基于 GPU 的高性能密码计算

郑昉昱[1,2] 董建阔[1,2,3] 林璟锵[1,2,3] 高莉莉[1,2,3]
[1](中国科学院数据与通信保护研究教育中心 北京 100093)
[2](信息安全国家重点实验室(中国科学院信息工程研究所) 北京 100093)
[3](中国科学院大学网络空间安全学院 北京 100049)
(zhengfangyu@iie.ac.cn)

High-Performance Cryptographic Computations in GPUs

Zheng Fangyu[1,2], Dong Jiankuo[1,2,3], Lin Jingqiang[1,2,3], and Gao Lili[1,2,3]
[1](*Data Assurance and Communication Security Research Center*, *Chinese Academy of Sciences*, *Beijing* 100093)
[2](*State Key Laboratory of Information Security* (*Institute of Information Engineering*, *Chinese Academy of Sciences*), *Beijing* 100093)
[3](*School of Cyber Security*, *University of Chinese Academy of Sciences*, *Beijing* 100049)

Abstract Cryptology is an important foundation and tool of network security. In recent years, with the continuous and rapid development of big data industry, e-commerce and cloud computing, the amount of users, traffic volumes and the corresponding cryptographic calculations faced by various service providers are also rapidly rising. In response to this situation, researchers began to break the conventional pattern that cryptographic algorithms were implemented by CPUs, ASICs, and FPGAs, migrate them to various parallel computing platforms, such as graphics processing units (GPUs). Driven by huge demand of graphics rendering, artificial intelligence, etc., GPUs gain more than ten times of the computing power promotion over the last decade. Such performance advantages help the GPU-based cryptography implementations outperform others by a wide margin, and give them great potential. This paper summarizes the current progress of the GPU-based cryptography implementation, and gives a brief analysis of its development tendency.

Key words graphics processing unit; RSA; elliptic curve cryptography; cryptographic computation; parallel computing

摘 要 密码技术是保障网络安全的重要基石和工具.近年来,随着大数据行业、电子商务和云计算技术的持续快速发展,各个服务商面对的用户量、业务量和相应的密码计算量也在急速地攀升;面向这一情况,研究人员开始打破密码算法由 CPU,ASIC,FPGA 实现的传统格局,将密码算法迁移至 GPU 等各类并行计算平台上.受到高分辨率图形渲染、人工智能的巨大需求所带动,GPU 在过去 10 年获得超过 10 倍的计算性能提升,大幅领先于其他计算平台.这也使得基于 GPU 的高性能算法实现的性能远超其他平台的同类实现,显示出了 GPU 在密码算法实现领域的巨大潜能.内容主要

收稿日期:2018-09-30
基金项目:国家自然科学基金项目(61772518);国家重点研发计划网络空间安全重点专项(2017YFB0802100)

包括2部分:一是总结基于GPU的高性能密码计算的发展和研究现状;二是简要分析它未来的发展趋势.

关键词 图形处理器;RSA;椭圆曲线密码学;密码计算;并行计算

中图法分类号 TP309

密码技术是保障信息传输安全的重要工具.近些年来,随着大数据行业、电子商务和云计算技术的持续快速发展,各个服务商面对的用户量、业务量和相应的密码计算量也在急速地攀升,已有的密码算法计算技术已无法跟上这一发展趋势,特别是基于非对称密码算法的数字签名和密钥交换等.目前主流的非对称密码算法需要复杂的数学计算,比如RSA算法体系所依赖的模幂计算和椭圆曲线密码(elliptic curve cryptography, ECC)算法体系的点乘计算,这些高密集计算载荷成为了各类密码相关服务(如SSL/TLS,HTTPS)中的主要计算瓶颈.在2018年支付宝"双11"中,峰值交易量超过了每秒49.1万笔;假设每笔交易中服务器需要完成2次签名生成/签名验证操作(1次用于验证买家身份,1次用于对交易本身进行数字签名),那么总计需要每秒98.2万次的签名生成/签名验证操作.以目前的技术水平,市面上大多数已有的密码计算设备的非对称密码算法速率最多仅有每秒数万次的水平,尚不足以满足这一需求.

与此同时,随着高性能计算、大数据、人工智能等产业的驱动,各类并行平台在近数十年来取得了迅猛的发展.于是研究人员开始打破密码算法由CPU,ASIC,FPGA实现的传统格局,将密码算法迁移至各类并行计算平台上,并取得了不俗的效果.

1 研究现状

目前,主要的并行计算平台包括桌面GPU(graphics processing unit)、嵌入式GPU、Xeon Phi等,各国的研究人员在这些平台上实现了各类密码算法.

