基于线性判别分析的模幂掩码模板攻击方法

韩绪仓[1,2],陈波涛[3],曹伟琼[1,2],陈 华[1,2],李昊远[1,2]
(1. 中国科学院软件研究所可信计算与信息保障实验室,北京 100190;2. 中国科学院大学,北京 100049;
3. 北京中电华大电子设计有限责任公司,北京 102209)

摘 要: 掩码在模幂安全实现中被广泛采用,其抵抗侧信道分析的能力已被充分证明. 本文发现模乘运算中读操作数的功耗将泄露操作数的地址,进而提出了一种基于线性判别分析的模板攻击方法,可对模幂掩码实现进行攻击. 相比以往基于操作数的泄露,读操作数的功耗泄露将不受掩码的影响,对常见的带掩码防护的模幂实现仍有效. 本文提出的方法首先将测试向量泄露检测技术应用于泄露特征提取,降低了无关点对攻击的影响;然后将线性判别分析扩展用于对曲线的分类和降维,提升了曲线的类可分离性. 最后,本文以硬件模幂掩码实现为实验对象,通过实验验证了基于读取操作数的泄露分布在整个模乘运算中,且对不同类型模乘分类的准确率可达到99.98%.

关键词: 指数掩码;模幂掩码;模板攻击;线性判别分析;泄露检测;操作数读取
基金项目: 国家自然科学基金(No.62172395)
中图分类号: TN918;TP309 **文献标识码:** A **文章编号:** 0372-2112(2023)11-3024-09
电子学报 URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20230419

Linear Discriminant Analysis-Based Template Attack for Masked Implementation of Modular Exponentiation

HAN Xu-cang[1,2], CHEN Bo-tao[3], CAO Wei-qiong[1,2], CHEN Hua[1,2], LI Hao-yuan[1,2]
(1. *Trusted Computing and Information Assurance Laboratory*, *Institute of Software*, *Chinese Academy of Sciences*, *Beijing* 100190, *China*;
2. *University of Chinese Academy of Sciences*, *Beijing* 100049, *China*;
3. *CEC Huada Electronic Design Co.*, *Ltd.*, *Beijing* 102209, *China*)

Abstract: Masking is widely used in secure implementations of modular exponentiation, and its ability of side-channel resilient has been well-demonstrated. During the modular multiplication in modular exponentiation, we discovered that there are several fetch operations, and variations in the power consumption, which revealed the address of the operands, and then proposed a template attack based on linear discriminant analysis aiming at this vulnerability. In contrast to operand-based leakage, fetch-based leakage is not affected by mask and thus can be effective in attacking masking-based modular exponentiation. In our analysis, we extended testing vector leakage detection to the extraction of leaked features, which reduced the influence of irrelevant points. Second, linear discriminant analysis was utilized to trace classification and reduced the dimensionality of traces, which improved the ability of trace separability. Finally, an attack was conducted on a hardware implementation of masking-based modular exponentiation. Results showed that fetch-based leakage was distributed in the entire modular multiplication operation, and the correct ratio of modulo multiplication identification is up to 99.98%.

Key words: exponential masking; masked implementation of modular exponentiation; template attack; linear discriminant analysis; testing vector leakage detection; operand loading
Foundation Item(s): National Natural Science Foundation of China (No.62172395)

1 引言

模幂是RSA算法中的关键运算,其实现易受侧信道分析的威胁. 掩码是在模幂安全实现中被广泛采用的防护措施[1],其消除了中间数据与侧信道信息之间的关联. 其中指数掩码使得每条曲线对应的密钥比特不同,将攻击限制在单条曲线中[2];底数掩码对输入消息进行随机化,使得与选择输入相关的攻击[3]均不可行.

模幂在密钥比特的控制下被转化为一系列模乘运

收稿日期:2023-05-10;修回日期:2023-08-30;责任编辑:覃怀银

算,而侧信道分析则利用模乘之间的区别恢复密钥. 对于经典的平方-乘模幂算法实现,攻击的关键是区分模平方和模乘两种操作,如利用汉明重分布之间的区别[4],或检测两个模乘是否共用同一个操作数[5],或基于平方对称性的单条曲线攻击[6]. 当模平方和模乘基于相同软硬件模块实现,并且模乘的操作数被掩码防护随机化,那么二者之间的区别将极小,给侧信道分析带来了极大挑战.

