存在性能差距, 对于 Curve25519 曲线而言, 目前最优的 GPU 平台实现超过 CPU OpenSSL 算法库的 600 倍, 具有显著的性能优势.

表 2 基于 GPU 的传统公钥密码加速性能对比

实现工作	密码类型	实现平台	吞吐性能 (kops/s)	计算延时 (ms)
OpenSSL[59]	RSA-2048	Intel Xeon E5-2699 v3	1.58	0.6
Emmart等人[76]	RSA-2048	GTX 780Ti	62.36	60.07
Yang等人[77]	RSA-2048	GT 750m	5.24	195.27
Dong等人[70]	RSA-2048 RSA-4096	GTX Titan Black	52.74 6.43	18.20 74.5
OpenSSL[59]	Curve25519	Intel Xeon E5-2699 v3	21.57	—
FourQlib[78]	(ECC) FourQ	Intel Xeon E5-2699 v3	58.97	—
Cui等人[79]	(ECC) 224-bit Edwards	GTX285	115	19.20
Mahe等人[80]	(ECC) Curve25519	GTX Titan	524	—
Zheng等人[81]	(ECC) SM2	GTX 780Ti	391	—
Gao等人[82]	(ECC) Curve448	Tesla P100	1 183	—
Dong等人[71]	(ECC) Curve25519	GTX Titan	1 394	—
Gao等人[72]	(ECC) Edwards25519 (ECC) Curve25519	Titan V	7 216 13 558	1.51 2.84

3.3 后量子密码并行加速

2015 年, 美国伍斯特理工学院团队[83]介绍了如何在 GPU 上加速 BCD 算法, 以及如何在 GPU 上加速信息集解码算法. 作者使用了 NVIDIA K20 GPU, 通过并行化信息集的生成和输出, 以及使用 GF(2) 高斯消元算法来加速解码过程. 作者的实验结果表明, 使用 GPU 加速后, ball collision decoding 算法的性能比 CPU 快了一个数量级. 2016 年, 埃及姆努菲亚大学团队[84]对 McEliece 密码系统在 NVIDIA GTX780 GPU 上使用 OpenCL 框架的不同实现. 其实现结果表明, 当应用矢量数据类型的本地内存加密 331 条消息时, GPU 比 CPU 快 216 倍.

2019 年, 比利时恩智浦公司团队[85]提出了提出一种新的算法, 用于在嵌入式设备上进行基于同源性的加密. 该算法使用了一种新的技术, 称为“pi 同源性”, 它可以在嵌入式设备上实现更快的模算术运算. 文章表明, 虽然服务器可以利用更快的模算术, 但必须评估 p 同源性, 这比评估小 p 同源性略微昂贵. 但是, 服务器通常可以承受一些性能价格, 特别是当面临在大规模部署的嵌入式设备上进行公钥加密的场景时. 同年, 韩国仁荷大学团队[86]提出了一种基于后量子密码学的面部安全系统, 该系统使用了名为 NewHope 密码学的后量子加密算法, 以保护从视频中提取的面部图像. 该方法将输入数据进行排列以进行加密和解密过程, 从而显著减少了加密和解密时间. 该面部安全系统在 NVIDIA GTX 2080Ti GPU 上成功地使用了数据并行计算模型进行加速. 平均面框 (190×190 像素) 仅需要 2.2 ms 和 2.7 ms 的总加密和解密时间, 具有可比性的保密性证明. 结果表明, 该系统提供了与先前系统相当的保密性.

2021 年, 韩国国民大学团队[87]在 GPU 上加速超奇异同源基加密机制 (SIKE), 提出了一种在 GPU 上实现 SIKE 机制的有效方法, 针对提供安全级别 2 (至少与 SHA256 一样难以破解) 的 SIKEp503 安全参数的 GPU 实现. 文章优化了底层的字段算术, 并充分利用了 GPU 体系结构的属性, 包括内存层次结构和 CUDA 流. 所提出的基于 RTX 2080T 的 GPU 软件每秒提供了约 36 376.61 个 KeyGens、25 603.72 个 Encaps 和 22 211.61 个 Decaps. 这些数字分别比 Intel i9-10900K CPU 上 SIKE CPU 软件快 140.64 倍 157.66 倍和 146.81 倍. 2021 年, 美国北亚利桑那大学团队在文献 [88] 中指出 RBC 协议通过使服务器独立更正其自己的种子来解决此问题. 但是, 随着 PUF 错误率的增加, 种子校正的计算要求呈指数级增长. 因此, 采用 GPU 等架构并行执行此种子校正. 文章提出了已知第 1 个在 GPU 上实现 CRYSTALS-Dilithium 并使用此实现开发文献中首次报告的后量子 RBC 协议. 并且将 GPU 加速的 CRYSTALS-DilithiumRBC 算法与使用多核 CPU 并行化的基线实现进行了比较. 其使用 GPU 的方法在安全级别 2、3 和 5 上分别实现了 69.03 倍、82.52 倍和 90.70 倍的加速. 该团队还将 PUF 种子分成子种子, 这允许在固定时间阈值下给定更高的 PUF 误差率. 同年, 该团队在文献 [89] 中探讨了使用并行计算技术在合理的时间限制内

快速找到客户的公钥. 研究了在共享内存环境中使用多核 CPU 和众核图形处理单元. 该团队专注于性能工程 RBC 搜索中使用的几个 CUDA 内核, 这些内核系统地探索了这个空间. 其 RBC 搜索算法具有高度可扩展性: 多核 CPU 算法在 61 个 CPU 内核上实现了 82.64 倍的加速, 其多 GPU 算法在 2 个 GPU 上实现了近乎完美的 93.3 倍加速. 其 GPU 算法可以在 6 s 内对用户进行身份验证, 这远低于身份验证时间阈值.

2021 年, 新加坡国立大学研究团队[90]详细设计 NTT 和 INTT 模块, 提出了用于细粒度实现的 SIMD 并行化范式, 将其应用于密钥封装算法 NewHope 中. 此外, CuNH 还支持批处理操作, 可以同时处理多个密钥交换请求, 通过将 NewHope 移植到 GPU 上并使用 CuNH 实现, 可以在加速 post-quantum 密钥交换的同时, 还可以降低系统成本和功耗. 同年, 美国加利福尼亚大学团队研究人员[91]探讨了基于 NTT 和卷积的不同多项式乘法器在 GPU 上的效率. 在 NVIDIA Jetson TX2 上设计的基于 NTT 的 512 度和 1 024 度的多项式乘法器分别比在 FPGA 上设计的基于 NTT 乘法器快 1.2 倍和 2 倍, 这种探索和指导方针可以帮助设计者选择合适的实现方式来实现量子电阻信号处理. 2021 年, 新加坡南洋理工大学团队[92]研究了 3 类不同的基于格的后量子算法: 带误差学习 (LWE)、环 LWE 和模块 LWE. 使用两种不同的实现方法展示了算成本高的算法在不同场景中的实际适用性, 如 NTT、矩阵乘法和 Keccak. 对于 NewHope 和 Kyber, 该实现能够分别执行大约 504k 和 473k 的密钥交换, 与参考 C 实现相比, 速度提高了近 53.1 和 51.05 倍. 如图 5 所示, 2022 年, 中国科学院信息工程研究所研究团队在文献 [93] 中, 利用 NVIDIA AI 加速器 tensor core 来加速多项式乘法. 他们采取措施适应张量核心的矩阵乘加模式, 并在精度和性能之间进行权衡, 将其用作高性能 NTT 盒, 通过 CUDA C++ WMMA API 执行 NTT/INTT. 同时, 文章以 CRYSTALS-Kyber 作为 RTX 3080 与 Ampere tensor core 的案例研究. 实证结果表明, 多项式向量 (n=256, k=4) 使用的 NTT 盒子, 获得的加速比约为同一 GPU 平台上最先进实现的 6.47 倍. 2022 年, 韩国高阳大学团队[94]提出了几种并行算法, 以允许 tensor core 处理灵活的矩阵大小和短暂的密钥对. 作者将基于 tensor core 的多项式卷积技术应用于 NTRU, 作者们将其扩展到其他具有小模数的基于格的加密系统: LAC 和 FrodoKEM 中的两个变体参数集. 考虑到物联网网关设备和云服务器需要处理来自传感器节点的大量连接, GPU 上提出的高吞吐量实现在保护物联网通信方面非常有用. 2022 年, 韩国嘉泉大学研究团队[95]通过点积指令加速矩阵乘法和并设计专门的数据结构实现多项式卷积运算, 在 GPU 上实现两种基于格的后量子密码算法并评估其性能来展示其方法的有效性, 相对于传统的基于 CPU 的实现, 实现了显著的性能提升. 其中, FrodoKEM 的工作实现比当时最先进的 V100 高 4.37 倍的吞吐量. 同年, 该研究团队[96]提出具有组合层的数论变换 (NTT) 的全并行实现, 该实现比当时 GPU 上最先进的结果快 2.65 倍. 其他提出的技术包括并行拒绝采样、优化内存访问的中心二项式分布和并行细粒度 AES-256. 这些技术在 RTX 2060 GPU 上实现了每秒 162 760 次封装和每秒 107 631 次解封装的高吞吐量性能, 以满足物联网应用程序的需求. 2023 年, 西安交通大学团队[97]提供了 3 种并行的 XMSS 方案: 算法并行、多密钥对数据并行和单密钥对数据并行. 设计了自定义并行策略, 这些策略对 NIST 提供的所有参数使用 10 000 多个内核. 作者还分析了大多数先前串行优化的可用性, 并探索了许多技术来充分利用 GPU 性能.