1.1 目前主流的并行计算平台

研究者最常使用的并行计算平台为NVIDIA GPU.在2007年,著名显卡厂商NVIDIA发布了统一计算设备架构(compute unified device architecture, CUDA).CUDA可以支持在NVIDIA的GPU上完成通用计算,使得研究人员、工程师都可以方便地使用GPU上强大的计算能力完成各类科学计算.自2007年以来,NVIDIA GPU已历经数代,包括Tesla,Fermi,Kepler,Maxwell,Pascal,Volta等,计算性能由380 GFLOPS一路飙升到15 000 GFLOPS,增长超过30倍,展现出强大的计算性能和潜力.除了传统的桌面式GPU外,一些低功耗、高性能的嵌入式GPU计算平台纷纷面世,比如NVIDIA的Tegra K1/X1/X2等,在不到10 W的功耗下可提供超过1 TFLOPS的浮点数计算能力,显示了在自动驾驶、物联网领域的强大性能优势.

同时期,著名CPU厂商Intel也不甘落后,于2012年宣布了集成众核的第1款产品Intel Xeon Phi Knights Corner.这也是一款具有极大的吞吐量,又以向量指令为主要计算能力的协处理器.它和NVIDIA GPU一样,都可以完成通用计算.Xeon Phi目前已经推出了包括Knights Landing,Knights Mill等产品,也是一款非常具有潜力的并行计算平台.

相比于GPU,Xeon Phi上的密码算法实现相对较少,本文主要介绍GPU上的相关实现.

1.2 GPU平台上的已有密码算法实现

研究者在GPU平台上实现了包括对称密码算法、非对称密码算法和密码杂凑算法等一系列算法.事实上,由于对称密码算法和密码杂凑算法本身的实现效率就相当可观,因此,它们在并行计算平台上实现时性能瓶颈已经不在密码计算上,而在其他方面,如硬件层面的数据传输等.比如对称密码算法AES,SM4在GPU上可达到数百Gbps的速率[1-2],而其接口PCIE 3.0 X16最多仅能达到64 Gbps.因此,研究人员在非对称密码算法的实现上投入了更多的精力.

非对称密码学又称公钥密码学,由Diffie和Hellman[3]在1976年提出.公钥密码学很好地弥

补了在对称密码体系中的缺陷，目前被广泛用于数字签名、密钥协商技术中，以保证信息传输的完整性、对发送者身份的认证以及防止交易中的抵赖发生和通信双方共享密钥的协商生成.

目前，主流的非对称密码算法包括以下三大类，定义于 NIST 的 FIPS 186-4 [4]：1）有限域密码学，如 ElGamal 及其变种数字签名算法（digital signature algorithm，DSA）；2）椭圆曲线密码学，如 NIST 的 ECDSA（elliptic curve digital signature algorithm）、韩国的 EC-KCDSA（Korean certificate-based digital signature algorithm）和我国的 SM2 椭圆曲线公钥密码算法；3）整数分解密码学，如 RSA.

研究者们充分利用 GPU 强大的算术计算性能，在包括 Eurocrypt，CHES，IPDPS 在内的国际顶级安全和并行计算会议上，针对以上主流非对称密码算法发表了大量研究成果，展示了并行计算平台在非对称密码算法加速方面的显著优势. 无论是早期的文献[5-8]，还是最近 2 年的相关文献[9-10]在 GPU 上的非对称密码算法实现均获得了同时代同一水平 CPU，ASIC，FPGA 的几倍乃至几十倍的性能提升.

对于非对称密码算法的并行实现可以根据算法是否并行化以及算法并行的粒度进行划分，主要包括完全串行、细粒度并行和粗粒度并行 3 种实现方式.

1.2.1 完全串行方案

文献[5-6，10-14]采用的类似于 CPU 上的串行编程方式，利用一个线程来实现一个完整的密码算法. 这种方法有几个好处：一是不需要为 GPU 专门设计特定的算法，将已有的普通串行算法直接移植即可；二是节省了线程同步和线程间通信带来的性能损失. 但是，完全串行方案往往会造成比较大的延迟：文献[5]的 2 048 b 大小的模幂延迟达到 55 s；文献[13]实现的 RSA-1024 解密操作，单次计算延迟达到 150 ms.

使用完全串行计算的方案一般来说更常应用在 ECC 上，一方面因为其更难以并行，另一方面由于 ECC 计算量较小，并且可以通过预计算的方式加速，即使串行执行延迟也不会很大. 文献[10]采用这种方法实现了 ECDSA，是目前性能最优的 ECC 算法实现.

1.2.2 细粒度并行方案

如果单纯为了达到高吞吐，完全串行方案是一个非常有效的途径，尽可能降低了算法并行化所带来的额外计算负载. 但是对于非对称密码算法，由于计算量较大，单次计算延迟非常大，尤其是 RSA，因此有时不得不将算法进行并行化以降低延迟；同时，对于并行计算而言，应当尽量使用片上存储（寄存器、GPU 的共享内存等）进行计算，而非对称密码算法涉及到的变量普遍较多，有时不得不将算法进行并行化，以使得每个线程内所分配的片上存储足以容纳所需的变量. 根据实施并行化的算法层次划分，可分为细粒度并行和粗粒度并行方案.