目前,水平侧信道分析和基于深度学习的侧信道分析是两种针对模幂掩码实现的分析方法. 水平相关性分析[7]为单条曲线分析方法,可视为对简单能耗分析[8]的拓展,对模幂掩码同样适用. 该方法通过计算模乘内中间数据与功耗曲线之间的相关性进行攻击,还可被进一步扩展为水平侧信道分析[9]、水平碰撞分析[10]、在线模板攻击[11,12]和基于聚类的水平侧信道分析[13]等多种方法. 而基于深度学习的侧信道分析最初是用在对称算法中[14],近期也被应用于公钥密码算法的分析中[15],并对一系列的RSA模幂实现进行了攻击[16~19].

水平侧信道分析属于单条曲线攻击,其攻击效果易受噪声等的影响;而基于深度学习的侧信道分析属于建模类攻击,存在着如训练数据量大,模型可解释性低等问题. 相关研究主要利用了模乘运算中与操作数相关的泄露,此类泄露可通过掩码进行防护,极大的提升了相关攻击的难度. 针对这一问题,本文发现模乘运算中读操作数的功耗存在泄露,并将线性判别分析(Linear Discriminant Analysis,LDA)[20]与模板攻击相结合,提出了一种针对模幂掩码实现的新型模板攻击方法. 实验结果表明,本文攻击方法针对模乘分类的准确率可达到99.98%,能够恢复出模幂掩码实现的密钥.

2　基础知识

2.1　线性判别分析

LDA通过确定一组投影方向,基于线性变换将数据从高维空间中的投影到低维空间中,使得投影后的数据具有最佳的可分离性. 与主成分分析[20]等无监督类学习方法相比,LDA属于监督类学习,需要使用已知类别的数据进行学习和训练.

对于数据集 $D=\left\{\left(\boldsymbol{x}_1,y_1\right),\left(\boldsymbol{x}_2,y_2\right),\cdots,\left(\boldsymbol{x}_l,y_l\right)\right\}$,其中,$\boldsymbol{x}_i$ 为高维待分析数据,y_i 为对应的类别. 按照对应类别,D 可分为不同类:

$$D_j=\left\{\left(\boldsymbol{x}_i,y_i\right)\in D|y_i=j\right\},\ j=1,2,\cdots,K$$

定义类内距离 $\boldsymbol{S}_0$ 和类间距离 $\boldsymbol{S}_1$ 为

$$\boldsymbol{S}_0=\sum_{i=1}^{K}n_i\left(\boldsymbol{u}_i-\overline{\boldsymbol{u}}\right)\left(\boldsymbol{u}_i-\overline{\boldsymbol{u}}\right)^{\mathrm{T}} \tag{1}$$

$$\boldsymbol{S}_1=\sum_{i=1}^{K}\sum_{y_j=i}\left(\boldsymbol{u}_i-\boldsymbol{x}_j\right)\left(\boldsymbol{u}_i-\boldsymbol{x}_j\right)^{\mathrm{T}} \tag{2}$$

其中,n_j 为 D_j 中数据个数,$\boldsymbol{u}_j=\dfrac{1}{n_j}\sum_{\left(\boldsymbol{x}_i,y_i\right)\in D_j}\boldsymbol{x}_i$ 为 D_j 中数据的均值,$\overline{\boldsymbol{u}}=\dfrac{1}{l}\sum_{i=1}^{l}\boldsymbol{x}_i$ 为 D 中所有数据的均值. 定义代价函数:

$$J=\frac{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{S}_0\boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{S}_1\boldsymbol{w}} \tag{3}$$

LDA的目标即寻找最佳投影向量 $\boldsymbol{w}$ 使 J 最大,即投影后的数据有最小类内距离和最大类间距离. 由于 $\boldsymbol{S}_0$ 为正定矩阵,那么当 $\boldsymbol{S}_1$ 非奇异时,最优投影方向即为 $\left(\boldsymbol{S}_1\right)^{-1}\boldsymbol{S}_0$ 最大特征值 λ 对应的特征向量 $\boldsymbol{w}$,进而将数据投影为 $x_i'=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i$. 对于多类别的分类问题,基于LDA可以选择多个特征向量对数据进行降维.

2.2　泄露检测

TVLA(Test Vector Leakage Assessment)[21]是一种基于统计检验的泄露检测方法,用于对两个方差和大小不一致的样本集,检测二者均值是否有显著差异. 对于两个样本集,构造统计量:

$$t=\frac{\left|\mu_0-\mu_1\right|}{\sqrt{\dfrac{\delta_0^2}{n_0}+\dfrac{\delta_1^2}{n_1}}} \tag{4}$$

其中,$\left(\mu_0,\delta_0^2,n_0\right)$ 和 $\left(\mu_1,\delta_1^2,n_1\right)$ 分别为两个样本集的均值、方差和样本数. 根据显著性水平和自由度,可确定出阈值 C. 当 $t>C$ 时,则拒绝零假设,并判定 $\mu_0\neq\mu_1$. 此时表明 S_0 和 S_1 的均值有显著性差异,密码算法实现过程中存在信息泄露. 当 $v\geqslant 1\,000$ 时,可将判定准则简化为:若 $t>4.5$,则判定 $\mu_0\neq\mu_1$.