如表 3 所示, 上文已经提到目前已有一些基于 GPU 的后量子密码并行加速方案被提出, 主要涉及几类后量子密码算法, 比如, 基于格的公钥加密和密钥封装机制 (KEM), 如 NewHope、Kyber、FrodoKEM; 基于格的数字签名, 如 Dilithium、XMSS; 基于同源的密钥封装机制, 如 SIKE; 基于编码的公钥加密, 如 McEliece. 这些方案主要采用几种技术来优化 GPU 上的后量子密码运算, 比如, 利用 SIMD(单指令多数据) 或 SIMT (单指令多线程) 的并行化范式来设计细粒度的 GPU 实现, 充分利用 GPU 上的多核心和多线程; 采用任务级别的批处理来提高硬件资源的利用率和吞吐量; 采用动态任务调度机制来平衡不同任务之间的执行时间和硬件占用率; 采用异步计算和多流技术来隐藏数据传输延迟和最大化 CPU 和 GPU 之间的计算能力; 采用内存优化技术来减少内存使用量和 IO 延迟, 解决内存冲突和流水线停滞问题; 采用点积指令或其他特殊指令来加速数值运算, 如 NTT (数论变换)、逆 NTT、模乘等. 根据以上参考文献中提供的实验数据, 基于 GPU 的后量子密码并行加速方案可以在不同安全级别下实现毫秒级或微秒级的执行时间, 并且可以达到 CPU 性能的几十倍甚至几百倍. 例如, 在安全级别为 3 时, 基于 GPU 的 Dilithium 签名方案可以在 32 ms 内同时完成 10 000 个签名任务, 而基于 CPU 的 Dilithium 签名方案需要约 1.5 s 才能完成一个签名任务, 即基于 GPU 的 Dilithium 签名方案比基于 CPU 的方案快了约 470 倍.

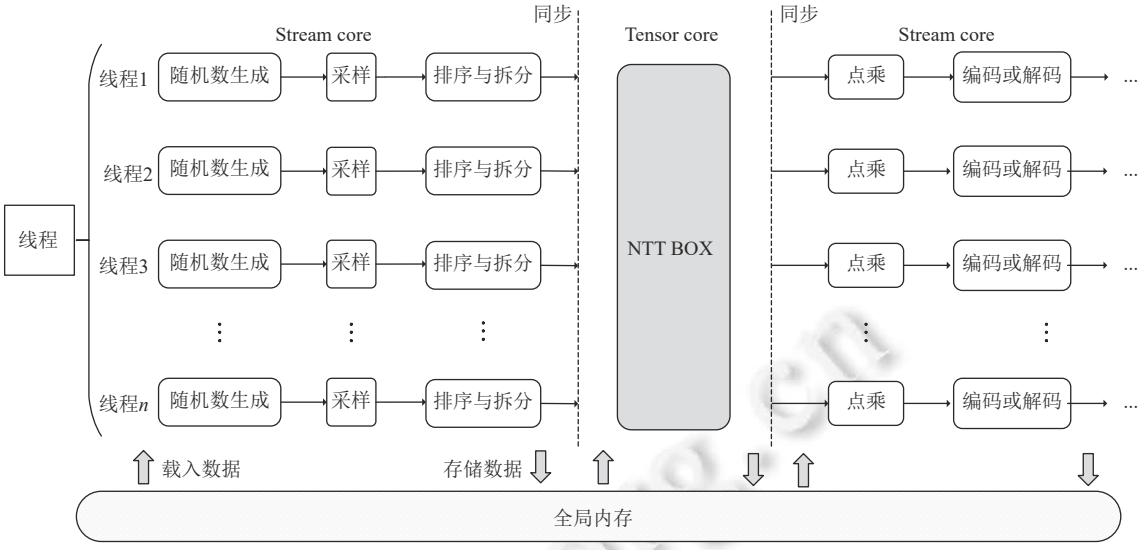

图 5 基于 GPU tensor core 的格密码计算架构

表 3 基于 GPU 的后量子密码并行加速性能对比

实现工作	密码类型	实现平台	吞吐性能 (kops/s)	计算延时 (ms)
Fogel[83]	BCD	NVIDIA K20	—	264 354
Elsobky等人[84]	McEliece	NVIDIA GTX780	20	0.053
Seo[87]	SIKE	RTX 2080Ti	35	508.95
Gao等人[90]	NewHope	GeForce GTX 1650	1 000	1 000
Nejatollahi等人[91]	(PQC) NTT	NVIDIA Jetson TX2	—	0.49
Gupta等人[92]	NewHope	NVIDIA GV100	699	80
	Kyber	NVIDIA GV100	606	—
Wan等人[93]	CRYSTALS-Kyber-1024	RTX 3080	819.7	1.99×10^{-3}
Lee等人[94]	FrodoKEM	RTX 3080	6 208	—
		V100	4 885	—
		T4	2 638	—
	Saber KEM	RTX 3080	124 418	—
		V100	120 463	—
		T4	31 658	—
Lee等人[95]	XMSS-SHA2_20/2_256	GeForce RTX 3090	419.887 (Verify)	39.02 (Verify)

3.4 同态密码并行加速

2012 年, 美国伍斯特理工学院研究团队[98]将 Gentry 和 Halevi 全同态加密方案[36]在 GPU 上进行了性能优化, 利用 GPU 的大规模并行性对基于 Schönhage-Strassen 的大整数乘法进行加速, 并使用了 Barrett 模块化约减算法, 相较于 CPU 实现, 加密、解密和重加密的性能分别为提升了 7.68 倍、7.4 倍和 6.59 倍. 2014 年, 美国伍斯特理工学院戴伟团队[99]提出了一个支持全同态加密的针对 NVIDIA GPU 的大型多项式算术库, 利用余数定理将大系数多项式转化为多个小系数多项式, 并使用离散傅里叶变换完成模乘, 实验表明 Prince 和 AES 两种分组密码的同态方案相较于 CPU 实现加速了 2.57 倍和 7.6 倍. 2015 年, 该团队[100]通过优化 CUDA 模块化乘法、规约和模切换的代码来支持大系数多项式在 GPU 上的运算, 并为多项式 CRT 域表示和部分同态方案的模切换选择了相同的素数, 通过组合两个算数域减少域转换次数提升执行速度, 与 CPU 软件实现相比, 索引比较加速了 14–34 倍, 数据聚

合加速了 4–18 倍.

2016 年, 美国伍斯特理工学院戴伟团队[101]提出了一个 CUDA GPU 库用于加速定义在多项式环上的同态加密方案, 扩展了利用 NTT 和 CRT 方法构建的用于处理多项式操作数的算术函数, 所提出的 CUDA 库在单 GPU 和三 GPU 上相较于其他 GPU 实现提高了 25 倍和 51 倍. 2018 年, 新加坡国立大学研究团队[102]提出了一个基于 CUDA 的 FV 近似同态加密方案实现, 利用 CRT、RNS、DGT 加速 GPU 上的 FV 运算, 与 SEAL 和 NFLlib-FV 相比性能提升 5–22 倍. 2020 年, 韩国首尔大学研究团队[103]分析了 NTT 和 DFT 的算法特征并提出了利用旋转因子优化 NTT 的动态根生成方案, 在 GPU 上实现了 4.2 倍的性能提升. 2020 年, 新加坡信息研究所研究团队[104]在多 GPU 上实现了 FV 全同态加密方案的 HPS 变体, 利用分区方法在多 GPU 间均匀分配 FV 中的工作负载, 对比 CPU 实现提供了 1–3 个数量级的加速; 2021 年, 马来西亚拉曼大学研究团队[105]改进了 GPU 中的 NTT 实现, 将旋转因子存储于 GPU 的寄存器中, 利用 warp shuffle 指令实现跨线程访问, 比美国伍斯特理工学院 Dai 等人的最新实现[101]更快.

2021 年, 韩国首尔大学研究团队[106]首次完成了自举 CKKS 的 GPU 实现, 根据以内存为中心的优化思想, 利用内核融合等关键技术提升完全同态加密在 GPU 上的性能, 与最新 GPU 实现相比单次全同态乘法运算提升了 7.02 倍, 相较单线程 CPU 提升了 257 倍. 2022 年, 复旦大学研究团队[107]提出了在异构物联网系统中实现 CUDA 加速 RNS 同态乘法的方案 CARM, 这是第 1 个涵盖 BGV、BFV、CKKS 的 GPU 优化实现, 对比 CPU 实现提供了高达 378.4、234.5、287.2 倍的加速. 同年, 加利福尼亚大学[108]首次利用 Intel GPU 优化同态密码算法, 为 Microsoft SEAL API 提供第 1 个优化的 Intel GPU 库, 从指令级、算法级和应用级进行优化, 其关键算法 NTT (number theoretic transform) 加速了高达 9.93 倍. 最终性能超过 2.3 倍. 中国科学院信息工程研究所研究团队[109]于 2023 年提出了一种基于 GPGPU 的全同态加密加速方案 TensorFHE, 利用 TCU 来促进 NTT 运算, 为了充分利用 GPGPU 的并行能力引入了操作级批处理, 实验表明该实现方案与最新 ASIC 加速器具有相当的性能. 如图 6 所示, 2023 年, 中国科学院信息工程研究所研究团队[110]提出了一种基于 GPU 的高效 FHE 加速设计, 对于单 GPU 加速, 将 FHE 方案中的 5 个常见相位映射到 GPU 并行架构, 并提出了一种线程间的本地同步来利用线程级并行性, 对于多 GPU 加速提出了一种可扩展的并行化设计, 利用不同表示下的细粒度数据分区实现数据级并行性, 实验结果相较于 cuHE 实现了 170.5 倍的性能增长.

