针对 AKCN-MLWE 算法的故障攻击

杨博麟[1),2)] 张 帆[3),4),8)] 赵运磊[5),6)] 张维明[7),8)] 赵新杰[3)]

[1)](浙江大学信息与电子工程学院 杭州 310027)

[2)](浙江省区块链与网络空间治理重点实验室 杭州 310027)

[3)](浙江大学计算机科学与技术学院/网络空间安全学院 杭州 310027)

[4)](移动终端安全技术浙江省工程研究中心 杭州 310027)

[5)](复旦大学计算机科学与技术学院 上海 200433)

[6)](密码科学技术国家重点实验室 北京 100878)

[7)](战略支援部队信息工程大学 郑州 450000)

[8)](郑州信大先进技术研究院 郑州 450001)

摘 要 随着量子计算技术的飞速发展以及 Shor 算法的提出,未来成型的量子计算机将轻易求解大整数分解问题以及离散对数求解问题.由于传统公钥算法如 RSA、椭圆曲线问题等其安全性均基于这些数学问题,因此该类算法面临的安全威胁也日益突出.后量子密码算法是为对抗量子计算破解而设计的一类加密算法,在近年来成为密码学研究热点.其中,基于格的后量子密码算法最为学术界广泛研究与评估.目前,密码学已经达成共识,密码算法不仅仅需要考虑算法理论安全性,同时需要考虑实现安全性,包括旁路攻击和故障攻击安全性.本文针对中国密码学会征集的第二轮后量子密码算法 AKCN-MLWE 提出了一种嵌入式环境下的故障攻击方法.AKCN-MLWE 算法是一种基于格的公钥密码算法.本文提出的故障攻击向该算法中使用的数论转换模块(NTT)中的旋转因子注入故障并影响其输出结果.在分别针对密钥生成环节和加密环节进行故障注入后,利用有效的错误输出结果可以分别进行私钥的还原以及密文的解密.同时该故障注入并不会影响生成的公私钥对在后续通信中的使用.但是在对加密环节进行故障注入后,攻击者需要使用中间人攻击方法来维持该次通信.本文也对如何在真实环境下进行故障注入进行了讨论与实用性评估.本文所提出的故障攻击方法,在算法执行过程中仅需一次故障注入即可恢复整体私钥.最后,本文同时提出一种针对性的防御方法,在不影响实现效率的情况下可有效防止该类故障攻击的生效.

关键词 故障攻击;数论转换;后量子密码;格密码;公钥密码算法

中图法分类号 TP309 **DOI 号** 10.11897/SP.J.1016.2023.01396

Fault Attack on AKCN-MLWE

YANG Bo-Lin[1),2)] ZHANG Fan[3),4),8)] ZHAO Yun-Lei[5),6)] ZHANG Wei-Ming[7),8)] ZHAO Xin-Jie[3)]

[1)](*College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou* 310027)

[2)](*Zhejiang Key Laboratory of Blockchain and Cyberspace Governance, Hangzhou* 310027)

[3)](*College of Computer Science and Technology, Institute of CyberSpace Research, Zhejiang University, Hangzhou* 310027)

[4)](*Engineering Research Center of Mobile Security of Zhejiang Province, Hangzhou* 310027)

[5)](*School of Computer Science, Fudan University, Shanghai* 200433)

[6)](*State Key Laboratory of Cryptology, P.O. Box* 5159, *Beijing* 100878)

[7)](*Information Engineering University of Strategic Support Force, Zhengzhou* 450000)

[8)](*Zhengzhou Xinda Institute of Advanced Technology, Zhengzhou* 450001)

Abstract With the development of quantum computing and the proposal of the Shor's algorithm, quantum computers will easily solve the large integer factorization problem and the discrete logarithm

收稿日期:2022-04-24;在线发布日期:2023-01-06.本课题得到国家重点研发计划(2020AAA0107700,2022YFB2701600)、国家自然科学基金(62072398,U1804263,62172435,62227805,61877011)、信息系统安全技术重点实验室基金、浙江省重点研发计划(2021C01116)、阿里巴巴-浙江大学前沿技术联合研究中心、上海科技创新行动计划技术标准项目(21DZ2200500)、河南省网络空间态势感知重点实验室重点基金以及山东省重点研发项目(2017CXG0701,2018CXGC0701)资助.**杨博麟**,博士研究生,主要研究方向为硬件安全、密码学.E-mail:yangbolin@zju.edu.cn.**张 帆**(通信作者),博士,教授,主要研究领域为硬件安全、密码算法安全性分析.E-mail:fanzhang@zju.edu.cn.**赵运磊**,博士,教授,主要研究方向为密码技术与应用.**张维明**,硕士研究生,研究方向为旁路攻击和密码芯片安全技术.**赵新杰**,博士,高级工程师,研究方向为旁路分析、故障分析、密码学中的组合分析.

problem in the future. Since the traditional public key algorithms such as RSA and elliptic curve cryptography are based on these mathematical problems, the threats to these algorithms are severe. To protect the information security, new cryptographic algorithms need to be designed and evaluated. Post-Quantum Cryptography (PQC) is a kind of algorithms designed to resist quantum computing cracking. The algorithms and implementations of PQC have been widely investigated in recent years. The U. S. National Institute of Standards and Technology (NIST) called a competitive submission in 2016. Then in 2022, the NIST proposed a finalist for the PQC schemes to be standardized. Among the PQC algorithms, the lattice-based post-quantum cryptography algorithm is the most widely studied and evaluated scheme, because of its speed of running, and size of a public key, etc. The PQC schemes not only need to be evaluated about the theoretical security under Quantum Computing, they also need to be considered for the implementation security, like the security under Side Channel Attack and Fault Attack. The implementation security indicates that the cryptographic algorithms running on the physical device need to be secure under different physical attacks. The Fault Attack means the attacker can inject a fault into the algorithms when programs are running on a physical chip. The attacker can use the faulted output to deduce the secret information that the algorithms are encrypting. This paper proposes a fault attack method under the embedded environment on AKCN-MLWE. We use the ARM Cortex-M4 as the experimental device. This scheme is a post-quantum cryptography algorithm proposed in the Round 2 competition called by the Chinese Association for Cryptologic Research (CACR). The AKCN-MLWE is also a lattice-based public key scheme. This proposed attack injects fault into the Number Theory Transform (NTT) module used in the algorithm. The NTT is commonly used for accelerating the polynomial multiplication in lattice-based algorithms. We mainly target the twiddle factors used in NTT. The twiddle factors are pre-computed and saved in the memory. With the fault injection in key generation, the informative error results can be used to recover the secret key. Meanwhile, the generated error key-pair (public key and secret key) can still be used to build a normal communication successfully. When the attacker injects the fault into encryption function, the secret message can be directly deduced from the error output. While the attacker needs to use the Man-In-The-Middle (MITM) Attack to maintain the communication, because of the Fujisaki-Okamoto Transformation used in decryption for security checks. This paper also discusses and evaluates the practicability of the fault injection in the real world. Two fault injection scenarios are evaluated and discussed with the different fault injection methods. Our attack could recover the whole secret key or message with only one fault injection during the algorithm running. At the same time, this paper proposes a specific countermeasure method to prevent this kind of fault attack without affecting the implementation efficiency.

Keywords　fault attack; number theoretical transform; post quantum cryptography; lattice-based cryptography; public key cryptography

1　引　言

随着量子计算的发展以及 Shor 算法[1]的提出，在未来利用成型的量子计算机以及对应的量子算法，将能轻易破解传统公钥密码算法如 RSA、ECC 等，网络安全与信息安全将遭到严重威胁. 在此背景下，国内外密码学界开始研究后量子密码算法，以此对抗量子算法.

后量子密码算法通过采用其他数学问题的方法来对抗量子计算的破解. 美国 NIST 于 2016 年首次开始征集后量子密码算法，第一轮中的后量子密码算法基于的数学问题包括以下五类：(1) 格；(2) 编码；(3) 多变量；(4) 哈希；(5) 超奇异同源. 其中，基

于格问题的算法由于其运行速度、公私钥长度等等综合特性优于其他问题,因此研究最为广泛.2019 年,中国密码学会举办的算法竞赛征集中,提交的公钥密码算法中多数算法同样基于格问题.本文针对的 AKCN-MLWE 算法即其中之一.

然而,不论是上述的新型后量子密码算法,还是传统公钥算法,其本质均为在传统计算机上执行的算法,执行时均以传统物理芯片作为实现载体.而在芯片上实现算法就需要考虑算法的实现安全性问题.实现安全性问题指:若算法在芯片实现时由于芯片的物理特性,造成该实现方案存在较大信息泄漏隐患,则该算法将不具备大规模部署的条件.

芯片在执行算法过程中可能产生各种与算法以及密钥相关的物理信息泄漏,包括时间、功耗、电磁辐射等等信息,这些信息统称为旁路信息,利用旁路信息进行私密信息还原的方法被称为旁路攻击(Side-Channel Attack,SCA).