文献[5-6，9，15-26]采用的就是细粒度并行方法，从底层模乘算法开始进行拆分并行化，由多个线程计算一个大整数模乘算法. 这种方案最为适合使用的是剩余数系统（residue number system，RNS），包括文献[5-6，15-17，20-21]. RNS 的优势非常明显. 首先，它非常容易进行并行化，由于单个字的运算是完全独立的，1 个线程只需要负责 1 个字的运算，不需要向其他线程传递信息；其次，它所需乘法数量少，对于 $n\times n$ 字乘法只需要 n 次乘法. 但是 RNS 也有它的缺点，大整数表示和 RNS 表示之间的相互转换需要大量运算，而且利用 Montgomery 算法进行模约减算法的运算量也非常巨大. 根据文献[5-6，20]的对比，最终实现中，RNS 虽然有着更好的延迟，但运算吞吐率只有 Montgomery 乘法的一半左右.

文献[19，22-23]采用文献[27]中 Montgomery 乘法的分离操作数扫描（separated operand scanning，SOS）模式进行实现，采用向量指令或者类似于向量指令的形式，以字为单位（32 b 或 64 b 整型数）进行并行化. 但是这种单字的完全并行化会在线程同步和线程间通信上消耗大量资源，很大程度地降低了实际实现性能，其实现结果并不是非常令人满意.

目前比较主流的方案采用的方法是 Montgomery 乘法的非完全并行方案，即采用多个线程完成一个 Montgomery 乘法，但是每个线程处理多个字. 相比于 RNS 和完全并行方案，非完全并行方案可以在片上资源和延迟满足要求的情况

下，尽可能地发挥计算潜能，提高吞吐率，目前性能最优的几个 RSA 实现[9,24-26]都使用了该方案.

1.2.3 粗粒度并行方案

粗粒度方式并行方案指的是在底层算法上还采用串行的方式，在模幂或椭圆曲线算术层面采用并行的方式，即基于单线程实现有限域算法，使用多个线程处理模幂或椭圆曲线算术.

文献[7]使用 Montgomery 乘法的 SOS 进行 280 b 素域上的椭圆曲线算术，计算时利用的是单精度浮点数计算资源，每个线程作为一个乘法器，实现完整的乘法，并由多个线程完成一个模乘. 文献[8]采用粗粒度的并行方法，从点加/倍点算法层次开始并行化，以求获得较低的单次计算延迟，即：仍采用单个线程进行处理单个大整数模乘算法，但使用多个线程协同处理一个点加/倍点算法.

粗粒度的并行方案在一定程度上结合了串行方案和粗粒度并行的优点，但由于算法上的不对称特点（主要指椭圆曲线算术），并行算法在设计时比较困难，而且也容易造成计算资源的浪费. 因此这种方法并不是目前的主流.

1.3 RSA 和 ECC 在 GPU 平台上的典型实现

本节各选取了 RSA 和 ECC 的一个典型实现，对其基本实现思路和优化方法进行简要介绍.

1.3.1 RSA 算法典型实现

文献[9]提供了一种利用 GPU 浮点数完成 RSA 计算的方法，考虑到 RSA 所需的计算量和变量都较多，该方案采用了非完全并行的“细粒度并行”方案，即利用若干线程协同计算一个 Montgomery 乘法，各个线程分别处理 Montgomery 乘法的若干字. 该方案在 GTX Titan Black 平台上，实现的 RSA-2048 签名速度达到每秒 5.3 万次，验签速度达到每秒 124 万次，超过同平台整数实现 30%以上. 该方案根据 GPU 的特点，进行了以下具体优化：

1）浮点数实现. 本方案最大的亮点是使用 GPU 上超强的浮点数计算性能实现 RSA 计算，改变了以往实现采用 GPU 定点数的主流方式. 该方案针对 Montgomery 乘法的特点，利用 GPU 性能突出的浮点数积和熔加运算（fused multiply-add，FMA），使用包括浮点数符号位在内的计算资源，完整地设计出基于浮点数的并行 Montgomery 乘法（如图 1 所示）.

2）多线程进位优化处理. 之前大多数实现是在 Montgomery 乘法的并行实现过程中，采用循环检查进位的方法处理进位. 该方法一方面可能损失性能，另一方面由于循环轮数不固定，有可能泄露私钥的信息. 该方案专门设计了进位保留（carry-holding）与进位预测（carry-predicting）技术，实现进位处理的常量时间化，大大提升了多线程进位的处理效率，也降低了可能带来的安全风险.

3）完整的 RSA 实现. 该方案在模幂运算等核心算法实现的基础上，复用模幂运算的计算资源，完成了中国剩余定理计算的并行化，实现了完整的 RSA 公私钥操作，可直接为实际应用提供加速密码服务.