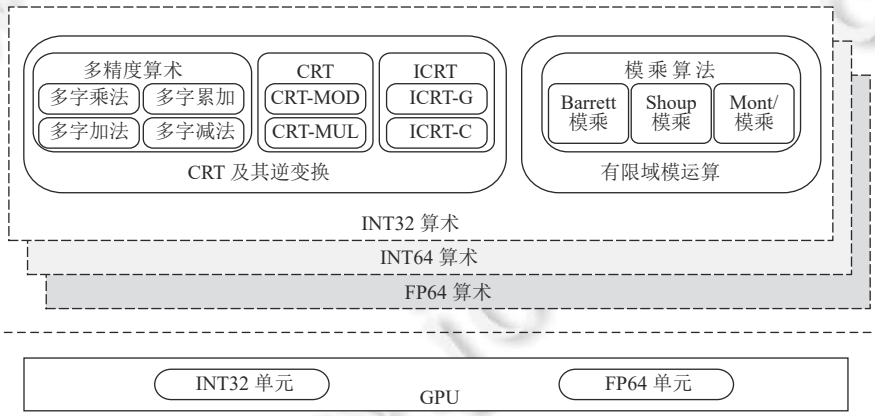

图 6 基于 GPU 的同态密码底层算子架构

如表 4 所示, 自 2021 年 Jung 等人[106]提出了以内存为中心的优化思想后, 利用 GPU 对同态加密进行加速这一领域出现了几大清晰的研究方向, 重点利用 GPU 特有的 SIMD 架构实现针对数据的并行化改造, 研究主要集中于同态加密中密集使用的 NTT、CRT 等算术运算的加速. 一方面, 部分研究学者充分利用 GPU 片上内存访存速率高的特点进行内存优化; 另一方面, 一些学者从指令集角度出发, 完成同态加密在 GPU 上的加速. 当前 GPU 上的单个同态加密乘法操作相比于 CPU, 计算时延实现了超过 1 000 倍的加速.

表 4 基于 GPU 的同态密码并行加速性能对比

实现工作	密码类型	实现平台	计算延时 (ms)
Jung等人[111]	CKKS	AVX-512 Titan RTX	74.8 38.1
Yang等人[112]	BFV	NVIDIA Tesla P100 GPU	0.294 (KeyGen_sk) 0.436 (KeyGen_pk) 1.356 (KeyGen_rlk) 1.377 (KeyGen_rtk) 1.400 (Enc) 0.430 (Dec) 0.088 (HAdd) 5.093 (HMult)
	CKKS	NVIDIA Tesla P100 GPU	0.316 (KeyGen_sk) 0.442 (KeyGen_pk) 1.355 (KeyGen_rlk) 1.388 (KeyGen_rtk) 1.469 (Enc) 0.088 (Dec) 0.089 (HAdd) 1.445 (HMult)
Al Badawi等人[102]	BFV	GTX 280	174.508 (KeyGen) 3.296 (Enc) 0.252 (Dec) 0.053 (HAdd) 11.747 (HMult)
Jung等人[106]	CKKS	NVIDIA Tesla V100	0.162 (HAdd) 2.960 (HMult) 0.135 (CMult) 0.490 (Rescale) 2.550 (HRot)
Türkoglu等人[113]	438 bits BFV	RTX 3060Ti	4 890
Wang等人[110]	881 bits BGV	NVIDIA Tesla V100	0.06 (HAdd) 2.42 (HMult) 2.31 (HRot)
	438 bits BGV	NVIDIA Tesla V100	0.02 (HAdd) 0.69 (HMult) 0.61 (HRot)

4 研究展望

目前已经有很多研究工作专注于利用硬件加速器优化密码算法, 特别是在 GPU 上加速各类密码算法的研究工作已经取得较大进展. 未来可进一步探索研究国产密码的硬件加速方案、在国产芯片上加速密码算法、对计算开销较大的新型密码的加速、适配人工智能加速器的密码加速以及提供抗密码攻击的密码安全防护硬件优化方案等.

4.1 国产密码

2019 年, 我国对密码实行分类管理, 将密码分为核心密码、普通密码、商业密码这 3 大类, 核心密码、普通密码用于保护国家秘密信息, 由密码管理部门统一管理, 商用密码可被依法用于保护网络与信息安全[114]. 近年来, 我国在密码学领域的研究和发展取得了显著进展, 其中商业密码算法的研究和应用是其中的重要领域.

国产商业密码算法主要包括 SM1、SM2[27]、SM3[115]、SM4[11]、SM7、SM9[116]、ZUC[13]等. 其中, SM1 是政府保密通信使用的加密算法, SM1 算法未公开, 普遍用于电子政务、警务机关、银行等领域. SM2 算法是我国研

发的具有自主知识产权的 ECC 算法, 其在安全性和实现效率方面表现优异, 甚至略优于国际上同类的 ECC 算法. 相比 RSA 算法, SM2 算法在公钥密码算法安全性和实现效率方面也具有更高的优势, 在未来的推广和应用方面具有广阔的前景. SM3 密码杂凑算法是国家商用密码杂凑算法标准, 消息分组长度为 512 比特, 摘要长度为 256 比特, 采用 Merkle Damgard 结构, 具备高强度和高安全性的哈希功能. 由于传统 MD 结构设计的杂凑函数被证明不安全, 因此, 对杂凑函数的设计与分析又成为密码学界的一大研究热点. SM4 算法是一种分组算法, 分组长度和密钥长度均为 128 比特, 采用 32 轮非线性迭代结构, 适用于无线局域网产品. SM4 算法是我国第 1 次由专业密码机构公布并设计的商用密码算法, 且迄今为止还没有发现任何攻击方法能够对其安全性造成威胁. SM7 算法是一种使用 128 比特分组长度和密钥长度的分组密码算法, 主要适用于非接触式 IC 卡, 例如门禁卡、一卡通等应用. 1984 年, Shamir 提出了标识密码[117]的理念, 用于简化公开密钥系统中的密钥和证书管理. 该理念使用用户标识(如邮件地址、手机号码、QQ 号码等) 作为公钥, 省略了数字证书和公钥交换过程, 从而使安全系统易于部署和管理. 2008 年, 我国发布商用密码 SM9 算法, 其不需要申请数字证书, 适用于互联网各种新兴应用, 可以实现数据加密、身份认证、通话加密、通道加密等安全应用, 具有使用方便、易于部署的特点. 2021 年, SM9 密钥交换算法正式成为 ISO/IEC 国际标准[118]. ZUC 密码算法是一种序列密码算法, 其分组长度和密钥长度均为 128 比特, 采用线性反馈移位寄存器 (linear feedback shift register, LFSR) 作为伪随机数生成器的核心组件, 被广泛应用于各种通信和存储设备中, 如移动通信 4G 网络和物联网领域, 并且已被多个国际组织认可. ZUC 算法由 3 个部分组成, 分别是祖冲之算法 (ZUC)、加密算法 (128-EEA3) 和完整性算法 (128-EIA3). 目前, 已经有专门针对 128-EEA3 和 128-EIA3 的硬件实现与优化[119].

在密码法中明确规定, 国家鼓励和支持密码科学技术的研究和应用. 由于国产密码研究已经取得了显著进展, 并且数字化时代对高性能密码计算技术的需求越来越强烈. 因此, 对国产密码的硬件加速, 特别是基于具备通用并行计算能力的 GPU 加速, 将成为未来密码硬件优化方案的研究重点.

4.2 国产芯片

近几年, 在国家的相关政策扶持下, 国产数字芯片厂商呈爆发式增长, 我国国产芯片的发展已经取得了显著进展[120]. 在 CPU 领域, 中国已经取得了一定的自主可控成果. 例如, 基于 x86 架构的兆芯 CPU、基于 MIPS 架构的龙芯 CPU 以及基于 ARM 架构的华为海思、全志 T3 和瑞芯微 RK3328 的 CPU 等国产 CPU 芯片[121]已经实现市场化, 可应用于手机、平板电脑、个人电脑、服务器以及面向边缘计算的嵌入式设备等多种领域. 在 GPU 领域, 中国的显卡品牌正逐渐崛起, 包括图形处理 GPU、通用计算 GPU、AI 加速 GPU 等. 景嘉微、芯动科技、天数智芯等公司已经推出了自己的显卡产品, 并且有部分 GPU 采用国产架构. 2020 年底, 天数智芯发布了自研的通用并行 GPU 芯片, 具有强大的可编程性和通用性, 并支持针对云端 AI 训练和推理以及 HPC 通用计算的软硬件架构, 支持多种精度的数据类型, 包括浮点数和定点数. 2021 年底, 芯动科技发布了名为“风华 1 号”的 GPU 芯片, 支持 X86、ARM 等不同指令集处理器, 单精度浮点性能可达 5 TFLOPS, 显存带宽最大可达 304 GB/S, 且功耗超低, 在桌面应用中小于 20 W. 在 5G 通信、工业控制以及专业细分领域, FPGA 具有无法替代的优势, 国内科技企业也在积极投入到其研发和生产领域, 例如紫光国微、安路科技、复旦微电等企业. 其中, 安路科技的第 1 代 FPGA 架构支持高达 600K 逻辑阵列容量.

总体来说, 我国国产芯片发展已经取得了不少成就. 因此, 基于国产芯片的密码算法加速方案也是未来的一个重要研究方向.

4.3 安全多方计算

随着大数据与人工智能等领域的快速发展, 数据采集、分析等应用也在不断增加, 随之而来的是数据的所有权问题和隐私性问题. 数据计算过程中缺乏有效的数据保护技术限制了企业之间数据的互通, 导致数据价值无法得到充分发挥, 从而形成数据孤岛. 为了解决这些问题, 隐私计算 (privacy-preserving computation) 技术应运而生, 又称为隐私增强 (privacy enhance compute) 技术, 是指在不泄露数据拥有方原始隐私数据的前提下实现数据的分析计算的一系列技术[122], 使得数据在可用的同时不会泄露, 从而共享数据价值.