另一种攻击与旁路攻击相比更为主动:即人为地在芯片执行过程中注入各类故障,使得算法执行结果出错,该错误结果可以向攻击者提供额外信息,这种方法被称为故障攻击(Fault Attack,FA).

本文针对 AKCN-MLWE 算法提出一种新型的故障攻击方法,该方法主要针对格密码中常用的加速模块:数论转换模块(Number Theoretical Transform,NTT)注入故障,可以使得 NTT 模块输出结果的复杂度降至极低.该攻击可以分别对 AKCN-MLWE 的密钥生成函数以及加密函数中的 NTT 模块注入故障,通过对函数输出的简单分析即可获取密钥生成函数所生成的私钥或直接破解被加密的信息,且注入故障后的分析求解过程可以自动化完成.

针对该攻击的故障注入部分,本文根据不同的攻击场景,提出两种可行的故障注入方法:第一种是攻击者拥有物理层面的攻击能力,可以近距离接触被攻击者的物理设备,在该算法执行过程中通过电磁脉冲对芯片进行即时故障注入;第二种是攻击者拥有网络通信层面的攻击能力,通过伪造或中间人攻击等方法使得被攻击者获取的可执行二进制文件本身存在故障,被攻击者在执行该二进制文件后即输出攻击者可以利用的错误结果,获得与第一种等效的攻击效果.

本文同时提出一种针对物理层面故障攻击的防御方法.一些常见故障防御方法往往会在空间使用率或时间效率上对算法产生负面影响,而本文所述防御方法经过真实环境下测试,不会对算法的执行效率产生影响.

本文第 2 节将简述与格密码以及故障攻击相关的已公开工作;第 3 节将给出本文所需的背景知识;第 4 节将针对 AKCN-MLWE CPA 安全等级的算法进行故障攻击分析;第 5 节将分析 CCA 安全等级下该故障攻击的可行性;第 6 节将给出两种真实环境可行的故障注入方法;第 7 节描述本文提出的防御方法;最后第 8 节对全文进行总结.

2 相关工作

在国内外学术界,对格问题的实现安全性问题研究近年来已经成为热点,其中针对格问题的旁路分析工作数量较多,然而针对格问题的故障攻击研究目前还较少.最早是 Bindel 等人于 2016 年针对基于格问题的数字签名算法如 GLP、BLISS 等进行故障攻击[2].该工作的故障模型要求在算法每一次执行过程中注入大量故障,该要求在真实环境下很难实现.Valencia 等人在 2018 年研究了各种故障注入类型对基于 R-LWE 的加密算法的影响[3],包括单比特翻转、单比特置零以及跳转指令等等.除此之外,Espitau 等人在 2016 年研究了各种基于格的签名方案对于循环中止故障(Loop Abort Faults)[4]的敏感性.在 CHES2018 上,Bruinderink 等人对两种基于格的签名方案 Dilithium 和 qTESLA 进行了差分故障分析攻击(Differential Fault Attacks)[5],他们通过时钟毛刺在实际的 ARM 平台上注入故障,在对特定中间变量注入故障后,对算法的输出结果进行差分分析,可以成功地恢复私钥的一部分.在 COSADE2019 上,Ravi 等人展现了一种针对基于格的后量子密码算法的实际的故障攻击[6],包括 NewHope、Kyber、Frodo 和 Dilithium.他们在实际的 ARM 平台上,通过电磁故障注入的方式,利用指令跳转故障模型,使 LWE 问题中的两个秘密多项式由同一个随机数种子生成,极大地降低了未知变量的复杂度,因此可以精确的恢复密钥信息.同年,AsiaCCS 会议上,Ravi 等人提出一种针对基于格的签名算法的故障攻击[7],通过电磁故障注入的方式达到跳过指令的目的,从而可以恢复部分密钥,并利用该部分密钥可对任何信息进行签名.2021 年,Pessl 等人针对 Kyber 和 Newhope 的解封装函数的解码环节提出一种新型故障攻击[8],在跳过解封装

函数的某一特定指令后，根据解封装函数输出是否被故障影响以还原私钥. 同年，Hermelink 等人对 Pessl 等人的工作进行优化[9]，借助选择密文攻击且修改了故障注入方法以及攻击位置，可以更有效地进行故障攻击.

综上所述，目前国内外已有故障工作中，故障注入类型和注入目标各有不同，而本文是首个针对 NTT 中旋转因子的故障攻击工作，也是在针对基于格的密码算法故障攻击中，首个使用篡改数组基地址此类故障模型的工作.

3　基本知识

3.1　符号表示

变量表示. 黑体小写字母如 $\boldsymbol{x}$ 表示一个向量，针对 MLWE 问题中用到的多项式向量，$\boldsymbol{x}=(\boldsymbol{x}_0,\boldsymbol{x}_1,\cdots,\boldsymbol{x}_{k-1})$ 表示该多项式向量中每个元素为一个多项式，而本文中默认每个多项式也以向量的形式表示，即 $\boldsymbol{x}_i=(x_i^0,x_i^1,\cdots,x_i^{n-1})$，其中下标表示该多项式在多项式向量 $\boldsymbol{x}$ 中的索引数，上标表示该多项式 $\boldsymbol{x}_i$ 中元素的索引数. 在 AKCN-MLWE 中，k 取 3，n 取 256，即对于一个多项式向量 $\boldsymbol{x}$，该向量包括 3 个多项式，每个多项式的维数为 256. 黑体大写字母如 $\boldsymbol{A}$ 表示一个矩阵，在 MLWE 问题中，$\boldsymbol{A}$ 可以表示一个 $k\times k$ 的多项式矩阵，该矩阵中的每个元素为一个维数为 256 的多项式. 字母 $\mathcal{B}$ 用来表示一个字节，$\mathcal{B}^n$ 表示长为 n 的字节向量. 对有理数 x，$\lfloor x\rfloor$ 表示小于等于 x 的最大整数，$\lfloor x\rceil$ 表示最接近 x 的整数. 对于一个集合 S，$s\leftarrow S$ 表示从集合 s 中随机均匀采样得到元素 s.

环. $R=\mathbb{Z}[X]/(X^n+1)$ 表示系数在整数域，模 (X^n+1) 的多项式环；$R_q=\mathbb{Z}_q[X]/(X^n+1)$ 则表示多项式系数为模 q 的整数且模 (X^n+1) 的多项式环. 在 AKCN-MLWE 中 q 取 7681.

3.2　AKCN-MLWE 算法

前文提到，格密码是目前后量子密码算法中的主流问题之一. 从数学角度考虑，格密码的困难问题有最短向量问题(Shortest Vector Problem，SVP)，最近向量问题(Closest Vector Problem，CVP)等等. 另有一类 NTRU 算法则基于名为 NTRU 的基于格的数学问题.

近年来，学习容错问题(Learning with Error，LWE)被证明对于建立算法来讲其功能更加全面. 由于 LWE 问题自身特性，基于 LWE 问题所建立的密码算法为满足安全性要求，其密钥长度会过大，对设备的存储及运算能力有较高要求. 因此，具有特定结构的 Ring-LWE(R-LWE)和 Module-LWE(M-LWE)问题被提出. 其中 M-LWE 问题平衡了 LWE 和 R-LWE 的安全性和带宽问题，拥有较好的特性. 最终 NIST 选择的 Kyber 算法[10]也是基于 M-LWE 问题. 将 M-LWE 问题抽象为数学形式可以表示为：给定多项式矩阵 $\boldsymbol{A}$，随机采样得到多项式向量 $\boldsymbol{s},\boldsymbol{e}$. 其中，$\boldsymbol{A}\in\mathbb{Z}_q^{n\times n}$，$\boldsymbol{s},\boldsymbol{e}\in\mathbb{Z}_q^{n\times 1}$. 在 LWE 问题中，要区分计算得到的 $\boldsymbol{b}=\boldsymbol{A}\boldsymbol{s}+\boldsymbol{e}$ 与随机采样得到的 $\boldsymbol{b}'$ 是困难问题，即决定性 LWE 问题；而在已知 $\boldsymbol{b},\boldsymbol{A}$ 的情况下求解 $\boldsymbol{s}$ 的值也是困难问题，即搜索性 LWE 问题.

AKCN-MLWE 算法是也基于 M-LWE 困难问题假设的密钥封装算法. 与其他主流密钥封装算法相同，AKCN-MLWE 算法同样分为选择明文攻击(Chosen Plaintext Attack，CPA)安全等级的加密算法(IND-CPA Public Key Encryption，PKE)以及选择密文攻击(Chosen Ciphertext Attack，CCA)安全等级的密钥封装算法(IND-CCA Key Encapsulation Mechanism，KEM). 其中 AKCN-MLWE-CPA-PKE 算法依赖于带噪声的非对称密钥共识算法(Asymmetric Key Consensus with Noise，AKCN)，而 AKCN-MLWE-CCA-KEM 算法由 PKE 算法借助 FO 变换(Fujisaki-Okamoto Transformation)转换而成. 密钥封装算法通常首先利用公钥算法，使得通信双方得以在非安全信道上进行协商，并得到双方认可的会话密钥，后续的数据通信将利用该会话密钥进行加密.