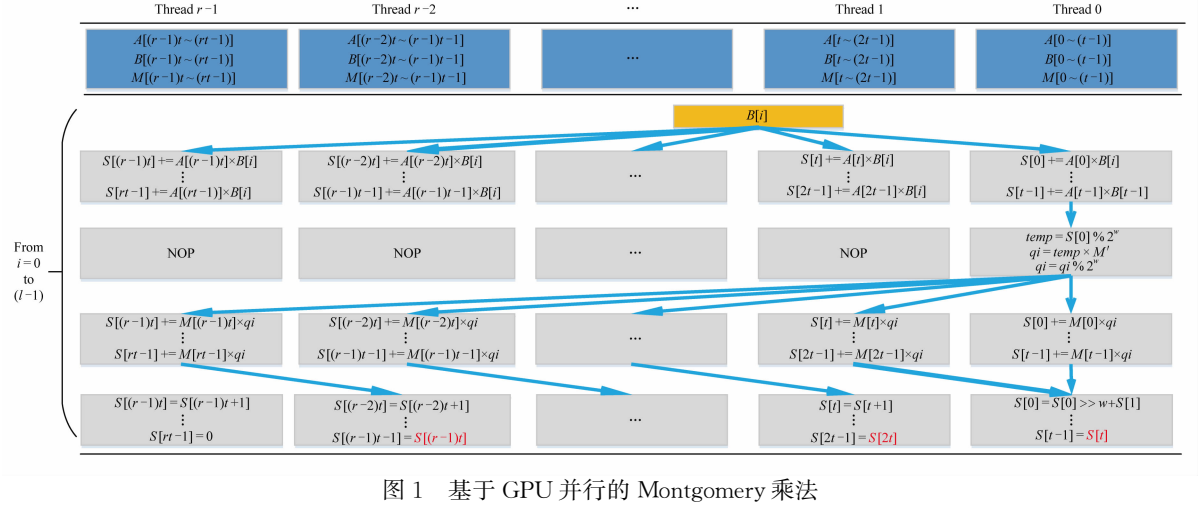

图 1 基于 GPU 并行的 Montgomery 乘法

1.3.2 ECC 算法典型实现

文献[10]使用 GPU 定点数性能，基于曲线 NIST P-256 曲线，设计实现了一整套完整的签名验签网络服务原型.

考虑到 ECC 计算所需计算量和变量均较少，该方案采用“完全串行计算”模式以获得尽可能高的性能，该方案在 GTX 780Ti 上，签名和验签速度分别达到每秒 893 万次和每秒 92 万次，是目前最快的 ECC 数字签名网络服务器. 该方案采用的是以下几种具体优化方法：

1) 有限域算法优化. 直接调用 GPU 汇编指令集，设计实现了高效的模乘、模平方算法，充分使用 GPU 上特殊的乘法指令，大大降低了加法指令的使用数量，数量从 $O(n^2)$ 降低至 $O(n)$.

2) 椭圆曲线算术优化. 为更高效地实现点加算法，该方案通过优化算法步骤与使用变量，有效地减少了寄存器使用数量，增加了并行度，提高了整体的并行计算性能. 同时，考虑到在 GPU 平台有着充裕的显存资源，专门设计了一种固定点乘预计算技术，大大提升了 ECC 数字签名的速率.

3) GPU-CPU 有效协同. GPU 并不适合所有运算，比如同时求逆运算（simultaneous inversion）[28]. 该方案将整体的 ECC 算法进行合理的划分，将不适合 GPU 的计算组件交由 CPU 实现，并尽量降低 CPU 和 GPU 之间的交互成本和数据传输，实现两者的高效协作.

4) GPU 化整为零. 为了降低计算延迟，减少达到峰值所需要的并发请求量，本方案将 GPU 划分为多个单元，实现 GPU 计算、数据传输和网络处理的流水线化，大大提升了 GPU 的计算效率，并极大地降低了数字签名网络服务的运算时延，整体服务架构如图 2 所示：

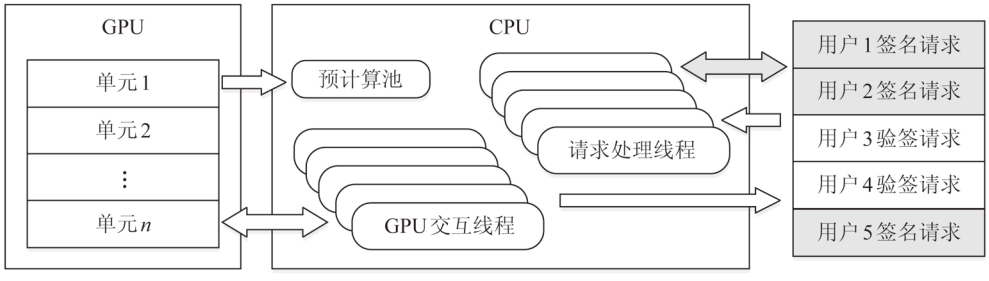

图 2 CPU-GPU 协同的 ECC 数字签名服务架构

2 发展趋势

随着密码算法和并行计算平台的不断演进和发展，近年来并行计算平台的密码算法实现呈现了新的发展趋势.