安全多方计算 (secure multi-party computation, MPC) 是密码学的一个重要研究领域, 与联邦学习 (federated learning, FL)、可信执行环境 (trusted execution environment, TEE) 构成当前隐私计算的 3 大主流路径[122]. MPC 问题起源于图灵奖获得者姚期智教授于 1982 年提出的百万富翁问题[123]——两位百万富翁想知道谁更富有, 但又不想向对方泄露自己的实际财产信息. MPC 协议是用于解决多方协同计算隐私保护问题的密码协议, 互不信任的各参与方 P_i 在没有可信第三方的前提下, 协同计算某个提前约定好的函数, 每个参与方除了自己的隐私输入 X_i 和相应的函数输出 Y_i 之外, 得不到其他额外的信息. MPC 基础协议包括混淆电路 (garbled circuit, GC)、秘密分享 (secret sharing, SS)、不经意传输 (oblivious transfer, OT)、同态加密等基本原语, 这些基本原语可单独实现某个专用功能, 也可组合应用实现安全的 MPC 协议. 如既可以通过 OT 实现 MPC, 也可以将 OT 作为 MPC 的基础原语.

一个多项式时间复杂算法的实现可以分解为基本算子的组合运算. 任意多项式时间算法都等价于多个电路门的组合运算, 其中包括加法电路、乘法电路、比较电路等; 也等价于多个逻辑门的组合运算, 包括与门、或门、非门、与或非门等. 根据适用性标准, MPC 可分为通用 MPC 和专用 MPC. 通用 MPC 支持大多数计算任务, 通过实现常见基本算子操作, 如加法、乘法、比较、矩阵乘法等基本运算, 从而实现复杂的计算任务. 专用 MPC 侧重于针对某些特定领域和应用场景实现专用的功能函数, 如隐私集合交集 (private set intersection, PSI)、隐私信息检索 (private information retrieval, PIR) 等. 专用功能函数可直接用于特定应用场景, 也可作为其他应用领域的基础构建模块. 在某些情况下, 专用 MPC 协议甚至比最优的通用 MPC 协议更高效. 在机器学习即服务 (machine learning as a service, MLaaS) 模式中, 数据持有者利用云服务器提供商提供的训练好的机器学习模型进行机器学习推理, 从而获得相应的预测服务、决策服务和推荐服务[124]等. 基于安全多方计算的隐私保护机器学习 (privacy-preserving machine learning, PPML) 是当前 MPC 领域的研究热点, 可以保护服务商的模型权重参数隐私以及数据拥有者的隐私数据信息, 在不泄露双方敏感数据的情况下, 协同实现安全的机器学习推理[125].

自 1986 年姚期智教授团队提出第 1 个安全多方计算协议[126]以来, 安全多方计算领域涌现了许多协议、框架和改进方案[127–133]. 安全多方计算包含复杂的密码学操作, 其通信和计算开销问题一直是实现落地应用的难点. 2011 年, 美国得克萨斯农工大学团队提出的 FastPlay[134]方案是基于 FairPlay[133]框架的 GPU 实现方案, 相较于 CPU 方案, 实现了 35–40 倍的加速. 目前已经有许多研究工作[135–141]利用 GPU 加速优势优化安全多方计算协议.

因此, 基于 GPU 加速安全多方计算是一项切实有效的安全多方计算优化方案, 也是安全多方计算领域未来的一个重要研究方向.

4.4 人工智能加速器

近年来, 人工智能 (artificial intelligence, AI) 应用发展迅猛, 通用处理器已难以满足高算力的需求, 因此出现了专门为人工智能应用 (如卷积神经网络、计算机视觉和机器学习) 设计的加速器/处理器, 被称为人工智能加速器或 AI 芯片. 人工智能加速器通常具备超强算力, 且能效比很高, 能够在小功耗下获得可观的性能. 目前市面上已有众多人工智能加速器产品, 包括 NVIDIA 的 tensor core、Apple 的人工智能加速器、Google 的 TPU、Intel 的 ANN 等. 众多厂商产品的关注点在于低精度运算、定制的数据流架构及内存计算能力, 并且通常采用多核设计, 主要面向数据密集型任务, 适用于资源受限的物联网设备和机器人上的算法等应用场景.

人工智能应用一般运算在浮点数上. 以 NVIDIA GPU 为例, 其单精度浮点数计算性能从 2010 年的 1 300 GFLOPS 发展到 2013 年的 5 000 GFLOPS, 而整数乘法的处理能力仅增长了 1/4[142,143]. 2017 年, NVIDIA 发布了第 1 代配备人工智能加速器的 Volta[144]架构, 首次引入用于加速人工智能应用的专用处理子单元, 即张量核心 (tensor core). 以 NVIDIA 早期的 Volta 架构为例, 人工智能加速器 tensor core 峰值性能可达到 125 TFLOPS, 而 V100 设备上的 CUDA 核心 (传统 GPU) 峰值性能只有 15 TFLOPS. 目前 NVIDIA GPU 已被广泛应用于深度学习领域中. 然而, 传统公钥密码 (如 RSA 和 ECC) 核心操作基于大整数乘法运算, 新型密码如后量子密码、安全多方计算等均运算在整数上, 人工智能加速器对特殊操作的定制化设计使其不具备通用计算性.

因此, 在目前更偏向于提升浮点数性能的人工智能加速器发展背景下, 研究如何充分利用其并行计算能力, 设计适配于人工智能加速器的密码加速方案也成为本领域的研究重点.

4.5 安全防护

由于强大的并行处理能力, GPU 在现代计算机系统中扮演重要角色, 在图像渲染、人工智能、高性能计算等领域发挥着越来越重要的作用. 然而, 针对 GPU 的安全攻击无法避免, 文献 [145] 研究了 GPU 可能遭受的密码攻击. 基于 GPU 的高性能密码计算技术, 可能存在侧信道攻击[145]、内存攻击[146]和心脏滴血[147]等漏洞. 攻击者通过分析 GPU 上的电磁泄漏或功耗变化, 可以推断出 GPU 正在执行的指令或数据, 从而获取机密信息. 进一步, 目前 GPU 正朝着支持并发内核执行的方向发展, 多个内核可以在同一个 GPU 上协同执行, 甚至可以在同一个流多处理器 (SM) 核心上协同执行. 并发的内核执行提高了硬件资源利用率, 但它也存在隐蔽通道和侧信道攻击漏洞的潜在风险, 争用共享资源导致跨内核信息泄漏. 攻击者还可以利用 GPU 驱动程序中的漏洞, 执行恶意代码或者获取敏感信息.

未来可以尝试基于 CPU 平台的安全计算防护技术[148–151], 展开针对 GPU 的安全防护技术研究, 利用 NVIDIA GPU 的动态并行技术[108]等. 因此, 研究针对硬件平台的安全计算防护技术也将是未来研究的重点, 以实现安全高效的并行密码计算技术.

5 总 结

密码学在保护信息安全方面发挥着越来越重要的作用, 高性能密码学的重要性也逐渐凸显. 现代密码学方向的不断拓展和 GPU 并行加速技术的不断发展, 为密码计算技术的加速提供了更多的发展空间. 本文主要介绍了密码学领域中基于 GPU 的高性能密码计算技术加速的发展现状. 基于 GPU 的密码加速方法不仅可以广泛应用于现代密码学领域, 如对称密码、公钥密码和杂凑密码等, 而且在后量子密码、同态密码等新兴密码学领域也有广泛的应用. 此外, 本文还介绍了该领域的研究展望, 包括国产密码、国产芯片、新型密码、人工智能加速器和安全防护等, 这些方向将是未来密码学硬件优化领域的研究热点. 期待未来更多的基于 GPU 的密码学加速方法的研究和应用, 进一步推动密码学技术的发展.

References:

[1] Rescorla E. HTTP over TLS. 2000. https://www.rfc-editor.org/info/rfc2818 [doi: 10.17487/RFC2818]

[2] Freier A, Karlton P, Kocher P. The secure sockets layer (SSL) protocol version 3.0. 2011. https://www.rfc-editor.org/info/rfc6101 [doi: 10.17487/RFC6101]

[3] Dierks T, Rescorla E. The transport layer security (TLS) protocol version 1.2. 2008. https://www.rfc-editor.org/info/rfc5246

[4] National Cryptography Administration. The executive meeting of the state council deliberated and adopted the regulations on the administration of commercial passwords (draft revision). 2023 (in Chinese). https://www.oscca.gov.cn/sca/xwdt/2023-04/20/content_1061005.shtml

[5] Adams C, Lloyd S. Understanding Public-key Infrastructure: Concepts, Standards, and Deployment Considerations. Indianapolis: Macmillan Technical Publishing, 1999.

[6] NVIDIA. CUDA C++ programming guide 9.0. 2017. https://docs.nvidia.com/cuda/cuda-c-programming-guide/

[7] Shannon CE. Communication theory of secrecy systems. The Bell System Technical Journal, 1949, 28(4): 656–715. [doi: 10.1002/j.1538-7305.1949.tb00928.x]

[8] Diffie W, Hellman ME. New directions in cryptography. In: Slayton R, ed. Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman. New York: Association for Computing Machinery, 2022. 365–390. [doi: 10.1145/3549993.3550007]

[9] Coppersmith D. The data encryption standard (DES) and its strength against attacks. IBM Journal of Research and Development, 1994, 38(3): 243–250. [doi: 10.1147/rd.383.0243]

[10] Nechvatal J, Barker E, Bassham L, Burr W, Dworkin M, Foti J, Roback E. Report on the development of the advanced encryption standard (AES). Journal of Research of the National Institute of Standards and Technology, 2001, 106(3): 511–576. [doi: 10.6028/jres.106.023]

[11] General Administration of Quality Supervision, Inspection and Quarantine of the People's Republic of China, Standardization Administration of the People's Republic of China. GB/T 32907-2016 Information security technology—SM4 block cipher algorithm. Beijing: Standards Press of China, 2017 (in Chinese).