CPA 安全等级的 AKCN 算法示意如图 1 所示，其中 $(params,Con,Rec)$ 为 AKCN 算法定义好的参数以及子函数. 最终会话发起方与应答方达成密钥共识即 $K_1=K_2$. AKCN 算法的安全性证明请参考 AKCN-MLWE 算法提交文档以及相关论文[11].

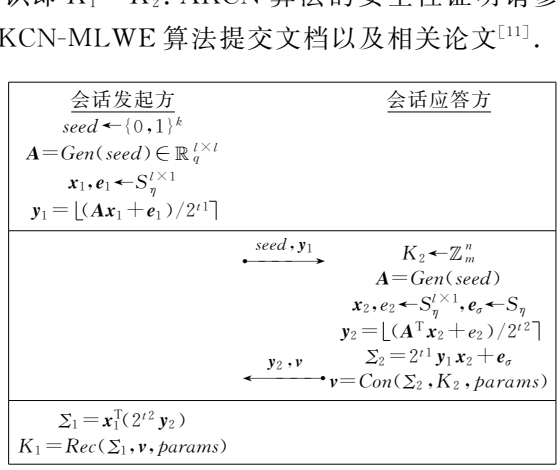

图 1　基于 AKCN 的 CPA 安全公钥加密

3.3 压缩与编码

由于基于格的密码算法中常常涉及对多项式的各项操作,在本地进行处理时用向量或数组代表多项式的系数即可.但是数据在通信过程中需要以字符串或字节串的形式发送.因此多项式在传输过程中需要进行恰当的编码.同时经算法设计者验证,公钥和密文中多项式各项系数的低位比特对正确解密的概率没有太大的影响,因此基于格的密码算法往往在生成密钥和加密之后对要传输的多项式进行压缩以及编码,在应答方收到消息后进行解码以及解压缩即可恢复多项式并进行后续解密操作.

定义压缩函数 $Compress_q(x,d)$,该函数将输入的整数 x 压缩为 x',x'的最大位数为 d,其关系为

$$x'=\lfloor(2^d/q)*x\rceil \bmod^+ 2^d \tag{1}$$

即压缩过后的 x'为$[0,2^d-1]$区间的一个整数,且该压缩函数输出与输入之间存在关系:

$$|x-x' \bmod^+ q|\leq\left\lfloor\frac{q}{2^{d+1}}\right\rceil \tag{2}$$

3.4 NTT

数论转换模块(Number Theoretical Transform,NTT)是近年来最常用的加速格密码中多项式乘法的工具之一,在 NTT 域中的多项式乘法为对应位置系数相乘,时间复杂度上要低于传统多项式乘法.因此常见的使用方法为,先将两个多项式分别用 NTT 函数转换到 NTT 域,在 NTT 域内对应位置系数相乘后将结果进行逆 NTT 函数操作,得到原域内多项式相乘的结果.如何有效地使用 NTT 与算法选择的参数有关.对于一个多项式:

$$\boldsymbol{a}=\sum_{i=0}^{n-1}a_i x^i\in R^q.$$

定义其在 NTT 域上的多项式为

$$NTT(\boldsymbol{a})=\hat{\boldsymbol{a}}=\sum_{i=0}^{n-1}\hat{a}_i x^i,\ \hat{a}_i=\sum_{j=1}^{n-1}\Psi^j a_i\omega^{i\cdot j}.$$

在AKCN-MLWE中,根据参数的选择,固定$\omega=3844$,$\Psi=\sqrt{\omega}=62$.对于 AKCN-MLWE 选择的参数,可以高效地计算 NTT 和逆 NTT 而无需额外内存,然而这种实现的结果是输出系数是以特定的顺序排列在内存中.

在 AKCN-MLWE 算法标准实现代码中,NTT 的运算将分解为多个基本函数进行,在 NTT 中将该基本函数称为蝶形运算.以维度为 8 的多项式 NTT 运算为例,如图 2 所示.

图中每个椭圆形代表一个变量,有向箭头代表将该变量乘特定系数后得到后一变量,由图可以看出,每两个输入变量经由互相运算后得到两个输出变量,该运算即为一个蝶形运算.每一纵列的所有蝶形运算代表 NTT 的一轮,不同轮和不同位置所需的特定系数不同.对于维度为 n 的多项式,NTT 将有 $\log_2 n$ 轮.

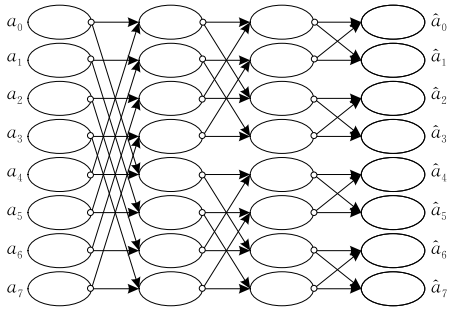

图 2 以维度为 8 为例的 NTT 运算

在 AKCN-MLWE 标准实现代码中,上述的特定系数组成了根据算法的参数选择而预计算得到的一个整数数组,该数组中的数通常被称为旋转因子(Twiddle Factors).本文进行故障攻击的对象即为该数组.

3.5 中间人攻击(Man-In-The-Middle,MITM)

在网络安全领域,近年来多种多样的网络攻击层出不穷.以常见的中间人攻击(Man-In-The-Middle,MITM)[12]为例:在网络通信场景下,合法 用户作为通信双方可通过各类信道进行数据传输通信,例如 GSM、LTE、Wi-Fi 等.在双方进行通信时,若攻击者有能力入侵特定信道,分别对通信的两方伪装身份,并进行数据的收发,则通信的两方收到的都将是攻击者提供的虚假数据,这类攻击对传输中数据的可信度、完整性等等存在不同程度的威胁.

当通信双方在该信道中使用一些身份认证协议进行验证时,可以检测并防止传统中间人攻击.但是借助密码学分析的方法,攻击者有可能对该类身份认证协议再度进行破解,并对双方分别进行身份认证,以此来继续进行中间人攻击.

本文中对 CCA 等级的密钥封装协议攻击以及网络环境下的故障注入方法都可以借助中间人攻击的概念进行.

4 针对 AKCN-MLWE CPA-PKE 的故障攻击

本文针对 AKCN-MLWE CPA-PKE 的故障攻击

将分为两类：一类针对算法的密钥生成算法 KeyGen 模块；一类针对算法的加密算法 Enc 模块. 本节将首先阐述故障模型及故障注入后 NTT 运算将产生的结果. 之后分别对密钥生成算法和加密算法进行故障攻击分析.

4.1　故障模型

在阐述故障模型前，我们需要明确蝶形运算的一个特性，如图 3 所示.

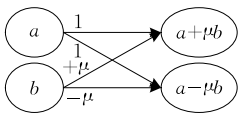

图 3　蝶形运算示意图

在一个蝶形运算中，位于 a 位置的变量，其后续运算的系数均为 1，而位于 b 位置的变量，其后续运算将与上述提到的旋转因子 μ 相乘，所有旋转因子将被预存储在存储器中，在执行时被读取到寄存器中进行运算.

本文提出的故障模型为：注入故障，使得算法在执行过程中，读取的旋转因子被篡改，且均被篡改为 0. 即在每一个蝶形运算中，最终输出结果均等于变量 a. 在此故障模型下，仍以图 2 中维数为 8 的 NTT 运算为例，经过验证，得到的多项式各系数 $\hat{a}_0 \sim \hat{a}_7$ 均等于 a_0. 即注入故障后的 NTT 运算输出多项式所有系数均为确定值，且为输入多项式的第一项系数. 在此故障模型下我们提出了分别针对 AKCN-MLWE 算法中密钥生成函数和加密函数的两种注入故障后的分析方法，在本节的后两节进行说明. 而具体的真实环境下故障注入方法将在第 6 节进行说明.

4.2　针对 CPA-PKE 密钥生成函数的故障攻击

4.2.1　目标函数说明

AKCN-MLWE 算法的 CPA 安全等级密钥生成函数如算法 1 所示.

算法 1.　AKCN-MLWE PKE KeyGen密钥生成.