2.1 并行平台下的新型密码算法实现

随着 2013 年爱德华·斯诺登“棱镜门”事件的曝光，人们对于 NIST 发布使用的密码算法和协议开始产生了质疑，由此密码算法的研究呈现了新的趋势. 研究人员的密码算法实现研究方向正从 NIST 发布的 RSA，DSA，ECDSA 等主流算法中逐步转向新型的 ECC 算法以及更为前沿的后量子算法.

2.1.1 新型椭圆曲线算法

用于密码学的椭圆曲线主要包括短 Weierstrass，Montgomery，Twisted Edwards 等. 其中，短 Weierstrass 是目前最为主流的椭圆曲线，包括美国 ECDSA，ECDH 和中国的 SM2 算法都采用该类曲线. 目前最为常用的椭圆曲线是由 NIST 公布的 Curve P 曲线，然而在 2013 年爱德华·斯诺登指出 NIST 标准中使用的 Dual_EC_DRBG 算法存在后门之后，人们对于 NIST 以及其推出的 Curve P 曲线的安全性开始产生了质疑.

于是，近年来众多原来除了学术界无人问津的新型 ECC 密码算法实现开始获得广泛关注，有的甚至开始被大规模的使用. 比如，密钥交换算法 X25519/448（RFC 7748[29]）和数字签名算法 Ed25519/448（RFC 8032[30]）已经被 IETF 推荐使用，众多开源项目开始将原有的算法替换为 X25519/Ed25519，OpenSSL 1.1.0 版本后也加入了这些算法，并且最新版本的 OpenSSH 等将其作为默认的密码算法. 同时，面对人们的质疑，NIST 也开始考虑添加新型 ECC 算法，并于 2015 年开

始征集下一代 ECC 算法标准. 2017 年,NIST 宣布 Curve25519 和 Curve448 将会加入到 Special Publication 800-186 中. 2018 年,传输层安全协议 TLS 1. 3(RFC 8446[31])正式发布,增加 X25519 与 X448 作为密钥协商算法,增加 Ed25519 与 Ed448 作为签名验签算法.

X25519/448 所使用的椭圆曲线 Curve25519/448 为 Montgomery 曲线. Curve25519 由著名的密码学家 Bernstein 于 2006 年提出,在 2013 年开始被广泛运用. Curve25519 由于算法各参数的选取很明确,不存在潜在后门,且性能明显优于 NIST P-256 曲线,已逐渐成为 NIST P-256 曲线的替代品,被广泛应用于各种开源库和应用. Ed25519/448 使用的曲线则是由 Curve25519/448 变换而来的 Twisted Edwards 曲线. 其他目前还受到研究者普遍关注的椭圆曲线还包括 Montgomery 曲线 Curve41417[32] 和 Twisted Edwards 曲线 Four$\mathbb{Q}$[33]等.

这些算法在嵌入式软件、FPGA 上的实现得到了广泛的关注,但是这些新型的椭圆曲线算法的并行计算平台实现还处于方兴未艾的态势,针对 Twisted Edwards 曲线的文献包括文献[7,12],针对 Montgomery 曲线的文献包括文献[34-35]. 其中文献[35]中实现的 Curve25519 在同一平台下是文献[34]性能的 3 倍.

2.1.2 后量子密码算法

后量子密码算法指使用量子计算机进行攻击仍然能够保证安全的密码算法. 现有的对称密码算法和杂凑算法往往都满足该要求,但目前主流非对称密码算法所基于的三大困难问题(大整数分解问题、离散对数问题和椭圆曲线离散对数问题)都可以在多项式时间内在量子计算机上利用 Shor 算法进行破解,因此后量子算法主要指非对称密码算法. 目前后量子算法主要分为基于哈希(hash-based cryptography)、基于格(lattice-based cryptography)、基于编码(code-based cryptography)、多变量(multivariate cryptography)、超奇异椭圆曲线同源密码学(supersingular elliptic curve isogeny cryptography)等[36]. NIST 的后量子算法标准还在征集中,目前比较常用的后量子算法包括 NTRU,BLISS,New Hope,XMSS,McEliece,Rainbow 等,其中 NTRU,BLISS,New Hope 已经在 IPSec VPN 内核实现 StrongSwan 中得到了使用;XMSS 也被 IETF 所采纳.

但是,目前在 GPU 并行计算平台上实现的后量子算法还十分匮乏,主要是对基于格的 NTRU[37-40]和 Ring-LWE[41-42]的相关实现. 但即使对于这类算法,相关研究也起步较晚、尚不成熟,离实际的使用还有距离,这方面还存在很大的研究空白. 随着美国 NIST 后量子算法标准的征集,在并行计算平台上对这些算法进行评估将显得非常重要.