[12] Fluhrer S, Mantin I, Shamir A. Weaknesses in the key scheduling algorithm of RC4. In: Proc. of the 8th Int'l Workshop on Selected Areas in Cryptography. Toronto: Springer, 2001. 1–24. [doi: 10.1007/3-540-45537-X_1]

[13] General Administration of Quality Supervision, Inspection and Quarantine of the People's Republic of China, Standardization Administration of the People's Republic of China. GB/T 33133.1-2016 Information security technology—ZUC stream cipher algorithm—Part 1: Algorithm description. Beijing: Standards Press of China, 2016 (in Chinese).

[14] Merkle RC. Secrecy, Authentication, and Public Key Systems. Stanford: Stanford University, 1979.

[15] Rivest RL. The MD4 message digest algorithm. In: Proc. of the 10th Annual Int'l Cryptology Conf. on Advances in Cryptology. Santa Barbara: Berlin, 1990. 303–311. [doi: 10.1007/3-540-38424-3_22]

[16] Rivest R. The MD5 message-digest algorithm. 1992. https://www.rfc-editor.org/info/rfc1321 [doi: 10.17487/RFC1321]

[17] Krawczyk H, Bellare M, Canetti R. HMAC: Keyed-hashing for message authentication. 1997. https://www.rfc-editor.org/info/rfc2104 [doi: 10.17487/RFC2104]

[18] Dobbertin H, Bosselaers A, Preneel B. RIPEMD-160: A strengthened version of RIPEMD. In: Proc. of the 3rd Int'l Workshop on Fast Software Encryption. Cambridge: Springer, 1996. 71–82. [doi: 10.1007/3-540-60865-6_44]

[19] Barreto P, Rijmen V. The Whirlpool hashing function. 2000. https://www.researchgate.net/publication/228610491_The_Whirlpool_hashing_function

[20] Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 1978, 21(2): 120–126. [doi: 10.1145/359340.359342]

[21] Rabin MO. Digitalized Signatures and Public-key Functions as Intractable as Factorization. Cambridge: Massachusetts Institute of Technology, 1979.

[22] Zhang Y, Lin Y, Hao L. Summarize of elliptic curve cryptosystem research. Computer Engineering, 2004, 30(3): 127–129 (in Chinese with English abstract). [doi: 10.3969/j.issn.1000-3428.2004.03.052]

[23] ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. on Information Theory, 1985, 31(4): 469–472. [doi: 10.1109/TIT.1985.1057074]

[24] Kerry CF, Gallagher PD. FIPS 186-4 Digital signature standard (DSS). 2013. https://www.docin.com/p-928808588.html

[25] Lim CH, Lee PJ. The Korean certificate-based digital signature algorithm. Computers & Electrical Engineering, 1999, 25(4): 249–265. [doi: 10.1016/S0045-7906(99)00011-7]

[26] Bernstein DJ. Curve25519: New Diffie-Hellman speed records. In: Proc. of the 9th Int'l Workshop on Public Key Cryptography. New York: Springer, 2006. 207–228. [doi: 10.1007/11745853_14]

[27] National Cryptography Administration. Announcement of the National Cryptography Administration on the release of SM2 elliptic curve public key cryptography algorithm. 2010 (in Chinese). https://oscca.gov.cn/sca/xxgk/2010-12/17/content_1002386.shtml

[28] Shor PW. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Review, 1999, 41(2): 303–332. [doi: 10.1137/S0036144598347011]

[29] Grover LK. Quantum computers can search rapidly by using almost any transformation. Physical Review Letters, 1998, 80(19): 4329–4332. [doi: 10.1103/PhysRevLett.80.4329]

[30] Schwabe P, Avanzi R. Pqcrystals. 2017. https://pq-crystals.org/

[31] Lyubashevsky V, Ducas L. Pqcrystals. 2017. https://pq-crystals.org/

[32] Prest T, Fouque PA. Falcon-sign. 2017. https://falcon-sign.info/

[33] Hülsing A, Bernstein DJ. Sphincs. 2015. https://sphincs.org/

[34] Rivest RL, Adleman L, Dertouzos ML. On data banks and privacy homomorphisms. Foundations of Secure Computation, 1978, 4(11): 169–180.

[35] Gentry C. Fully homomorphic encryption using ideal lattices. In: Proc. of the 41st Annual ACM Symp. on Theory of Computing. Bethesda: Association for Computing Machinery, 2009. 169–178. [doi: 10.1145/1536414.1536440]

[36] Gentry C, Halevi S. Implementing gentry's fully-homomorphic encryption scheme. In: Proc. of the 30th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Tallinn: Springer, 2011. 129–148. [doi: 10.1007/978-3-642-20465-4_9]

[37] Brakerski Z, Gentry C, Vaikuntanathan V. (Leveled) fully homomorphic encryption without bootstrapping. ACM Trans. on Computation Theory, 2014, 6(3): 13. [doi: 10.1145/2633600]

[38] Fan JF, Vercauteren F. Somewhat practical fully homomorphic encryption. 2012. https://eprint.iacr.org/2012/144

[39] Cheon JH, Kim A, Kim M, Song Y. Homomorphic encryption for arithmetic of approximate numbers. In: Proc. of the 23rd Int'l Conf. on the Theory and Application of Cryptology and Information Security. Hong Kong: Springer, 2017. 409–437. [doi: 10.1007/978-3-319-70694-8_15]

[40] Microsoft. Microsoft SEAL Release 3.7.2. 2021. https://github.com/Microsoft/SEAL/releases/tag/v3.7.2

[41] Homomorphic encryption library (HElib) community. HElib. 2021. https://github.com/homenc/HElib

[42] Sidorov V, Wei EYF, Ng WK. Comprehensive performance analysis of homomorphic cryptosystems for practical data processing. arXiv:2202.02960, 2022.

[43] Owens JD, Houston M, Luebke D, Green S, Stone JE, Phillips JC. GPU computing. Proc. of the IEEE, 2008, 96(5): 879–899. [doi: 10.1109/JPROC.2008.917757]

[44] NVIDIA. NVIDIA developer. 2023. https://developer.nvidia.com/

[45] Intel. Intel® products. 2023. https://www.intel.com/content/www/us/en/products/overview.html

[46] Gilger J, Barnickel J, Meyer U. GPU-acceleration of block ciphers in the OpenSSL cryptographic library. In: Proc. of the 15th Int'l Conf. on Information Security. Passau: Springer, 2012. 338–353. [doi: 10.1007/978-3-642-33383-5_21]

[47] Lee WK, Goi BM, Phan RCW, Poh GS. High speed implementation of symmetric block cipher on GPU. In: Proc. of the 2014 Int'l Symp. on Intelligent Signal Processing and Communication Systems (ISPACS). Kuching: IEEE, 2014. 102–107. [doi: 10.1109/ISPACS.2014.7024434]

[48] Lee WK, Cheong HS, Phan RCW, Goi BM. Fast implementation of block ciphers and PRNG in Maxwell GPU architecture. Cluster Computing, 2016, 19(1): 335–347. [doi: 10.1007/s10586-016-0536-2]

[49] Abdelrahman AA, Fouad MM, Dahshan H, Mousa AM. High performance CUDA AES implementation: A quantitative performance analysis approach. In: Proc. of the 2017 Computing Conf. London: IEEE, 2017. 1077–1085. [doi: 10.1109/SAI.2017.8252225]

[50] Nishikawa N, Amano H, Iwai K. Implementation of bitsliced AES encryption on CUDA-enabled GPU. In: Proc. of the 11th Int'l Conf. on Network and System Security. Helsinki: Springer, 2017. 273–287. [doi: 10.1007/978-3-319-64701-2_20]

[51] Cheng WZ, Zheng FY, Pan WQ, Lin JQ, Li HR, Li BY. High-performance symmetric cryptography server with GPU acceleration. In: Proc. of the 19th Int'l Conf. on Information and Communications Security. Beijing: Springer, 2018. 529–540. [doi: 10.1007/978-3-319-89500-0_46]

[52] Hajihassani O, Monfared SK, Khasteh SH, Gorgin S. Fast AES implementation: A high-throughput bitsliced approach. IEEE Trans. on Parallel and Distributed Systems, 2019, 30(10): 2211–2222. [doi: 10.1109/TPDS.2019.2911278]

[53] Chen ZW, Chen JG, Meng WZ, The JS, Li P, Ren BQ. Analysis of differential distribution of lightweight block cipher based on parallel processing on GPU. Journal of Information Security and Applications, 2020, 55: 102565. [doi: 10.1016/j.jisa.2020.102565]

[54] Fu XL, Di XQ, Lu HM. Parallel and high-speed implementation of SM4 encryption algorithm on OpenCL. In: Proc. of the 2021 Int'l Conf. on Frontiers of Electronics, Information and Computation Technologies. Changsha: ACM, 2021. 94. [doi: 10.1145/3474198.3478218]

[55] Choi H, Seo SC. Fast implementation of SHA-3 in GPU environment. IEEE Access, 2021, 9: 144574–144586. [doi: 10.1109/ACCESS.2021.3122466]

[56] Eum SW, Kim HJ, Kwon HD, Jang KB, Kim HJ, Seo HJ. Implementation of SM4 block cipher on CUDA GPU and its analysis. In: Proc. of the 2022 Int'l Conf. on Platform Technology and Service (PlatCon). Jeju: IEEE, 2022. 71–74. [doi: 10.1109/PlatCon55845.2022.9932098]

[57] Lee WK, Seo HJ, Seo SC, Hwang SO. Efficient implementation of AES-CTR and AES-ECB on GPUs with applications for high-speed FrodoKEM and exhaustive key search. IEEE Trans. on Circuits and Systems II: Express Briefs, 2022, 69(6): 2962–2966. [doi: 10.1109/TCSII.2022.3164089]

[58] Dong JK, Lu S, Zhang PC, Zheng FY, Xiao F. G-SM3: High-performance implementation of GPU-based SM3 hash function. In: Proc. of the 28th IEEE Int'l Conf. on Parallel and Distributed Systems (ICPADS). Nanjing: IEEE, 2023. 201–208. [doi: 10.1109/ICPADS56603.2022.00034]