输入：无

输出：(pk, sk)

1. $(\rho, \sigma) = G(d), d \leftarrow \mathcal{B}^{32}$
2. $\hat{\boldsymbol{A}} = Parse(XOF(\rho))$ //生成矩阵 $\boldsymbol{A}$(NTT 域)
3. $\boldsymbol{s}, \boldsymbol{e} = CBD_{\eta}(PRF(\sigma, N))$ //生成向量 $\boldsymbol{s}, \boldsymbol{e}$
4. $\hat{\boldsymbol{s}} = NTT(\boldsymbol{s})$ //$\boldsymbol{s}$ 的 NTT 变换，故障注入位置
5. $\boldsymbol{t} = NTT^{-1}(\hat{\boldsymbol{A}} \circ \hat{\boldsymbol{s}}) + \boldsymbol{e}$ //$\hat{\boldsymbol{A}} \circ \hat{\boldsymbol{s}}$ 为 NTT 域内乘法
6. $pk = (Encode(Compress(\boldsymbol{t}, d_t)) \| \rho)$
7. $sk = Encode(\hat{\boldsymbol{s}} \bmod q)$

其中，G, PRF, XOF 均为生成标准长度种子数据的函数，根据使用情况可选择 AES、SHAKE 等函数，具体函数的选择与使用请参考 AKCN-MLWE 算法设计文本. Parse 函数表示从特定长度数据中均匀采样得到假设 NTT 域下的多项式矩阵. CBD 为从特定长度种子数据中使用二项分布采样得到小系数多项式的函数. 其采样结果为

$$\boldsymbol{s} = (\boldsymbol{s}_0, \boldsymbol{s}_1, \boldsymbol{s}_2),$$

其中每一项 $\boldsymbol{s}_i$ 均是维度为 256，系数取值范围为 $[-2, 2] \cap \mathbb{Z}$ 的多项式. 在 AKCN-MLWE 算法的参考实现代码中，为了不使用负数，利用环的特性，将系数的取值范围定为 $[q-2, q+2]$，即 $[7679, 7683]$.

在采样得到多项式 $\boldsymbol{s}, \boldsymbol{e}$ 后，将 $\boldsymbol{s}$ 经过 NTT 变换后通过算法 1 第 5 行，生成 $\boldsymbol{t}$. 分别将 $\boldsymbol{s}, \boldsymbol{t}$ 压缩编码后输出即为私钥和公钥.

4.2.2　故障注入及分析

在算法 1 第 4 行，代表对 $\boldsymbol{s}$ 进行 NTT 变换操作. 本文提出的故障攻击方法即在此位置注入故障. 将注入故障后 NTT 的执行结果表示为 $\hat{\boldsymbol{s}}' = (\hat{\boldsymbol{s}}'_0, \hat{\boldsymbol{s}}'_1, \hat{\boldsymbol{s}}'_2)$. 由 4.1 节可知，注入故障后，$\hat{\boldsymbol{s}}'_0, \hat{\boldsymbol{s}}'_1, \hat{\boldsymbol{s}}'_2$ 分别仍为维度 256 的多项式，但每个多项式中 256 个系数均相等，其值分别为 $\boldsymbol{s}_0, \boldsymbol{s}_1, \boldsymbol{s}_2$ 的第一项系数 s_0^0, s_1^0, s_2^0，该系数取值范围为 $[-2, 2]$，在 C 代码中以 $[q-2, q+2] = [7679, 7683]$ 表示. 由于 $\hat{\boldsymbol{s}}'$ 被直接封装为私钥 sk，因此若对 $\hat{\boldsymbol{s}}'$ 进行遍历，找到正确的 $\hat{\boldsymbol{s}}'$ 即可破解私钥 sk. 在注入故障后该密钥生成函数产生的 $\hat{\boldsymbol{s}}'$ 复杂度极低，即每个多项式只有 5 种可能的取值，$\hat{\boldsymbol{s}}'$ 中有 3 个多项式，即搜索空间仅为 $5^3 = 125$.

根据算法 1 第 5 行，$\boldsymbol{t} = NTT^{-1}(\hat{\boldsymbol{A}} \circ \hat{\boldsymbol{s}}) + \boldsymbol{e}$，其中 $\boldsymbol{t}$ 在经过压缩与编码后成为公钥 pk 的一部分，而 $\hat{\boldsymbol{A}}$ 也可以通过公钥中的 ρ 重新计算得到. 因此攻击者在已知公钥的情况下可以对公钥进行解码以及解压缩操作，获知 $\hat{\boldsymbol{A}}$ 以及经过压缩后的 $\boldsymbol{t}'$. 则有下式：

$$\boldsymbol{t}' = NTT^{-1}(\hat{\boldsymbol{A}} \circ \hat{\boldsymbol{s}}) + \boldsymbol{e} + \Delta \boldsymbol{t} \tag{3}$$

其中 $\hat{\boldsymbol{s}}'$ 为待遍历的多项式，$\Delta \boldsymbol{t}$ 为压缩函数对 $\boldsymbol{t}$ 造成的误差. 由 3.3 节式(1)、(2)中对压缩函数的定义可得该误差的最大值为

$$\Delta \boldsymbol{t}_i^j \le \left\lfloor \frac{q}{2^{d_t+1}} \right\rceil = \left\lfloor \frac{7681}{2^{12}} \right\rceil = 2, \ \forall i, j.$$

因此有下式：

$$[\Delta \boldsymbol{t} + \boldsymbol{e}]_i^j = [\boldsymbol{t}' - NTT^{-1}(\hat{\boldsymbol{A}} \circ \hat{\boldsymbol{s}}')]_i^j \bmod q \le 4.$$

该式表示不等号左边所示多项式的所有系数均满足该不等式关系，该式即为恢复私钥所利用的约束条件.

经实验测试以及相应的理论证明，在此约束条件下遍历 $\hat{\boldsymbol{s}}'$ 的所有可能的 125 种取值，可以唯一确

定满足约束条件的正确的 $\hat{\boldsymbol{s}}'$. 即可以根据注入故障后得到的公钥,还原其对应生成的私钥 sk. 具体证明过程参见附录证明 1.

4.2.3 故障攻击场景描述

由于在未注入故障的算法中,$\boldsymbol{s}$ 即为随机采样得到,而故障注入后,中间变量 $\hat{\boldsymbol{s}}'$ 的各项系数仍在算法正确取值范围内,且公钥是通过中间变量 $\hat{\boldsymbol{s}}'$ 计算得到,因此注入故障后密钥生成函数输出的公钥与私钥仍为一对符合算法要求生成的公私钥对. 因此该故障不影响生成的公私钥后续使用的正确性. 当被攻击者 1 使用故障注入后生成的公钥进行加密后,被攻击者 2 使用该公钥对应的私钥仍然可以对密文正确解密,而攻击者也可对该条信息进行正确解密.

因此,针对 AKCN-MLWE PKE 密钥生成函数的故障攻击方法适用于:攻击者有能力对会话发起方或密钥分发方进行故障注入,在获取会话应答方所使用的公钥后,即可利用公钥对发起方持有的私钥进行还原. 在还原私钥后即可解密会话应答方以对应的公钥进行加密的信息.

4.3 针对 CPA-PKE 加密函数的故障攻击

4.3.1 目标函数说明

AKCN-MLWE 算法的 CPA 安全等级加密函数如算法 2 所示.

算法 2. AKCN-MLWE PKE Enc 加密函数.

输入:公钥 pk,明文 m,随机数 r

输出:密文 c

1. $\boldsymbol{t}=Decompress_q(Decode_{d_t}(pk),d_t)$
2. $\rho\leftarrow pk$ //ρ 从公钥后半部分取得
3. $\hat{\boldsymbol{A}}=Parse(XOF(\rho))$ //生成矩阵 $\boldsymbol{A}$(NTT 域)
4. $\boldsymbol{r},\boldsymbol{e}_1,e_2=CBD_\eta(PRF(\boldsymbol{r},N))$ //生成向量 $\boldsymbol{r},\boldsymbol{e}_1$,多项式 e_2
5. $\hat{\boldsymbol{r}}=NTT(\boldsymbol{r})$ //$\boldsymbol{r}$ 的 NTT 变换,故障注入位置
6. $\boldsymbol{u}=NTT^{-1}(\hat{\boldsymbol{A}}\circ\hat{\boldsymbol{r}})+\boldsymbol{e}_1$
7. $v=NTT^{-1}(NTT(\boldsymbol{t})\circ\hat{\boldsymbol{r}})+e_2+Decode(Decompress(m,1))$
8. $c=Compress(\boldsymbol{u},d_u)\|compress(v,d_v)$

其中,$\boldsymbol{r},\boldsymbol{e}_1$ 为多项式向量,与上一节中 $\boldsymbol{s},\boldsymbol{e}$ 一致. $Decode(Decompress(m,1))$ 代表将长度为 256 比特的 m 转化为维度 256 的向量,每一项系数对应 m 的一个比特. 若 m 中该比特为 0,则向量中对应系数也为 0,若 m 中该比特为 1,则向量中对应系数将记为 $q/2$. 为了书写方便,本文后续将 $Decode(Decompress(m,1))$ 简写为 $De(m)$.