3　攻击方法

3.1　模乘中的访存操作

模幂实现中的操作数通常为大整数,采用分块方式存储. 操作数 a 可表示为 $\left(a_{s-1},\cdots,a_1,a_0\right)$,其中 a_i 为 u 比特,即

$$a=a_{s-1}2^{u(s-1)}+\cdots+a_1 2^u+a_0$$

实现中首先分配连续的区域分块存储,并将 a_0(或 a_{s-1})的地址作为 a 的地址,而其它数据块 a_i 的地址可由 a 的地址确定.

蒙哥马利模乘算法[22]是一种被广泛采用的模乘实现算法,将对模数 N 的约减转化为对特殊模数 R 的约减,其中,$R=2^{\lfloor\log_2 N\rfloor}$,且 $\gcd\left(R,N\right)=1$. 对于 a 和 b,蒙哥马利模乘计算:$c=a\times b\times R^{-1}\bmod N$.

根据对数据处理方式的不同,蒙哥马利模乘可分

为多种实现方法[22]，其中，FIOS（Fined Integrated Operated Scanning）是一种典型的实现方法，如算法1所示.

算法1 基于FIOS的蒙哥马利模乘实现算法

输入：$a=(a_{s-1},\cdots,a_1,a_0),b=(b_{s-1},\cdots,b_1,b_0)$

$N=(N_{s-1},\cdots,N_1,N_0),N'=-N^{-1} \bmod 2^u$

输出：$z=a\times b\times 2^{-us} \bmod N$

1. $z=(0,0,\cdots,0)$
2. FOR $i=0$ to $s-1$
3. $(C,S)=z_0+a_0\times b_i$
4. $z_1=z_1+C$
5. $g=S\times n' \bmod 2^t$
6. $(C,S)=S+g\times N_0$
7. FOR $j=1$ to $s-1$
8. $(C,S)=z_j+a_j\times b_i+C$
9. $z_{j+1}=z_{j+1}+C$
10. $(C,S)=S+g\times N_j$
11. $z_{j-1}=S$
12. $(C,S)=z_s+C$
13. $z_{s-1}=S$
14. $z_s=z_{s+1}+C$
15. $z_{s+1}=0$
16. IF $z\geqslant N$ THEN $z=z-N$

算法1中分别执行形如$a_j\times b_i$乘法运算，其中$0\leqslant i,j<s$. 乘法运算可采用如阵列乘法器、booth编码等多种架构实现. 无论采用何种架构，其中都包括多次数据的读写操作. 首先从指定地址读取a_j和b_i存入到乘数寄存器和被乘数寄存器；然后对a_j和b_i执行乘法运算；最后在运算结束后，再将运算结果写入到乘积寄存器中.

模乘运算中包括多次对数据块的读操作. 在算法1中，第3行执行$a_0\times b_i$，首先需访问存储a_0和b_i的地址，然后再执行乘法运算；第8行执行$a_j\times b_i$，则需依次访问存储a_j和b_i的地址. 结合算法1的运算流程，读取操作数的顺序如下：

$$b_0,a_0,a_1,\cdots,a_{s-1}$$
$$b_1,a_0,a_1,\cdots,a_{s-1}$$
$$\cdots$$
$$b_{s-1},a_0,a_1,\cdots,a_{s-1}$$

本文称模乘运算中如上对操作数的去操作为访存操作. 该操作是模乘运算的内部操作，在所有模乘实现中都存在，区别仅在于对数据的访问顺序不同. 该操作的功耗具有地址依赖性，即访问不同地址对应的功耗服从不同分布. 利用此操作的功耗可以确定出操作数的地址，称这种现象为基于访存的泄露. 操作数存储的地址仅由所在存储区域决定，而与具体存储的数据无关. 该泄露与操作数的具体取值无关，掩码方案或模幂算法并不会改变其泄露特性. 综上，尽管访存操作在模乘运算中具有普遍性，且不受掩码等中间数据随机化防护措施的影响，但该泄露目前较少被关注.

3.2 泄露分析

模幂实现中通常分配一块专用存储区域以存储中间数据，由此在运算中只需对数据存储地址进行配置，而不需对数据本身直接进行读写操作. 定义配置模式CR即模乘对应的输入输出地址.