[59] Foundation OS. OpenSSL cryptography and SSL/TLS toolkit. 2016. http://www.openssl.org/

[60] Sun SZ, Zhang R, Ma H. Hashing multiple messages with SM3 on GPU platforms. Science China Information Sciences, 2021, 64(9): 199103. [doi: 10.1007/s11432-018-9648-x]

[61] Dat TN, Iwai K, Kurokawa T. Implementation of high speed hash function Keccak using CUDA on GTX 1080. In: Proc. of the 5th Int'l Symp. on Computing and Networking (CANDAR). Aomori: IEEE, 2017. 475–481. [doi: 10.1109/CANDAR.2017.47]

[62] Li J, Xie WB, Li LC, Wu XN. Parallel implementation and optimization of SM4 based on CUDA. In: Proc. of the 1st EAI Int'l Conf. on Applied Cryptography in Computer and Communications. Springer, 2021. 93–104. [doi: 10.1007/978-3-030-80851-8_7]

[63] Li CX, Wu HW, Chen SF, Li XC, Guo DH. Efficient implementation for MD5-RC4 encryption using GPU with CUDA. In: Proc. of the 3rd Int'l Conf. on Anti-counterfeiting, Security, and Identification in Communication. Hong Kong: IEEE, 2009. 167–170. [doi: 10.1109/ICASID.2009.5276924]

[64] Szerwinski R, Güneysu T. Exploiting the power of GPUs for asymmetric cryptography. In: Proc. of the 10th Int'l Workshopon Cryptographic Hardware and Embedded Systems. Washington: Springer, 2008. 79–99. [doi: 10.1007/978-3-540-85053-3_6]

[65] Bernstein DJ, Chen TR, Cheng CM, Lange T, Yang BY. ECM on graphics cards. In: Proc. of the 28th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Cologne: Springer, 2009. 483–501. [doi: 10.1007/978-3-642-01001-9_28]

[66] Cohen AE, Parhi KK. GPU accelerated elliptic curve cryptography in GF(2^m). In: Proc. of the 53rd IEEE Int'l Midwest Symp. on Circuits and Systems. Seattle: IEEE, 2010. 57–60. [doi: 10.1109/MWSCAS.2010.5548560]

[67] Hamburg M. Fast and compact elliptic-curve cryptography. 2012. https://eprint.iacr.org/2012/309.pdf

[68] Antão S, Bajard JC, Sousa L. RNS-based elliptic curve point multiplication for massive parallel architectures. The Computer Journal, 2012, 55(5): 629–647. [doi: 10.1093/comjnl/bxr119]

[69] Pan WQ, Zheng FY, Zhao Y, Zhu WT, Jing JW. An efficient elliptic curve cryptography signature server with GPU acceleration. IEEE Trans. on Information Forensics and Security, 2017, 12(1): 111–122. [doi: 10.1109/TIFS.2016.2603974]

[70] Dong JK, Zheng FY, Emmart N, Lin JQ, Weems C. sDPF-RSA: Utilizing floating-point computing power of GPUs for massive digital signature computations. In: Proc. of the 2018 IEEE Int'l Parallel and Distributed Processing Symp. (IPDPS). Vancouver: IEEE, 2018. 599–609. [doi: 10.1109/IPDPS.2018.00069]

[71] Dong JK, Zheng FY, Cheng JJ, Lin JQ, Pan WQ, Wang ZY. Towards high-performance X25519/448 key agreement in general purpose GPUs. In: Proc. of the 2018 IEEE Conf. on Communications and Network Security (CNS). Beijing: IEEE, 2018. 1–9. [doi: 10.1109/CNS.2018.8433161]

[72] Gao LL, Zheng FY, Emmart N, Dong JK, Lin JQ, Weems C. DPF-ECC: Accelerating elliptic curve cryptography with floating-point computing power of GPUs. In: Proc. of the 2020 IEEE Int'l Parallel and Distributed Processing Symp. (IPDPS). New Orleans: IEEE, 2020. 494–504. [doi: 10.1109/IPDPS47924.2020.00058]

[73] Dong JK, Zheng FY, Lin JQ, Liu Z, Xiao F, Fan G. EC-ECC: Accelerating elliptic curve cryptography for edge computing on embedded GPU TX2. ACM Trans. on Embedded Computing Systems, 2022, 21(2): 16. [doi: 10.1145/3492734]

[74] Dong JK, Zhang PC, Sun KS, Xiao F, Zheng FY, Lin JQ. EG-FourQ: An embedded GPU based efficient ECC cryptography accelerator for edge computing. IEEE Trans. on Industrial Informatics, 2023, 19(6): 7291–7300. [doi: 10.1109/TII.2022.3205355]

[75] Hu XY, He DB, Luo M, Peng C, Feng Q, Huang XY. High-performance implementation of the identity-based signature scheme in IEEE P1363 on GPU. ACM Trans. on Embedded Computing Systems, 2023, 22(2): 25. [doi: 10.1145/3564784]

[76] Emmart N, Weems C. Pushing the performance envelope of modular exponentiation across multiple generations of GPUs. In: Proc. of the 2015 IEEE Int'l Parallel and Distributed Processing Symp. Hyderabad: IEEE, 2015. 166–176. [doi: 10.1109/IPDPS.2015.69]

[77] Yang Y, Guan Z, Sun HP, Chen Z. Accelerating RSA with fine-grained parallelism using GPU. In: Proc. of the 11th Int'l Conf. on Information Security Practice and Experience. Beijing: Springer, 2015. 454–468. [doi: 10.1007/978-3-319-17533-1_31]

[78] Microsoft. Microsoft's FourQ library. 2016. https://github.com/microsoft/FourQlib

[79] Cui SJ, Großschädl J, Liu Z, Xu QL. High-speed elliptic curve cryptography on the NVIDIA GT200 graphics processing unit. In: Proc. of the 10th Int'l Conf. on Information Security Practice and Experience. Fuzhou: Springer, 2014. 202–216. [doi: 10.1007/978-3-319-06320-1_16]

[80] Mahé EM, Chauvet JM. Fast GPGPU-based elliptic curve scalar multiplication. 2014. https://eprint.iacr.org/2014/198.pdf

[81] Zheng FY, Pan WQ, Lin JQ, Jing JW, Zhao Y. Exploiting the potential of GPUs for modular multiplication in ECC. In: Proc. of the 15th Int'l Workshop on Information Security Applications. Jeju Island: Springer, 2014. 295–306. [doi: 10.1007/978-3-319-15087-1_23]

[82] Gao L, Zheng F, Wei R, *et al*. DPF-ECC: A framework for efficient ECC with double precision floating-point computing power. IEEE Trans. on Information Forensics and Security, 2021, 16: 3988–4002. [doi: 10.1109/TIFS.2021.3098987]

[83] Fogel LA. Cryptanalysis of the mceliece cryptosystem on GPGPUs. 2015. https://www.semanticscholar.org/paper/Cryptanalysis-of-the-McEliece-Cryptosystem-on-Major/186cb696d28eaefe2b506063e4d4188adf74dc44?p2df

[84] Elsobky AM, Farag AK, Keshk A. Efficient implementation of McEliece cryptosystem on graphic processing unit. In: Proc. of the 10th Int'l Conf. on Informatics and Systems. Giza Egypt: ACM, 2016. 247–253. [doi: 10.1145/2908446.2908491]

[85] Bos JW, Friedberger SJ. Faster modular arithmetic for isogeny-based crypto on embedded devices. Journal of Cryptographic Engineering, 2020, 10(2): 97–109. [doi: 10.1007/s13389-019-00214-6]

[86] Duong-Ngoc P, Tan TN, Lee H. Efficient NewHope cryptography based facial security system on a GPU. IEEE Access, 2020, 8: 108158–108168. [doi: 10.1109/ACCESS.2020.3000316]

[87] Seo SC. SIKE on GPU: Accelerating supersingular isogeny-based key encapsulation mechanism on graphic processing units. IEEE Access, 2021, 9: 116731–116744. [doi: 10.1109/ACCESS.2021.3106551]

[88] Wright J, Gowanlock M, Philabaum C, Cambou B. A CRYSTALS-Dilithium response-based cryptography engine using GPGPU. In:

Proc. of the 2021 Future Technologies Conf. (FTC). Vancouver: Springer, 2021. 32–45. [doi: 10.1007/978-3-030-89912-7_3]

[89] Lee K, Gowanlock M, Cambou B. SABER-GPU: A response-based cryptography algorithm for SABER on the GPU. In: Proc. of the 26th IEEE Pacific Rim Int'l Symp. on Dependable Computing (PRDC). Perth: IEEE, 2021. 123–132. [doi: 10.1109/PRDC53464.2021.00024]

[90] Gao YW, Xu J, Wang HB. cuNH: Efficient GPU implementations of post-quantum KEM NewHope. IEEE Trans. on Parallel and Distributed Systems, 2022, 33(3): 551–568. [doi: 10.1109/TPDS.2021.3097277]

[91] Nejatollahi H, Shahhosseini S, Cammarota R, Dutt N. Exploring energy efficient architectures for RLWE lattice-based cryptography. Journal of Signal Processing Systems, 2021, 93(10): 1139–1148. [doi: 10.1007/s11265-020-01627-x]

[92] Gupta N, Jati A, Chauhan AK, Chattopadhyay A. PQC acceleration using GPUs: FrodoKEM, NewHope, and Kyber. IEEE Trans. on Parallel and Distributed Systems, 2021, 32(3): 575–586. [doi: 10.1109/TPDS.2020.3025691]

[93] Wan LP, Zheng FY, Fan G, Wei R, Gao LL, Wang YW, Lin JQ, Dong JK. A novel high-performance implementation of CRYSTALS-Kyber with AI accelerator. In: Proc. of the 27th European Symp. on Research in Computer Security. Copenhagen: Springer, 2022. 514–534. [doi: 10.1007/978-3-031-17143-7_25]