4.3.2 故障注入及分析

在算法 2 第 5 行,对多项式向量 $\boldsymbol{r}$ 进行 NTT 变换操作. 本文仅对该 NTT 操作进行故障注入,但不对第 7 行的 $NTT(\boldsymbol{t})$ 操作进行故障注入. 原因将在本节最后解释. 故障注入后的 $\hat{\boldsymbol{r}}'$ 与 4.2.2 节中 $\hat{\boldsymbol{s}}'$ 相同,遍历搜索空间为 $5^3=125$,且存在关系:

$$v'=NTT^{-1}(NTT(\boldsymbol{t})\circ\hat{\boldsymbol{r}})+e_2+De(m)+\Delta v \tag{4}$$

此处与 4.2.2 节不同之处在于,由于加密函数中对多项式压缩的参数 $d_v=3<d_t=11$,因此

$$\Delta v_i^j\leq\left\lfloor\frac{q}{2^{d_v+1}}\right\rceil=\left\lfloor\frac{7681}{2^4}\right\rceil=480,\ \forall i,j,$$

即由于压缩多项式而引入的误差范围变大,但同样可以建立约束条件:

$$[\Delta v+e_2+De(m)]_i^j=v-NTT^{-1}(NTT(\boldsymbol{t})\circ\hat{\boldsymbol{r}}) \tag{5}$$

由于 $De(m)$ 的取值范围固定在 0 或 $q/2$,e_2 的取值范围为[−2,+2],因此,$v-NTT^{-1}(NTT(\boldsymbol{t})\circ\hat{\boldsymbol{r}})$ 的取值范围应在[0−480−2,0+480+2]或[q/2−480−2,q/2+480+2]之内. 同样经实验测试以及与上节所述类似的理论证明可得,在此约束条件下遍历 $\hat{\boldsymbol{r}}'$ 的所有可能取值,可以唯一确定满足约束条件的正确的 $\hat{\boldsymbol{r}}'$. 且此时根据差值落在 0 值附近还是 $q/2$ 附近即可直接还原被加密的明文 m.

前文提到不对第 7 行的 $NTT(\boldsymbol{t})$ 进行故障注入. 原因在于:本故障攻击方法在遍历 $\hat{\boldsymbol{r}}'$ 时,需要以满足等式(5)为约束条件,寻找 $\hat{\boldsymbol{r}}'$ 的正确值. 若该处 NTT 函数存在同样的故障,则 $NTT(\boldsymbol{t})$ 的各项系数也将取同一个值,此时 $NTT(\boldsymbol{t})\circ\hat{\boldsymbol{r}}$ 处计算得到的结果也为各项系数相等,可满足等式(5)的 $\hat{\boldsymbol{r}}'$ 候选值数量增加,相当于缺少一个约束条件,将无法唯一确定正确的 $\hat{\boldsymbol{r}}'$. 因此本文提到的故障攻击方法不对 $NTT(\boldsymbol{t})$ 进行故障注入.

4.3.3 故障攻击场景描述

针对 AKCN-MLWE PKE 加密函数的故障攻击方法适用于:攻击者有能力对会话应答方或加密方进行故障注入,且可以获取注入故障后的密文结果. 攻击者可利用密文对加密环节中的中间变量进行遍历,唯一确认被加密的明文 m.

5 针对 AKCN-MLWE CCA-KEM 的故障攻击

上一节中讨论了如何针对 CPA 安全等级下的 PKE 算法进行故障攻击. 前文提到,CCA-KEM 算法由 CPA-PKE 算法经过 FO 变换得到,即 CCA-KEM 算法中可直接调用 CPA-PKE 算法,经 FO 变换中的一系列包装后输出双方建立的共享会话密钥

K. 在本节中，将重点阐述在 FO 变换的 CCA-KEM 算法下该故障攻击方法是否仍然可行.

5.1　针对 CCA KEM 密钥生成函数的故障攻击

AKCN-MLWE 算法的 CCA 安全等级密钥生成函数如算法 3 所示.

算法 3.　AKCN-MLWE KEM KeyGen密钥生成.

输入：无

输出：(pk, sk)

1. $z \leftarrow \mathcal{B}^{32}$
2. $(pk, sk') = \text{AKCN\_MLWE\_PKE.KeyGen}()$
3. $sk = (sk' \| pk \| H(pk) \| z)$ //私钥为拼接而成

AKCN-MLWE 算法的 CCA 安全等级解封装函数如算法 4 所示.

算法 4.　AKCN-MLWE KEM Dec 解封装函数.

输入：密文 c，私钥 sk

输出：共享密钥 $K \in \mathcal{B}^{32}$

1. $s, pk, h = H(pk), z \leftarrow sk$
2. $m' = \text{AKCN\_MLWE\_PKE.Dec}(s, c)$
3. $(\overline{K}', r') = G(m' \| h)$
4. $c' = \text{AKCN\_MLWE\_PKE.Dec}(pk, m', r')$
5. IF $c = c'$ THEN
   返回 $K = H(\overline{K}' \| H(c))$
6. ELSE
   返回 $K = H(z \| H(c))$.

本文 3.2 节中提到，CCA-KEM 算法是由 CPA-PKE 算法经由 FO 变换得到. 由算法 1 和算法 3 对比可知，在密钥生成函数中 CCA-KEM 与 CPA-PKE 相比没有增加其他运算，仅仅是在输出的私钥字节串 sk 后拼接了公钥、公钥的哈希值以及随机字节串 z. 对攻击者来说，对 CCA-KEM 密钥生成函数攻击与 CPA-PKE 密钥生成函数攻击是相同的，得到的是注入故障后生成的私钥 sk'，且该私钥将作为正确生成的私钥被使用.

而对于解封装函数，根据算法 4，FO 转换体现在：首先将接收到的密文用 CPA-PKE 解密函数进行解密，得到 m'，之后使用 sk 中保存的公钥信息将 m' 重新加密，得到新的密文 c'，通过比较输入密文 c 和计算得到的 c'，可以验证该密文是否是利用正确公钥加密得到的未经篡改的结果，以对抗基本的选择密文攻击. 若比较结果不相等，则 FO 机制下会输出由随机字节串 z 生成的结果.

攻击者在使用 CCA-KEM 解封装函数利用恢复的 sk' 对密文进行解密时，由于 sk' 中存储的 sk 和 pk 为同时生成的一对公私钥对，因此重新加密的结果也是正确密文，在算法 4 中将满足第 5 行的条件，输出正确的共享密钥，因此解密时与 sk 中的随机数 z 的取值无关. 综上所述，针对 CPA-PKE 密钥生成算法的故障攻击方法在 CCA-KEM 密钥生成算法上仍旧可以实现.

5.2　针对 CCA KEM 密钥封装函数的故障攻击

AKCN-MLWE 算法的 CCA 安全等级密钥封装函数如算法 5 所示.

算法 5.　AKCN-MLWE KEM Enc 密钥封装.

输入：公钥 pk

输出：密文 c，共享密钥 $K \in \mathcal{B}^{32}$

1. $m \leftarrow \mathcal{B}^{32}, m \leftarrow H(m)$
2. $(\overline{K}, r) = G(m \| H(pk))$
3. $c = \text{AKCN\_MLWE\_PKE.Enc}(pk, m, r)$
4. $K = H(\overline{K} \| H(c))$.

CCA 等级下的密钥封装函数，首先随机生成数据 m 并对其进行哈希操作，得到特定长度的随机数. 之后对该随机数利用 CPA-PKE 的加密函数进行加密，得到密文 c，再利用密文 c 的哈希以及前文中生成的部分随机数生成会话密钥 K.

利用 4.3 节中针对 PKE 加密算法的故障攻击方法，在算法 5 第 3 行可以在已知密文 c 和公钥 pk 的情况下，还原信息 m. 此时攻击者可以利用算法 5 第 2 行的定义重新计算得到 $\overline{K}$，最终即可通过第 4 行计算得到共享密钥 K.

由此可得，针对 CPA-PKE 加密算法的故障攻击方法在 CCA-KEM 密钥封装算法中同样适用.

然而，在针对 CCA-KEM 密钥封装算法进行此类故障攻击后，可以发现我们相当于篡改了算法 5 中第 3 行中参数 r 的值. 以此方法生成的密文 c^* 在合法用户进行算法 4 的解封装时，由于 r 的不匹配，将无法输出正确的会话密钥 K，则该次会话建立将失败，双方可能会重新进行密钥封装，那么此次故障攻击也宣告失败.

因此，在针对 CCA-KEM 密钥封装算法进行故障攻击后，攻击者需要使用中间人攻击的方法帮助该会话正确建立：在攻击者对加密方注入故障并获取加密方生成的密文后，对该密文进行破解，还原信息 m，并获取会话密钥 K. 接着，攻击者需要用公钥 pk 以及信息 m 重新进行一次无故障的正确密钥封装，并得到正确密钥封装的结果 c. 攻击者将该结果 c 发送给解密方，则双方即使用该会话密钥 K 进行会话，攻击者则利用会话密钥 K 对双方后续的对话信息进行破解.