2.2 并行计算平台上的高速密码计算实现如何在实际场景中应用

事实上,学界大部分对于并行加速密码算法的研究还处在实验室阶段,仅针对非对称密码算法的核心部分(如 RSA 的模幂计算和 ECC 的椭圆曲线点乘算法)进行研究,离实际落地还有"最后一公里"要走,比如 RSA 的 CRT 计算部分、SM2 的外围运算在之前大部分的文献内都没有加以考虑,然而这些看似轻计算负载的部分如果没有被妥善处理,将会拖累整体算法的性能.

同时,随着并行计算平台在密码计算加速方向的广泛应用,研究者们近年来开始对其在进行密码计算过程中可能存在的安全问题展开研究. GPU,Xeon Phi 这些并行计算平台实现的密码算法虽然运行在硬件上,但是它本质还是软件实现,容易受到传统 CPU 平台上可能遭受的攻击,比如内存泄露、DMA 攻击、Cold-boot 攻击等针对内存中密钥的直接攻击以及侧信道攻击、错误注入攻击这些间接攻击的方式获取密钥. 目前已经存在了针对 GPU 的密钥攻击. 面向密钥的直接攻击方面,在 2017 年的 PPoPP-GPGPU 上,Zhu 等人[43]利用 NVIDIA GPU 上的 debug 工具等对寄存器内存储的密钥进行攻击并成功获取,打破了文献[44]的安全假设. 面向密钥的间接攻击方面,2016 年在顶级高性能系统会议 HPCA 上,Jiang 等人[45]成功实现了针对 AES 对称密码算法的计时攻击(timing attack).

若想将基于并行计算平台的密码算法加速用于实际场景,特别是金融、电子商务等敏感领域,密钥安全是必须加以严格对待的难点和问题. 然而在并行计算平台上实现的密码算法大部分仅考虑了高性能,对安全性的考虑非常少,甚至连一些

非常常见的密钥攻击手段都没有对应地考虑，比如大部分文献所使用的带调整步骤的 Montgomery 算法容易遭受文献[46]中的计时攻击，文献[22-23]实现使用的滑动窗口算法的执行时间与私钥的形式密切相关，明显易受计时攻击影响. 最近的相关文献在进行高性能实现的同时，也开始逐步加入了对常见攻击的考虑，比如文献[9]针对可能遭受的计时攻击和故障攻击(fault attack)，进行了专门的考虑和防护.

2.3 算法实现如何适应并行计算平台的发展趋势

近年来，并行计算平台本身也呈现了新的发展趋势. 一方面由于人工智能、机器学习等的迫切需求，一些并行计算平台开始整合特殊的计算器件，如 NVIDIA Tegra X1 引入的半精度运算单元，在牺牲精度的同时获得了计算性能的大幅度提升，这对于一些核心运算为低精度运算的后量子密码算法有着重大的意义；另一方面，由于自动驾驶、模式识别等方面的需求，一些低功耗、高性能的嵌入式并行计算平台纷纷面世，这些平台有望在某些领域取代传统的密码计算芯片.

然而，利用这些特殊的器件和特性进行密码算法实现并不是轻而易举的：首先，很多算法并不是为并行平台量身设计的，比如 RSA 和 ECC 的核心操作是大整数的乘法，因此大部分方案往往利用整型数实现，没有充分利用其强大的浮点数性能；其次，即便使用了平台的特性，如果无法很好驾驭，它反而会成为限制整体算法性能的瓶颈. 比如，文献[7]试图使用 GPU 上的浮点数进行非对称密码算法实现，但是它的算法设计存在缺陷，实现性能仅有之后同平台实现[12]的 1/6.

3 小　　结

基于 GPU 的高性能密码计算由于需要大规模并发请求才能达到峰值性能，且功耗相对较高，并不能替代其他传统密码算法实现方式. 但在目前云计算、大数据等领域中，它仍然有很广阔的应用前景. 因此，在并行计算平台上实现高效、安全的各类密码算法，为实际应用场景提供切实可用的密码算法服务，解决密码应用的性能瓶颈问题，具有很强的理论和现实意义，且颇具挑战.