[94] Lee WK, Seo H, Zhang ZF, Hwang SO. TensorCrypto: High throughput acceleration of lattice-based cryptography using tensor core on GPU. IEEE Access, 2022, 10: 20616–20632. [doi: 10.1109/ACCESS.2022.3152217]

[95] Lee WK, Seo H, Hwang SO, Achar R, Karmakar A, Mera JMB. DPCrypto: Acceleration of post-quantum cryptography using dot-product instructions on GPUs. IEEE Trans. on Circuits and Systems I: Regular Papers, 2022, 69(9): 3591–3604. [doi: 10.1109/TCSI.2022.3176966]

[96] Lee WK, Hwang SO. High throughput implementation of post-quantum key encapsulation and decapsulation on GPU for Internet of Things applications. IEEE Trans. on Services Computing, 2022, 15(6): 3275–3288. [doi: 10.1109/TSC.2021.3103956]

[97] Wang ZH, Dong XS, Chen H, Kang Y. Efficient GPU implementations of post-quantum signature XMSS. IEEE Trans. on Parallel and Distributed Systems, 2023, 34(3): 938–954. [doi: 10.1109/TPDS.2022.3233348]

[98] Wang W, Hu Y, Chen LM, Huang XM, Sunar B. Accelerating fully homomorphic encryption using GPU. In: Proc. of the 2012 IEEE Conf. on High Performance Extreme Computing. Waltham: IEEE, 2012. 1–5. [doi: 10.1109/HPEC.2012.6408660]

[99] Dai W, Doröz Y, Sunar B. Accelerating NTRU based homomorphic encryption using GPUs. In: Proc. of the 2014 IEEE High Performance Extreme Computing Conf. (HPEC). Waltham: IEEE, 2014. 1–6. [doi: 10.1109/HPEC.2014.7041001]

[100] Dai W, Doröz Y, Sunar B. Accelerating SWHE based pirs using GPUs. In: Proc. of the 2015 Int'l Conf. on Financial Cryptography and Data Security. San Juan: Springer, 2015. 160–171. [doi: 10.1007/978-3-662-48051-9_12]

[101] Dai W, Sunar B. cuHE: A homomorphic encryption accelerator library. In: Proc. of the 2nd Int'l Conf. on Cryptography and Information Security in the Balkans. Koper: Springer, 2016. 169–186. [doi: 10.1007/978-3-319-29172-7_11]

[102] Al Badawi A, Veeravalli B, Mun CF, Aung KMM. High-performance FV somewhat homomorphic encryption on GPUs: An implementation using CUDA. IACR Trans. on Cryptographic Hardware and Embedded Systems, 2018, 2018(2): 70–95. [doi: 10.13154/tches.v2018.i2.70-95]

[103] Kim S, Jung W, Park J, Ahn JH. Accelerating number theoretic transformations for bootstrappable homomorphic encryption on GPUs. In: Proc. of the 2020 IEEE Int'l Symp. on Workload Characterization (IISWC). Beijing: IEEE, 2020. 264–275. [doi: 10.1109/IISWC50251.2020.00033]

[104] Al Badawi A, Veeravalli B, Lin J, Xiao N, Kazuaki M, Mi AKM. Multi-GPU design and performance evaluation of homomorphic encryption on GPU clusters. IEEE Trans. on Parallel and Distributed Systems, 2021, 32(2): 379–391. [doi: 10.1109/TPDS.2020.3021238]

[105] Goey JZ, Lee WK, Goi BM, Yap WS. Accelerating number theoretic transform in GPU platform for fully homomorphic encryption. The Journal of Supercomputing, 2021, 77(2): 1455–1474. [doi: 10.1007/s11227-020-03156-7]

[106] Jung W, Kim S, Ahn JH, Cheon JH, Lee Y. Over 100× faster bootstrapping in fully homomorphic encryption through memory-centric optimization with GPUs. IACR Trans. on Cryptographic Hardware and Embedded Systems, 2021. 114–148.

[107] Shen SY, Yang H, Liu Y, Liu Z, Zhao YL. CARM: CUDA-accelerated RNS multiplication in word-wise homomorphic encryption schemes for internet of things. IEEE Trans. on Computers, 2023, 72(7): 1999–2010. [doi: 10.1109/TC.2022.3227874]

[108] Zhai YJ, Ibrahim M, Qiu YQ, Boemer F, Chen ZZ, Titov A, Lyashevsky A. Accelerating encrypted computing on Intel GPUs. In: Proc. of the 2022 IEEE Int'l Parallel and Distributed Processing Symp. (IPDPS). Lyon: IEEE, 2022. 705–716. [doi: 10.1109/IPDPS53621.2022.00074]

[109] Fan SY, Wang ZW, Xu WZ, Hou R, Meng D, Zhang MZ. TensorFHE: Achieving practical computation on encrypted data using GPGPU. In: Proc. of the 2023 IEEE Int'l Symp. on High-performance Computer Architecture (HPCA). Montreal: IEEE, 2023. 922–934.

[doi: 10.1109/HPCA56546.2023.10071017]

[110] Wang ZW, Li PN, Hou R, Li ZH, Cao JF, Wang XF, Meng D. HE-booster: An efficient polynomial arithmetic acceleration on GPUs for fully homomorphic encryption. IEEE Trans. on Parallel and Distributed Systems, 2023, 34(4): 1067–1081. [doi: 10.1109/TPDS.2022.3228628]

[111] Jung W, Lee E, Kim S, Kim J, Kim N, Lee K, Min C, Cheon JH, Ahn JH. Accelerating fully homomorphic encryption through architecture-centric analysis and optimization. IEEE Access, 2021, 9: 98772–98789. [doi: 10.1109/ACCESS.2021.3096189]

[112] Yang H, Shen SY, Dai WC, Zhou L, Liu Z, Zhao YL. Implementing and benchmarking word-wise homomorphic encryption schemes on GPU. 2023. https://eprint.iacr.org/2023/049.pdf

[113] Türkoglu ER, Özcan AŞ, Ayduman C, Mert AC, Öztürk E, Savaş E. An accelerated gpu library for homomorphic encryption operations of BFV scheme. In: Proc. of the 2022 IEEE Int'l Symp. on Circuits and Systems (ISCAS). Austin: IEEE, 2022. 1155–1159. [doi: 10.1109/ISCAS48785.2022.9937503]

[114] National Cryptography Administration. Cryptography Law of the People's Republic of China. 2023 (in Chinese). https://www.oscca.gov.cn/sca/xxgk/2023-06/04/content_1057225.shtml

[115] National Cryptography Administration. Announcement of the National Cryptography Administration on the release of SM3 password hash algorithm. 2010 (in Chinese). https://www.oscca.gov.cn/sca/xxgk/2010-12/17/content_1002389.shtml

[116] National Cryptography Administration. Announcement of the National Cryptography Administration on the release of two cryptographic industry standards, including SM9 identification password algorithm. 2016 (in Chinese). https://www.oscca.gov.cn/sca/xxgk/2016-03/28/content_1002407.shtml

[117] Shamir A. Identity-based cryptosystems and signature schemes. In: Proc. of the 1985 Workshop on the Theory and Application of Cryptographic Techniques. Santa Barbara: Springer, 1985. 47–53. [doi: 10.1007/3-540-39568-7_5]

[118] National Cryptography Administration. Our SM9 key exchange protocol is officially ISO/IEC international standard. 2021 (in Chinese). https://www.sca.gov.cn/sca/xwdt/2021-11/05/content_1060901.shtml

[119] Wang JP, Zhang T, Zhang B, Jeremy-Gillbanks, Xin Z. An innovative FPGA implementations of the secure frequency hopping communication system based on the improved ZUC algorithm. IEEE Access, 2022, 10: 54634–54648. [doi: 10.1109/ACCESS.2022.3176609]

[120] AspenCore analyst team. Survey report of 35 domestic processor chip (CPU/GPU/FPGA) manufacturers. 2022 (in Chinese). http://www.infosecworld.cn/index.php?m=content&c=index&a=show&catid=40&id=1149

[121] Deng B, Sun JG. Review on development of domestic embedded processor. Aeronautical Computing Technique, 2021, 51(1): 120–124 (in Chinese with English abstract). [doi: 10.3969/j.issn.1671-654X.2021.01.028]

[122] CAICT. Privacy protection computing and compliance applications research report (2021). 2021 (in Chinese). http://www.caict.ac.cn/kxyj/qwfb/ztbg/202104/t20210401_372713.htm

[123] Yao AC. Protocols for secure computations. In: Proc. of the 23rd Annual Symp. on Foundations of Computer Science (SFCS 1982). Chicago: IEEE, 1982. 160–164. [doi: 10.1109/SFCS.1982.38]

[124] Tan ZW, Zhang LF. Survey on privacy preserving techniques for machine learning. Ruan Jian Xue Bao/Journal of Software, 2020, 31(7): 2127–2156 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6052.htm [doi: 10.13328/j.cnki.jos.006052]

[125] Guo JJ, Wang QX, Xu X, Wang TY, Lin JQ. Secure multiparty computation and application in machine learning. Journal of Computer Research and Development, 2021, 58(10): 2163–2186 (in Chinese with English abstract). [doi: 10.7544/issn1000-1239.2021.20210626]

[126] Yao ACC. How to generate and exchange secrets. In: Proc. of the 27th Annual Symp. on Foundations of Computer Science (SFCS 1986). Toronto: IEEE, 1986. 162–167. [doi: 10.1109/SFCS.1986.25]

[127] Wang X, Malozemoff AJ, Katz J. EMP-toolkit: Efficient multiparty computation toolkit. 2016. https://github.com/emp-toolkit/emp-readmem

[128] Goldreich O, Micali S, Wigderson A. How to play any mental game, or a completeness theorem for protocols with honest majority. In: Goldreich O, ed., Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali. New York: ACM, 2019. 307–328. [doi: 10.1145/3335741.3335755]

[129] Ben-Or M, Goldwasser S, Wigderson A. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Goldreich O, ed., Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali. New York: ACM, 2019. 351–371. [doi: 10.1145/3335741.3335756]

[130] Demmler D, Schneider T, Zohner M. ABY—A framework for efficient mixed-protocol secure two-party computation. In: Proc. of the 22nd Annual Network and Distributed System Security Symp. San Diego: NDSS, 2015.