6 故障注入方法评估

由第 4、5 节可知，本文提出的针对 AKCN-MLWE 算法的故障攻击方法可以有效还原私钥或破解密文，该攻击方法依赖于在算法执行过程中注入了恰当的故障，即使得 NTT 调用的旋转因子数组中各项均为 0. 为说明该故障注入的可操作性，本节将列举两种在不同场景下注入故障的方法，使得算法在芯片上执行时产生前文所描述的故障类型.

6.1 故障注入位置

在真实环境下，如何通过各种物理方式注入故障是故障攻击领域最根本的问题. 近年来，学术界发表了不少通过时钟毛刺、电磁脉冲、激光等等方法进行故障注入的工作. 由于微控制器的结构简单，因此上述各类攻击更易在微控制器平台实现. 本节将以 ARM 平台为例，说明本文提出的故障注入方法的可行性.

如前文所述，本文所需的故障为强制使得进程调用的一个特定数组中所有元素均为 0. 传统电磁故障注入往往只关注故障对指令自身执行状态的影响，如中断指令或跳过指令；或是瞬时对单个数据产生影响，如比特反转等. 而以上述方法，要使得数组的所有元素改变非常困难. 因此本文提出一个新的故障注入位置，具体描述如下：

在计算机使用的汇编指令中，当一个函数调用一个具体数组时，会在函数初始化时将数组第一个元素所在的地址作为基地址读取. 如果在函数中要多次调用一个数组的元素，通常使用的方法是：先将该数组在存储器中的基地址保存到一个寄存器中，之后每次使用读取指令时，以该寄存器中所存的地址加上立即数作为索引，读取到数组中确定位置的元素. 此处以 ARM 指令为例，如算法 6 所示.

算法 6. ARM 读取数组常见方法的汇编代码表示.

```
Func(zetas); //函数开始，zetas 为被调用数组
//初始化寄存器
//函数初始化时将数组 zetas 基地址预存在 r1
1. ldr r2, [r1]
……
2. ldr r3, [r1, #2]
```

其中第 1 行表示，将 r1 中所存地址中的数据读取到寄存器 r2，即 $zetas[0]$. 之后要读取其他元素时，如第 2 行则表示对基地址以 2 为偏移量向后读取数据并保存到寄存器 r3.

据此特征，我们的目标是将故障注入在旋转因子数组的基地址中，使得以基地址为基准读取到的所有数组的数均改变. 由于微控制器的存储器中未使用的部分存在连续的数据为 0 的区域，因此可以将指向旋转因子数组的基地址篡改为指向数据为 0 的地址. 如此以来后续所有以基地址为基准读取到的数据将均为 0.

6.2 物理平台故障注入方法

前文提到，传统电磁故障攻击使用的注入方法可能不适用于本文提出的故障模型. 然而在 2019 年，Alexandre 等人证明，利用电磁脉冲故障，可以在微控制器将数据从存储器读取到寄存器的过程中，使该数据出错. 该工作利用微控制器读取数据时的操作特征实现. 文中提到，微控制器在读取数据时，首先会将数据先从存储器读取到数据缓存区，之后再通过数据通路保存到寄存器中. 在上述流程中，经过实验验证，可以利用外部电磁脉冲在特定位置、特定时间进行故障注入，达到篡改正在读取的数据的目的. 被篡改的数据位置、数据大小与电磁脉冲的参数、电磁脉冲探头的位置等相关. 该类攻击的攻击平台如图 4 所示.

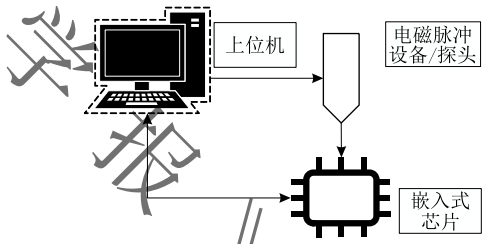

图 4 电磁故障注入示意图

在该攻击平台中，当嵌入式芯片执行到代码的具体位置时，提前设置该芯片提供触发信号到上位机，由上位机控制电磁设备发出电磁脉冲，在对整套设备进行多次调试后可以找到准确的电磁故障注入时间、空间位置. 具体实验原理及参数设置请参考文献[13].

针对本文提出的故障模型，由于基于 ARM 指令集的编译器在编译时，会将旋转因子这类数组的基地址预先设定好并保存在最终的二进制文件中. 因此利用电磁脉冲设备，调整好触发时间，在恰当位置恰当时间对从存储器中读取基地址的指令进行电磁脉冲故障注入，即可实现篡改基地址的故障目的.

我们在 STM32F4DISCOVERY 开发板上进行了攻击验证. 攻击的对象代码即 pqm4 中的 NTT 部

分. 首先利用示波器和电磁信号采集装置对开发板执行过程中产生的旁路信号进行采集，以定位要攻击的时间位置，接着利用二维云台将电磁故障注入装置在芯片范围内进行 x-y 轴扫描，寻找最佳故障注入位置. 在不断逼近下我们在真实环境下找到了合适的注入参数，并且达到了约 70%的故障注入成功几率.

6.3　网络环境故障注入方法

上节提到的物理平台电磁故障注入方法，虽然是实现安全性领域常用方法，且已有公开的论文对可行性进行证明，文献中描述其成功率也接近 100%. 但该方法对攻击设备的精度、攻击者对被攻击设备的熟悉程度等等都有较高要求. 因此本文讨论是否有其他攻击方式可以代替物理平台故障注入方法.

随着物联网领域各项技术的飞速发展，物联网设备的数量成倍增加. 在物联网设备中部署各类算法时，为了产品出厂效率可能存在不用源码编译，而是直接下载公开密码库的可执行二进制文件或动态库的情况.

在此背景下，若该公开密码库的二进制文件中存在漏洞，则所有下载该源文件并使用的物联网设备都将存在该漏洞. 而攻击者仅需设计恶意二进制代码，其难度与前文提到的在芯片特定位置特定时间注入故障相比非常低，重点是如何令被攻击者使用恶意二进制代码.

前文已提到，对同一型号的设备，旋转因子数组的基地址预先设定好并保存在二进制文件中. 因此，若攻击者利用中间人攻击方法，在网络环境下使被攻击者获取的二进制文件中该位置存在错误，则可使得被攻击者使用该二进制文件执行密钥封装算法，被攻击算法表面上在正确执行，顺利生成了公私钥并可以顺利进行后续的分发公钥并加密、私钥解密等等环节，因此该类故障较难被发现.

但是被攻击者所保存的私钥，其复杂度永远符合本文提出的故障攻击要求. 即被攻击者所产生的私钥或被加密的明文会被攻击者利用公钥或密文轻易攻破.

在此故障注入模式下需注意，将该方法应用于密钥生成函数时，因为该函数中仅存在一次 NTT 操作，因此不论是将二进制文件中的地址篡改，还是将二进制文件中旋转因子数组直接改为期望值均可达到预期结果.

但对于加密函数以及密钥封装函数，由于其中需执行两次 NTT 操作，除攻击目标位置外还包括对公钥的 NTT 操作，若对公钥的 NTT 操作存在同样故障，则 4.3.2 节中已提到，该故障攻击方法将失效. 因此，在网络环境下直接提供恶意二进制代码时，要注意区分两次 NTT 操作，以及对应的篡改方法.

7　故障防御方法设计

基于前文提出的故障攻击方法，本节同时提出一种对该攻击具有针对性的新型防御方法，可以有效抵御该类故障攻击的同时，又不会影响算法的执行效率以及代码量.

7.1　传统防御方法可行性分析

在实现安全性问题领域，针对故障攻击传统的防御方法有两种：重复计算和故障检测.

故障检测的方法往往建立在硬件设计环节，比如使用锁相环等硬件模块对时钟毛刺以及电磁脉冲等进行针对性检测. 其原理是，一旦由于此类故障在电路中产生了不合理的毛刺信号，该模块就会通过相位等参数检测到关键信号中存在毛刺并产生警告信号，中断执行. 该类方法可以从根本上防御特定的物理层面故障注入方法，因此对本文提出的物理层面攻击方法可以适用，但由于此类硬件设计模块需要在芯片设计环节进行对应的部署，具体细节在本文中将不再阐述，读者可参考文献[14].

对于重复计算的防御方法，字面意思即将关键函数重复执行两次，并对比两次执行的结果，若计算结果相同，证明在单次执行过程中不存在瞬时故障，可以输出结果；若不同，则终止执行. 该类方法可适用于使用时钟毛刺、电压毛刺、电磁脉冲等注入方法造成瞬时故障使单次计算中数据反转或跳过单个指令的场景.

针对本文提出的故障攻击方法，首先对网络环境故障注入方法进行分析. 在该注入方法中，攻击者拥有直接修改二进制文件的能力. 因此，在攻击者篡改之后，即使攻击者不在代码中直接抹去重复计算部分的代码，而仅仅篡改读取旋转因子时所用地址，则在每次执行时调用旋转因子所使用的地址均为错误地址，不论计算次数，每次执行时的结果将均存在故障，因此该防御方法无法对网络环境下的故障注入生效.