对于模数等固定参数，在运算中其地址固定不变；而对于模乘的输入输出，则需根据算法的需求进行配置. 不失一般性，记分配的存储区域为A,B和H，用于保存模乘的输入和输出；模乘支持4种配置模式，如表1所示. 模乘可分为S和M两种类型，其中S表示模乘两个操作数相等. 由配置模式可确定出模乘对应的类型. 当CR为0和1时，当前的模乘类型为S；而当CR为2和3时，当前模乘类型为M.

表1 配置模式与输入输出地址对应关系

配置模式	输入1	输入2	输出
0	A	A	B
1	B	B	A
2	A	H	B
3	B	H	A

算法2在在经典的平方-乘模幂实现的基础上，增加了底数掩码和指数掩码两种防护，其中r_1为与N等长的随机数，r_2为64比特随机数，len(·)为计算该数据的比特长度，read(·)为读取对应地址的数据. 算法2首先在地址H中写入m，然后在每个循环中处理1比特密钥，通过修改配置模式CR对模乘的输入地址和输出地址进行配置. 在算法种，若CR=0，则将A作为模乘输入，运算结果保存在B中，当前模乘为S. 若下一个模乘仍为S，那么则令CR=1，将B作为输入，而将结果存储到A中；反之若下一个模乘为M，则令CR=3. 在运算结束后，从对应的地址种读取数据，将其作为模幂的运算结果. 由此，模幂被转化为一系列的模乘.

模幂掩码实现的泄露与具体实现方式相关. 如对于算法2第7行的IF语句对应两个分支，若二者周期数不同，那么从功耗曲线中能够识别出该语句执行时间，进而确定出执行的分支，从而恢复出对应的密钥比特. 此类泄露主要存在于软件实现，而在硬件实现中则较易规避.

模乘是模幂中的关键运算，本文考虑利用模乘中的泄露分析模幂实现. 针对算法2攻击的关键即区分S和M两类模乘. S和M两类模乘基于同一硬件模块实现，因而二者首先主要体现在操作数分布不同. 由此产生的泄露称为基于操作数的泄露，也是目前相关分析主要关注的泄露. 此外，本文发现模乘运算种还存在基于访存的泄露，也可对模幂掩码实现进行攻击. 利用模乘中基于访存泄

算法2　带掩码防护的模幂实现

输入:$m, d, N, \varphi(N), r_1, r_2$

输出:$m^d \bmod N$

1. $m' = m \times r_1^{-e} \bmod N$
2. $d' = d + r_2 \times \varphi(N)$
3. $A = B = 1, H = m', \mathrm{CR} = 0$
4. FOR $i = \mathrm{len}(d') - 1$ to 0:
5. 　　启动模乘
6. 　　$\mathrm{CR} = 1 - \mathrm{CR}$
7. 　　IF($d'_i = 1$) THEN
8. 　　$\mathrm{CR} = 2 + \mathrm{CR}$
9. 　　启动模乘
10. 　　$\mathrm{CR} = 3 - \mathrm{CR}$
11. IF(CR = 0或2) THEN $T = \mathrm{read}(B)$
12. ELSE $T = \mathrm{read}(A)$
13. $T = T \times r \bmod N$
14. 返回 T

露可以确定出模乘操作数的存储地址,进而得到模乘对应的配置模式和类型,最终恢复出对应的密钥比特.

基于操作数的泄露与基于访存的泄露在模乘运算中可能同时存在,但在模幂掩码实现中二者利用的难度有明显的区别. 指数掩码使得每次模幂运算中对应的密钥比特被随机化,而底数掩码使得操作数分布的差异变小,不同模幂运算对应功耗曲线并无直接关联,这些都降低的降低了基于操作数的泄露. 而基于访存泄露仅与操作数地址有关,即使对于模幂掩码实现,操作数地址和访存操作的功耗分布未被改变. 因此,此类泄露刻画和利用难度相对较低.

3.3　基于LDA的模板攻击

针对模幂掩码实现的侧信道分析是一种典型的分类问题,即将功耗曲线按照攻击的需求进行分类. 由于模乘的泄露分布在多个采样点中,攻击需考虑如何对高维数据进行分析. 本文则通过LDA对曲线进行降维,进而提出了基于LDA的模板攻击方法. 该方法主要包括如下5步:

(1)曲线提取:从模幂对应的曲线中提取出模乘对应的曲线段,作为训练集和匹配集;

(2)泄露检测:对训练集进行泄露检测以识别出泄露特征点,并对训练集和匹配集进行泄露特征提取;

(3)模板刻画:基于训练集中曲线确定最佳投影方向及每类曲线对应的分布,作为该类曲线对应模板;

(4)模板匹配:利用所建立的功耗模板,对匹配集的曲线降维和模板匹配,确定出对应的曲线类别;

(5)密钥恢复:对模板匹配结果进行纠错,恢复出对应的密钥比特.