[131] Mohassel P, Rindal P. ABY^3: A mixed protocol framework for machine learning. In: Proc. of the 2018 ACM SIGSAC Conf. on

Computer and Communications Security. Toronto: ACM, 2018. 35–52. [doi: 10.1145/3243734.3243760]

[132] Keller M. MP-SPDZ: A versatile framework for multi-party computation. In: Proc. of the 2020 ACM SIGSAC Conf. on Computer and Communications Security. ACM, 2020. 1575–1590. [doi: 10.1145/3372297.3417872]

[133] Malkhi D, Nisan N, Pinkas B, Sella Y. FairPlay—A secure two-party computation system. In: Proc. of the 13th Conf. on USENIX Security Symp. San Diego: USENIX Association, 2004. 20.

[134] Pu S, Duan P, Liu JC. FastPlay—A parallelization model and implementation of SMC on CUDA based GPU cluster architecture. 2011. https://eprint.iacr.org/2011/097.pdf

[135] Watson JL, Wagh S, Popa RA. Piranha: A GPU platform for secure computation. In: Proc. of the 31st USENIX Security Symp. (USENIX Security 22). 2022. 827–844.

[136] Frederiksen TK, Jakobsen TP, Nielsen JB. Faster maliciously secure two-party computation using the GPU. In: Proc. of the 9th Int'l Conf. on Security and Cryptography for Networks. Amalfi: Springer, 2014. 358–379. [doi: 10.1007/978-3-319-10879-7_21]

[137] Lindell Y. Fast cut-and-choose-based protocols for malicious and covert adversaries. Journal of Cryptology, 2016, 29(2): 456–490. [doi: 10.1007/s00145-015-9198-0]

[138] Frederiksen TK, Nielsen JB. Fast and maliciously secure two-party computation using the GPU. In: Proc. of the 11th Int'l Conf. on Applied Cryptography and Network Security. Banff: Springer, 2013. 339–356. [doi: 10.1007/978-3-642-38980-1_21]

[139] Shelat A, Shen CH. Fast two-party secure computation with minimal assumptions. In: Proc. of the 2013 ACM SIGSAC Conf. on Computer & Communications Security. Berlin: ACM, 2013. 523–534. [doi: 10.1145/2508859.2516698]

[140] Zhang F, Chen Z, Zhang CY, Zhou AC, Zhai JD, Du XY. An efficient parallel secure machine learning framework on GPUs. IEEE Trans. on Parallel and Distributed Systems, 2021, 32(9): 2262–2276. [doi: 10.1109/TPDS.2021.3059108]

[141] Tan SJ, Knott B, Tian Y, Wu DJ. CryptGPU: Fast privacy-preserving machine learning on the GPU. In: Proc. of the 2021 IEEE Symp. on Security and Privacy (SP). San Francisco: IEEE, 2021. 1021–1038. [doi: 10.1109/SP40001.2021.00098]

[142] NVIDIA. CUDA C++ programming guide. 2015. http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html

[143] Wikipedia. Wikipedia: List of NVIDIA graphics processing units. 2014. http://en.wikipedia.org/wiki/Comparison_of_NVIDIA_Graphics_Processing_Units

[144] NVIDIA. The tensor core GPU architecture designed to bring AI to every industry. 2017. https://www.nvidia.com/en-in/data-center/volta-gpu-architecture/

[145] Vasiliadis G, Athanasopoulos E, Polychronakis M, Ioannidis S. PixelVault: Using GPUs for securing cryptographic operations. In: Proc. of the 2014 ACM SIGSAC Conf. on Computer and Communications Security. Scottsdale: ACM, 2014. 1131–1142. [doi: 10.1145/2660267.2660316]

[146] Halderman JA, Schoen SD, Heninger N, Clarkson W, Paul W, Calandrino JA, Feldman AJ, Appelbaum J, Felten EW. Lest we remember: Cold-boot attacks on encryption keys. Communications of the ACM, 2009, 52(5): 91–98. [doi: 10.1145/1506409.1506429]

[147] Durumeric Z, Li F, Kasten J, Amann J, Beekman J, Payer M, Weaver N, Adrian D, Paxson V, Bailey M, Halderman JA. The matter of heartbleed. In: Proc. of the 2014 Conf. on Internet Measurement Conf. Vancouver: ACM, 2014. 475–488. [doi: 10.1145/2663716.2663755]

[148] Müller T, Freiling FC, Dewald A. TRESOR runs encryption securely outside RAM. In: Proc. of the 20th USENIX Conf. on Security. San Francisco: USENIX Association, 2011. 17.

[149] Simmons P. Security through amnesia: A software-based solution to the cold boot attack on disk encryption. In: Proc. of the 27th Annual Computer Security Applications Conf. Orlando: ACM, 2011. 73–82. [doi: 10.1145/2076732.2076743]

[150] Guan L, Lin JQ, Luo B, Jing JW. Copker: Computing with private keys without RAM. In: Proc. of the 21st Annual Network and Distributed System Security Symp. San Diego: NDSS. 2014. 23–26.

[151] Guan L, Lin JQ, Luo B, Jing JW, Wang J. Protecting private keys against memory disclosure attacks using hardware transactional memory. In: Proc. of the 2015 IEEE Symp. on Security and Privacy. San Jose: IEEE, 2015. 3–19. [doi: 10.1109/SP.2015.8]

附中文参考文献:

[4] 国家密码管理局. 国务院常务会议审议通过《商用密码管理条例 (修订草案)》. 2023. https://www.oscca.gov.cn/sca/xwdt/2023-04/20/content_1061005.shtml

[11] 中华人民共和国国家质量监督检验检疫总局, 中国国家标准化管理委员会. GB/T 32907-2016 信息安全技术 SM4 分组密码算法. 北京: 中国标准出版社, 2017.

[13] 中华人民共和国国家质量监督检验检疫总局, 中国国家标准化管理委员会. GB/T 33133.1-2016 信息安全技术 祖冲之序列密码算

法 第 1 部分: 算法描述. 北京: 中国标准出版社, 2016.
[22] 张雁, 林英, 郝林. 椭圆曲线公钥密码体制的研究热点综述. 计算机工程, 2004, 30(3): 127–129. [doi: 10.3969/j.issn.1000-3428.2004.03.052]
[27] 国家密码管理局. 国家密码管理局关于发布《SM2 椭圆曲线公钥密码算法》公告. 2010. https://oscca.gov.cn/sca/xxgk/2010-12/17/content_1002386.shtml
[114] 国家密码管理局. 中华人民共和国密码法. 2023. https://www.oscca.gov.cn/sca/xxgk/2023-06/04/content_1057225.shtml
[115] 国家密码管理局. 国家密码管理局关于发布《SM3 密码杂凑算法》公告. 2010. https://www.oscca.gov.cn/sca/xxgk/2010-12/17/content_1002389.shtml
[116] 国家密码管理局. 国家密码管理局关于发布《SM9 标识密码算法》等 2 项密码行业标准公告. 2016. https://www.oscca.gov.cn/sca/xxgk/2016-03/28/content_1002407.shtml,
[118] 国家密码管理局. 我国 SM9 密钥交换协议正式成为 ISO/IEC 国际标准. 2021. https://www.sca.gov.cn/sca/xwdt/2021-11/05/content_1060901.shtml
[120] AspenCore 分析师团队. 35 家国产处理器芯片 (CPU/GPU/FPGA) 厂商调研报告. 2022. http://www.infosecworld.cn/index.php?m=content&c=index&a=show&catid=40&id=1149
[121] 邓豹, 孙靖国. 国产嵌入式处理器发展综述. 航空计算技术, 2021, 51(1): 120–124. [doi: 10.3969/j.issn.1671-654X.2021.01.028]
[122] 中国信通院. 隐私保护计算与合规应用研究报告 (2021 年). 2021. http://www.caict.ac.cn/kxyj/qwfb/ztbg/202104/t20210401_372713.htm
[124] 谭作文, 张连福. 机器学习隐私保护研究综述. 软件学报, 2020, 31(7): 2127–2156. http://www.jos.org.cn/1000-9825/6052.htm [doi: 10.13328/j.cnki.jos.006052]
[125] 郭娟娟, 王琼霄, 许新, 王天雨, 林璟锵. 安全多方计算及其在机器学习中的应用. 计算机研究与发展, 2021, 58(10): 2163–2186. [doi: 10.7544/issn1000-1239.2021.20210626]

董建阔(1992－), 男, 博士, 讲师, CCF 专业会员, 主要研究领域为公钥密码, 后量子密码, 并行计算.

郑昉昱(1988－), 男, 博士, 助理研究员, CCF 学生会员, 主要研究领域为应用密码学, 公钥密码, 并行计算.

黄跃花(1999－), 女, 硕士生, 主要研究领域为应用密码学, 并行计算.

林璟锵(1979－), 男, 博士, 教授, 博士生导师, 主要研究领域为密码工程, 系统安全.

付宇笙(1999－), 男, 硕士生, CCF 学生会员, 主要研究领域为公钥密码, 后量子密码, 并行计算.

董振江(1970－), 男, 博士, 教授, 博士生导师, 主要研究领域为网络空间安全, 人工智能.

肖甫(1980－), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为网络空间安全, 物联网技术.