对于物理环境下的电磁故障注入，由于本文提出的方法是在将地址从存储器读取到寄存器过程中进行故障攻击篡改，并未直接修改存储器中的数值，因此使用重复计算的方法可以对本文提出的方法进行防御. 然而对于基于格的密码算法，NTT 与哈希是整个算法执行时间中占比最大的两个模块，因此

若要重复计算 NTT 模块，则对于此类密码算法将极大增加算法执行时间，对算法执行效率造成一定影响.由于密码竞赛中一项重点评估参数即执行效率，因此本防御方法并不完全适用.

7.2 防御方法实现对象

同样以 ARM 平台为例.在 ARM 平台实现 AKCN-MLWE 算法时，可以参考开源的 pqm4 算法库针对 NIST 第三轮中算法的优化实现方法，将 NTT 函数用 ARM 汇编语言进行编写.具体文件请参考 pqm4① 公开库[15].这里将与本文相关的部分列举如下：

算法 7. pqm4 库中 ARM 汇编代码实现 NTT.

```
1. twiddle_ptr .req r1
//函数初始化时将数组 twiddle 基地址预存 r1
2. twiddle .req r10 //将旋转因子读取到 r10
3. load poly, poly0, poly1, poly2, poly3,
#0, #distance/4, #2 * distance/4, #3 * distance/4
//将输入数据读取到各 poly 变量中
4. ldrh twiddle, [twiddle_ptr] //将数组 twiddle 的首个元素读取到 twiddle 寄存器中
5. two_doublebutterfly b, b, poly0, poly4,
poly1, poly5, twiddle, tmp, tmp2, q, qinv
//将旋转因子及输入数据通过蝶形函数运算
```

其中 1、2 行代表在 NTT 函数初始化时，将旋转因子数组(这里用 twiddle 表示)的基地址保存在 r1，即变量 twiddle_ptr.再设定 r10 为一个变量 twiddle，即每次要更新蝶形运算使用的旋转因子时，将根据 r1 中的地址向后读取新的旋转因子并更新变量 twiddle.第 3 行表示将一个蝶形运算的输入读取到设定好的 poly 系列变量中.第 4 行即为执行第一个蝶形运算前，将该蝶形运算所需的旋转因子读取到变量 twiddle 中.第 5 行表示将所有更新过的系数输入预先定义好的函数 two_doublebutterfly 中，即执行蝶形运算操作.

由该代码可见，本文提出的故障攻击在 pqm4 所优化的 NTT 汇编代码实现中同样适用，因为在该实现中同样将基地址保存并多次调用.而本文所提出的防御方法主要修改第 4 行的实现方法.

7.3 物理层面故障防御方法具体实现

在编写代码时，有一种方式称为硬编码(Hard Coding)，通常也可被称为写死.即在代码中需要用到某些特定参数的时候，在代码中直接明确用该参数的具体数值，而不是通过其他方式获取，比如读取或重新计算.

本文提出的防御方法即基于硬编码方法.在诸如第 4 行从基地址读取旋转因子的操作时，我们将基地址换为该位置所需的旋转因子的具体数值.该数值以“#x”的形式在读取(ldr)指令中时，代表直接将该常数读取到变量“*twiddle*”中，例如：

```
ldr twiddle, #2571
```

上述汇编代码表示，直接将该次蝶形运算所需的具体数值 2571 赋值给变量 *twiddle*，而非从旋转因子的基地址 *twiddle_ptr* 处读取对应的数值.如此修改后，每次蝶形运算需要新的旋转因子时，都使用上述语句对变量 *twiddle* 进行更新.代码如此实现时，若攻击者希望篡改所有蝶形运算所使用的旋转因子，则需要对每一条类似上述形式的汇编代码进行攻击.在真实环境进行电磁故障注入时，仅仅攻击一个位置即读取基地址的操作是可行的，前文已经证明.然而在算法单次执行过程中要做到如此多次精准故障注入是不现实的，这受限于电磁脉冲产生装置的时延、频率等参数.因此，通过该方法攻击者无法有效的将一次 NTT 函数中所有蝶形运算中的旋转因子均改为 0，即无法达到本文所提出的故障攻击的前提，因此防御方法有效.

该防御方法仅仅是改变了旋转因子更新的方法，即从存储器中根据地址读取变为了代码直接赋值，而其他运算部分的代码均不需要修改，因此该防御方法并不会对算法实现效率造成影响.

然而，并不是 NTT 中所有该操作都适合硬编码，由图 2 可知，NTT 分为多轮运算，而根据 NTT 的运算特性，在 NTT 第一轮，所需的旋转因子均相等，第二轮之后所需的旋转因子数量依次成倍增加.且由于在基于格的算法中，多项式的维度往往较大，如 256，因此 NTT 在每一轮中往往用循环函数来实现，即对于旋转因子相同的蝶形运算，仅需更新一次旋转因子，之后的蝶形运算操作利用循环函数进行.这导致的结果是，在前几轮的循环中可以将旋转因子“写死”，因为同一个旋转因子可以在多个蝶形运算中重复使用.而后几轮若要“写死”，则有大量的蝶形运算开始前需要使用“写死”方法对旋转因子变量进行更新.这将导致无法使用循环函数对重复的代码进行优化，会成倍增加其代码量.在资源不受限的情况下全部“写死”可行，然而若将代码部署在资源受限的物联网设备中，则会对存储空间等资源造成影响.

7.4 非汇编代码的防御方法实现

若不在汇编代码层面进行修改，在 C 代码中同样可采用一种简单方法对物理层面的故障攻击进行

① https://github.com/mupq/pqm4

防御.

由前文可知，NTT 函数可以分 $\log_2 n$ 层，每层所需的旋转因子具体取值不同，个数也不相同. 因此可以考虑以层为单位，将各层所需的旋转因子保存在不同的数组中. 如此以来，每一层的旋转因子其分配到的基地址各不相同. 由于当前背景下硬件条件的限制，物理层面的电磁故障在一次算法执行进程中很难精准注入多次. 因此，在一次进程中无法将所有数组的基地址全部改变. 在此条件下，本文提出的故障攻击方法同样可被该方法防御.

8　总结与展望

本文基于中国密码学会征集的公钥算法第二轮候选算法：AKCN-MLWE，针对其使用的 NTT 模块进行了完整的故障攻击的研究. 首先提出适用于该算法以及 NTT 的故障模型，后针对该故障模型产生的结果进行故障分析，该分析方法可以获取在密钥生成函数中注入故障后生成的私钥，或还原在加密函数中被加密的明文.

本文同时给出了两种符合该故障模型的故障注入方法，分别基于物理层面和网络层面，分别面向不同的攻击场景. 在物理场景下，虽然故障注入难度大要求高，但该方法是目前实现安全性领域中较为常用的方法，因此仍需重视该故障敏感位置，并提供对应防御. 因此本文最后给出了可抵抗物理层面故障攻击的防御方法，且该方法对算法实现效率无较大影响.

对于本文提出的网络层面攻击方法，该方法对于攻击者的攻击能力要求较低. 对于传统中间人攻击进行防御的方法如增加校验码等等方法理论上仍有防御效果.

对于本文的可扩展工作有以下几点：本文提出的攻击方法针对 NTT 模块，而 NTT 模块为基于格算法的常用模块，因此，在其他使用 NTT 模块的算法上可以考虑迁移该攻击的可能性. 此外，可以考虑在各优化实现以及不同软硬件平台实现基于格的算法时，是否仍存在该故障敏感点.

参　考　文　献

[1] Shor P W. Algorithms for quantum computation: Discrete logarithms and factoring//Proceedings of the 35th Annual Symposium on Foundations of Computer Science. Santa Fe, USA, 1994: 124-134

[2] Bindel N, Buchmann J, Krämer J. Lattice-based signature schemes and their sensitivity to fault attacks//Proceedings of the 2016 Workshop on Fault Diagnosis and Tolerance in Cryptography. Santa Barbara, USA, 2016: 63-77

[3] Valencia F, Oder T, Güneysu T, et al. Exploring the vulnerability of R-LWE encryption to fault attacks//Proceedings of the 5th Workshop on Cryptography and Security in Computing Systems. Manchester, UK, 2018: 7-12

[4] Espitau T, Fouque P A, Gérard B, et al. Loop-abort faults on lattice-based fiat-shamir and hash-and-sign signatures//Selected Areas in Cryptography (SAC 2016)-23rd International Conference Revised Selected Papers. St. John's, NL, Canada, 2016: 140-158

[5] Bruinderink L G, Pessl P. Differential fault attacks on deterministic lattice signatures. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2018, (3): 21-43

[6] Ravi P, Roy D B, Bhasin S, et al. Number "not used" once-practical fault attack on pqm4 implementations of NIST candidates//Proceedings of the 10th International Workshop on Constructive Side-Channel Analysis and Secure Design (COSADE 2019). Darmstadt, Germany, 2019: 232-250