参　考　文　献

[1] Nishikawa N, Amano H, Iwai K. Implementation of bitsliced AES encryption on CUDA-enabled GPU [C] // Proc of Int Conf on Network and System Security. Berlin: Springer, 2017: 273-287

[2] Cheng W, Zheng F, Pan W, et al. High-performance symmetric cryptography server with GPU acceleration [C] //Proc of Int Conf on Information and Communications Security. Berlin: Springer, 2017: 529-540

[3] Diffie W, Hellman M. New directions in cryptography [J]. IEEE Trans on Information Theory, 1976, 22(6): 644-654

[4] Gallagher P, Kerry C. FIPS Pub 186-4: Digital signature standard [EB/OL]. (2015-12-22) [2018-10-10]. https://nvlpubs. nist. gov/nistpubs/FIPS/NIST. FIPS. 186-4. pdf

[5] Szerwinski R, Güneysu T. Exploiting the power of GPUs for asymmetric cryptography [C] //Proc of Int Workshop on Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2008: 79-99

[6] Harrison O, Waldron J. Public key cryptography on modern graphics hardware [EB/OL]. [2018-10-10]. https://www. researchgate. net/publication/228992449 _ Public_key_cryptography_on_modern_graphics_hardware

[7] Bernstein D J, Chen T R, Cheng C M, et al. ECM on graphics cards [C] //Proc of Annual Int Conf on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2009: 483-501

[8] Bos J W. Low-latency elliptic curve scalar multiplication [J]. International Journal of Parallel Programming, 2012, 40(5): 532-550

[9] Dong J, Zheng F, Emmart N, et al. sDPF-RSA: Utilizing floating-point computing power of GPUs for massive digital signature computations [C] //Proc of 2018 IEEE Int Parallel and Distributed Processing Symp. Piscataway, NJ: IEEE, 2018: 599-609

[10] Pan W, Zheng F, Zhao Y, et al. An efficient elliptic curve cryptography signature server with GPU acceleration [J]. IEEE Trans on Information Forensics and Security, 2017, 12(1): 111-122

[11] Fleissner S. GPU-accelerated montgomery exponentiation [C] //Proc of Int Conf on Computational Science. Berlin: Springer, 2007: 213-220

[12] Bernstein D J, Chen H C, Chen M S, et al. The billion-mulmod-per-second PC [EB/OL]. [2018-10-09]. https://www. researchgate. net/publication/254892277 _ The _ billion-mulmod-per-second_PC

[13] Neves S, Araujo F. On the performance of GPU public-key cryptography [C] // Proc of the 22nd IEEE Int Conf on Application-specific Systems, Architectures and Processors. Piscataway, NJ: IEEE, 2011: 133-140

[14] Zheng F, Pan W, Lin J, et al. Exploiting the potential of GPUs for modular multiplication in ECC [C] //Proc of Int Workshop on Information Security Applications. Berlin: Springer, 2014: 295-306

[15] Moss A, Page D, Smart N P. Toward acceleration of RSA using 3D graphics hardware [C] //Proc of IMA Int Conf on Cryptography and Coding. Berlin: Springer, 2007: 364-383

[16] Antao S, Bajard J C, Sousa L. Elliptic curve point multiplication on GPUs [C] //Proc of the 21st IEEE Int Conf on Application-specific Systems Architectures and Processors. Piscataway, NJ: IEEE, 2010: 192-199

[17] Antão S, Bajard J C, Sousa L. RNS-based elliptic curve point multiplication for massive parallel architectures [J]. The Computer Journal, 2011, 55(5): 629-647

[18] Pu S, Liu J C. EAGL: An elliptic curve arithmetic GPU-based library for bilinear pairing [C] //Proc of Int Conf on Pairing-Based Cryptography. Berlin: Springer, 2013: 1-19

[19] Jeffrey A, Robinson B D. Fast GPU based modular multiplication [EB/OL]. [2018-10-10]. http://on-demand.gputechconf.com/gtc/2014/poster/pdf/P4156_montgomery_multiplication_CUDA_concurrent.pdf

[20] Harrison O, Waldron J. Efficient acceleration of asymmetric cryptography on graphics hardware [C] //Proc of Int Conf on Cryptology in Africa. Berlin: Springer, 2009: 350-367

[21] Cruz-Cortés N, Ochoa-Jiménez E, Rivera-Zamarripa L, et al. A GPU parallel implementation of the RSA private operation [C] //Proc of Latin American High Performance Computing Conf. Berlin: Springer, 2016: 188-203

[22] Jang K, Han S, Han S, et al. SSLShader: Cheap SSL acceleration with commodity processors [EB/OL]. [2018-10-10]. https://www.researchgate.net/publication/242935693_SSLShader_Cheap_SSL_Acceleration_with_Commodity_Processors?ev=auth_pub

[23] Yang Y, Guan Z, Sun H, et al. Accelerating RSA with Fine-Grained Parallelism Using GPU [M]. Berlin: Springer, 2015: 454-468

[24] Zheng F, Pan W, Lin J, et al. Exploiting the floating-point computing power of GPUs for RSA [C] //Proc of Int Conf on Information Security. Berlin: Springer, 2014: 198-215

[25] Emmart N, Weems C. Pushing the performance envelope of modular exponentiation across multiple generations of GPUs [C] //Proc of 2015 IEEE Int Parallel and Distributed Processing Symp. Piscataway, NJ: IEEE, 2015: 166-176