[7] Ravi P, Jhanwar M P, Howe J, et al. Exploiting determinism in lattice-based signatures: Practical fault attacks on pqm4 implementations of NIST candidates//Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security. Auckland, New Zealand, 2019: 427-440

[8] Pessl P, Prokop L. Fault attacks on CCA-secure lattice KEMs. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2021, (2): 37-60

[9] Hermelink J, Pessl P, Pöppelmann T. Fault-enabled chosen-ciphertext attacks on Kyber//Proceedings of the International Conference on Cryptology in India. Jaipur, India, 2021: 311-334

[10] Bos J, Ducas L, Kiltz E, et al. CRYSTALS-Kyber: A CCA-secure module-lattice-based KEM//Proceedings of the 2018 IEEE European Symposium on Security and Privacy (EuroS&P). London, UK, 2018: 353-367

[11] Jin Z, Zhao Y. Generic and practical key establishment from lattice//Proceedings of the International Conference on Applied Cryptography and Network Security. Bogota, Colombia, 2019: 302-322

[12] Conti M, Dragoni N, Lesyk V. A survey of man in the middle attacks. IEEE Communications Surveys & Tutorials, 2016, 18(3): 2027-2051

[13] Menu A, Bhasin S, Dutertre J M, et al. Precise spatio-temporal electromagnetic fault injections on data transfers//Proceedings of the 2019 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC). Atlanta, USA, 2019: 1-8

[14] Breier J, Bhasin S, He W. An electromagnetic fault injection sensor using Hogge phase-detector//Proceedings of the 2017 18th International Symposium on Quality Electronic Design (ISQED). Santa Clara, USA, 2017: 307-312

[15] PQM4: Post-quantum crypto library for the ARM Cortex-M4. https://github.com/mupq/pqm4

附 录.

证明 1. 满足下式的 $\hat{\boldsymbol{s}}'$ 有且仅有一种可能取值：

$$[\Delta \boldsymbol{t}+e]_i^j=[\boldsymbol{t}'-NTT^{-1}(\hat{\boldsymbol{A}}\circ\hat{\boldsymbol{s}}')]_i^j \bmod q\leqslant 4,$$

其中，$\hat{\boldsymbol{s}}'=(\hat{\boldsymbol{s}}_0',\hat{\boldsymbol{s}}_1',\hat{\boldsymbol{s}}_2')$，$\hat{\boldsymbol{s}}_0'$，$\hat{\boldsymbol{s}}_1'$，$\hat{\boldsymbol{s}}_2'$ 分别为维度 256 的多项式，且由于已经注入故障，因此单个多项式中 256 个系数均相等，其值分别为 NTT 运算之前 $\boldsymbol{s}_0$，$\boldsymbol{s}_1$，$\boldsymbol{s}_2$ 的第一项系数 s_0^0，s_1^0，s_2^0，该系数取值范围为[−2,2]；$\boldsymbol{t}'$ 以及 $\hat{\boldsymbol{A}}$ 可对公钥进行解压缩和解码后得到.

证明. 以反证法证明.

设 $\hat{\boldsymbol{s}}'$ 存在两种可能的取值：$\widehat{s\boldsymbol{r}'}$，$\widehat{s\boldsymbol{w}'}$ 均满足

$$[\boldsymbol{t}'-NTT^{-1}(\hat{\boldsymbol{A}}\circ\hat{\boldsymbol{s}}')]_i^j \bmod q\leqslant 4.$$

在 M-LWE 中，设 $\boldsymbol{y}=\hat{\boldsymbol{A}}\circ\hat{\boldsymbol{s}}'$，则有

$$\boldsymbol{y}_0=\hat{\boldsymbol{A}}_0\circ\hat{\boldsymbol{s}}_0'+\hat{\boldsymbol{A}}_1\circ\hat{\boldsymbol{s}}_1'+\hat{\boldsymbol{A}}_2\circ\hat{\boldsymbol{s}}_2',$$

$$\boldsymbol{y}_1=\hat{\boldsymbol{A}}_3\circ\hat{\boldsymbol{s}}_0'+\hat{\boldsymbol{A}}_4\circ\hat{\boldsymbol{s}}_1'+\hat{\boldsymbol{A}}_5\circ\hat{\boldsymbol{s}}_2',$$

$$\boldsymbol{y}_2=\hat{\boldsymbol{A}}_6\circ\hat{\boldsymbol{s}}_0'+\hat{\boldsymbol{A}}_7\circ\hat{\boldsymbol{s}}_1'+\hat{\boldsymbol{A}}_8\circ\hat{\boldsymbol{s}}_2'.$$

以上式中多项式 $\boldsymbol{y}_0$ 为例，其任意项 $m\in[0,255]$ 系数，有以下关系：

$$y_0^m=\hat{A}_0^m*\hat{s}_0^{m'}+\hat{A}_1^m*\hat{s}_1^{m'}+\hat{A}_2^m*\hat{s}_2^{m'}.$$

即 $\boldsymbol{y}$ 的每一项均与 $\hat{\boldsymbol{s}}_0'$，$\hat{\boldsymbol{s}}_1'$，$\hat{\boldsymbol{s}}_2'$ 相关.

因此，若存在两种可能的取值：$\widehat{s\boldsymbol{r}'}$，$\widehat{s\boldsymbol{w}'}$ 均满足该式，则对于所有 $j\in[0,255]$ 同时满足如下两式：

$$y_0^m=\hat{A}_0^m*\widehat{sr}_0^{m'}+\hat{A}_1^m*\widehat{sr}_1^{m'}+\hat{A}_2^m*\widehat{sr}_2^{m'},$$

$$y_0^m=\hat{A}_0^m*\widehat{sw}_0^{m'}+\hat{A}_1^m*\widehat{sw}_1^{m'}+\hat{A}_2^m*\widehat{sw}_2^{m'}.$$

则对每个 m 的取值，将两式相减，均有

$$\hat{A}_0^m*(\widehat{sr}_0^{m'}-\widehat{sw}_0^{m'})+\hat{A}_1^m*(\widehat{sr}_1^{m'}-\widehat{sw}_1^{m'})+\hat{A}_2^m*(\widehat{sr}_2^{m'}-\widehat{sw}_2^{m'})=0.$$

对于上式，由于 $\boldsymbol{A}$ 中所有系数均为随机采样所得，服从[0,7681]上整数的均匀分布，而 $\hat{\boldsymbol{s}}_0'$，$\hat{\boldsymbol{s}}_1'$，$\hat{\boldsymbol{s}}_2'$ 单个多项式中 256 个系数均相等. 即当 $\hat{A}_x^m$ 中 x 变化时，$\widehat{sr}_0^{m'}-\widehat{sw}_0^{m'}$ 其取值不变.

易得，只有 $\widehat{s\boldsymbol{r}'}=\widehat{s\boldsymbol{w}'}$ 时，对于所有 $m\in[0,255]$ 均可满足上述条件. 证得仅存在一种 $\hat{\boldsymbol{s}}'$ 可能的取值满足所有条件. 证毕.

YANG Bo-Lin, Ph. D. candidate. His research interests include hardware security and cryptography.

ZHANG Fan, Ph. D., professor. His main research interests include hardware security and cryptography security analysis.

ZHAO Yun-Lei, Ph. D., distinguished professor. His main research interests include technology and applications of cryptography.

ZHANG Wei-Ming, M. S. candidate. His research interests include side channel attack and cryptographic chip security technology.

ZHAO Xin-Jie, Ph. D., senior engineer. His current research interests include side channel analysis, fault analysis, and combined analysis in cryptography.

Background

This paper proposes a novel fault attack on the lattice-based AKCN-MLWE algorithm. This algorithm was proposed for the public key cryptography competition called by CACR (Chinese Association for Cryptologic Research). The AKCN-MLWE algorithm is also one of the lattice-based post-quantum cryptographies (PQC).

The PQC schemes not only need to be evaluated about the theoretical security under Quantum Computing, they also need to be considered for the implementation security, like the security under Side Channel Attack and Fault Attack. The implementation security indicates that the cryptographic algorithms running on the physical device need to be secure under different physical attacks. The Fault Attack means the attacker can inject a fault into the algorithms when programs are running on a physical chip. The attacker can use the faulted output to deduce the secret information that the algorithms are encrypting.

This paper mainly talks about the implementation security of novel cryptographic algorithms like post-quantum cryptography.

There have been plenty of works about the side-channel attack on PQC, while the fault attack on PQC is not widely discussed. We list some fault attacks on other lattice-based algorithms in related work. To our best knowledge, few papers discussed the fault attack on AKCN-MLWE before. This work proposes a new fault model and fault location against lattice-based algorithms, especially on AKCN-MLWE. The analysis after the fault injection is also novel and easy to implement. We also give two different fault injection methods based on different attack scenarios.

According to the side channel attacks and fault attacks like this work, the designers of PQC will concern about enhancing the implementation security in the standardization process.