[26] Dong J, Zheng F, Pan W, et al. Utilizing the double-precision floating-point computing power of GPUs for RSA acceleration [EB/OL]. (2017-09-17) [2018-10-10]. https://www.hindawi.com/journals/scn/2017/3508786/

[27] Koc C K, Acar T, Kaliski B S. Analyzing and comparing Montgomery multiplication algorithms [J]. IEEE Micro, 1996, 16(3): 26-33

[28] Hankerson D, Menezes A J, Vanstone S. Guide to Elliptic Curve Cryptography [M]. Berlin: Springer Science & Business Media, 2006

[29] Langley A, Hamburg M, Turner S. Elliptic curves for security [EB/OL]. (2016-07-06) [2018-10-11]. https://datatracker.ietf.org/doc/rfc7748/

[30] Josefsson S, Liusvaara I. Edwards-curve digital signature algorithm [EB/OL]. [2018-10-11]. https://datatracker.ietf.org/doc/rfc8032/

[31] Rescorla E. The transport layer security (TLS) protocol version 1.3 [EB/OL]. (2018-08-28) [2018-10-10]. https://datatracker.ietf.org/doc/rfc8446/

[32] Bernstein D J, Chuengsatiansup C, Lange T. Curve41417: Karatsuba revisited [C] //Proc of Int Workshop on Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2014: 316-334

[33] Costello C, Longa P. FourQ [EB/OL]. (2015-08-26) [2018-10-11]. https://www.microsoft.com/en-us/research/project/fourqlib/?from=http%3A%2F%2Fresearch.microsoft.com%2Ffourqlib

[34] Mahe E, Chauvet J M. Fast GPGPU-based elliptic curve scalar multiplication [EB/OL]. [2018-10-10]. https://eprint.iacr.org/2014/198.pdf

[35] Dong J, Zheng F, Cheng J, et al. Towards high-performance X25519/448 key agreement in general purpose GPUs [C] //Proc of 2018 IEEE Conf on Communications and Network Security. Piscataway, NJ: IEEE, 2018: 1-9

[36] Bernstein D J. Introduction to Post-Quantum Cryptography [M]. Berlin: Springer, 2009: 1-14

[37] Hoffstein J, Pipher J, Silverman J H. NTRU: A ring-based public key cryptosystem [C] //Proc of Int Algorithmic Number Theory Symp. Berlin: Springer, 1998: 267-288

[38] Hermans J, Vercauteren F, Preneel B. Speed records for NTRU [C] //Proc of Cryptographers' Track at the RSA Conf. Berlin: Springer, 2010: 73-88

[39] Dai W, Doröz Y, Sunar B. Accelerating NTRU based homomorphic encryption using GPUs [C] //Proc of High Performance Extreme Computing Conf. Piscataway, NJ: IEEE, 2014: 1-6

[40] Dai W, Sunar B, Schanck J, et al. NTRU modular lattice signature scheme on CUDA GPUs [C] //Proc of Int Conf on High Performance Computing & Simulation. Piscataway, NJ: IEEE, 2016: 501-508

[41] Tan T N, Lee H. High-performance Ring-LWE cryptography scheme for biometric data security [J]. IEIE Trans on Smart Processing & Computing, 2018, 7(2): 97-106

[42] Al Badawi A, Veeravalli B, Aung K M M, et al. Accelerating subset sum and lattice based public-key cryptosystems with multi-core CPUs and GPUs [EB/OL]. [2018-12-11]. https://www.sciencedirect.com/science/article/pii/S0743731518302831

[43] Zhu Z, Kim S, Rozhanski Y, et al. Understanding the security of discrete GPUs [C] //Proc of the General Purpose GPUs. New York: ACM, 2017: 1-11

[44] Vasiliadis G, Athanasopoulos E, Polychronakis M, et al. Pixelvault: Using gpus for securing cryptographic operations [C] //Proc of the 2014 ACM SIGSAC Conf on Computer and Communications Security. New York: ACM, 2014: 1131-1142

[45] Jiang Z H, Fei Y, Kaeli D. A complete key recovery timing attack on a GPU [C] //Proc of 2016 IEEE Int Symp on High Performance Computer Architecture. Piscataway, NJ: IEEE, 2016: 394-405

[46] Kocher P C. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems [C] //Proc of Annual Int Cryptology Conf. Berlin: Springer, 1996: 104-113

郑昉昱

博士，助理研究员，主要研究方向为应用密码学、高性能计算和计算机算术.

zhengfangyu@iie.ac.cn

董建阔

博士研究生，主要研究方向为基于GPU的非对称密码算法安全高速实现.

dongjiankuo@iie.ac.cn

林璟锵

博士，研究员，主要研究方向为应用密码学、网络与系统安全.

linjingqiang@iie.ac.cn

高莉莉

博士研究生，主要方向为基于GPU的密码算法安全高速实现.

gaolili1994@iie.ac.cn