3.3.1　曲线提取

攻击需从模幂对应的曲线中识别出所有与密钥相关模乘对应的曲线段. 在算法2中,每个模乘都与密钥比特相关,只有已知所有的模乘对应的类型,才能恢复出密钥. 对于第 i 个模乘 M_i,令 $\boldsymbol{O}_i$ 表示 M_i 对应的曲线段. 定义 Y_i 为 $\boldsymbol{O}_i$ 对应的曲线模式,即 M_i 对应的配置模式. 定义 X_i 为 $\boldsymbol{O}_i$ 对应的曲线类型,当 M_i 为S时,$X_i = 0$;当 M_i 为M时,$X_i = 1$. 显然有:$X_i = \lfloor Y_i/2 \rfloor$.

攻击分为训练和匹配两个阶段. 在训练阶段中,首先提取出所有模乘对应的曲线段. 模幂中的输入设置为随机数,由此可保证模乘操作数随机,避免受到与操作数相关特征的干扰. 根据算法2可确定出模乘对应曲线的模式和类型. 定义训练集 $D_{\mathrm{L}} = \left\{ \left(\boldsymbol{O}_i, Y_i, X_i \right) | 1 \leqslant i \leqslant n_{\mathrm{L}} \right\}$,其中,下标L代表学习,$n_{\mathrm{L}}$ 为曲线的个数. D_{L} 中曲线对应的模式 Y_i 和类型 X_i 已知. D_{L} 中曲线按照模式可分为4类,即

$$C_{1,j} = \left\{ \left(\boldsymbol{O}_i, Y_i, X_i \right) \in D_{\mathrm{L}} | Y_i = j \right\}, \text{其中}, j = 0, 1, 2, 3$$

D_{L} 中的曲线按照类型可分为2类,即

$$C_{2,j} = \left\{ \left(\boldsymbol{O}_i, Y_i, X_i \right) \in D_{\mathrm{L}} | X_i = j \right\}, \text{其中}, j = 0, 1$$

在匹配阶段中,从模幂对应曲线中提取与密钥比特相关模乘对应曲线,得到匹配集 D_{E},其中下标E代表评估. 匹配集中曲线对应的模式及类型未知. 训练集中要求曲线模式已知,而模幂掩码实现中曲线模式通常未知. 因而在构造训练集时,要求攻击者已知密钥和掩码随机数,从而能够计算出掩码后的密钥指数,确定出每个模乘的类型及曲线模式. 由此,对攻击者的能力提出了额外的要求. 针对该问题,本节则利用RSA加密运算构造训练集. RSA加密通常采用与解密相同的软硬件模块实现,并且很少进行掩码. 攻击容易确定出模乘对应的配置模式,进而可得到足够多的曲线作为训练集. 此外,攻击中使用随机数作为RSA加密的输入,目的是保证模乘操作数服从均匀分布,避免攻击受特殊操作数的影响.

3.3.2　泄露检测

本节则利用TVLA方法确定训练集和匹配集中曲线的泄露特征点. 由于在模幂运算中4种配置模式交替出现,因此,泄露检测应能够满足所有配置模式的需求. 攻击的最终目标是确定曲线类型,因此本节将训练集 D_{L} 按照曲线类型分为 $C_{2,0}$ 和 $C_{2,1}$,然后进行TVLA检测.

D_{L} 中的第 i 条曲线记为 $\boldsymbol{O}_i = \left(O_{i,1}, O_{i,2}, \cdots, O_{i,l_{\mathrm{p}}} \right)$,其中下标p代表曲线中的样本点,$l_{\mathrm{p}}$ 为样本点的个数,$O_{i,k}$ 表示第 i 条曲线的第 k 个点. 不妨假设,曲线集 $C_{2,X}$ 中曲线的第 k 个点服从正态分布 $N\left(u_k^X, \delta_k^X\right)$,其中 $X \in \{0, 1\}$ 为曲线类别. 令 n_X 表示 $C_{2,X}$ 中曲线条数,那么其对应的均值和方差分别为

$$u_k^X=\frac{1}{n_X}\sum_{X_i=X}O_{i,k} \tag{5}$$

$$\delta_k^X=\frac{1}{n_X}\sum_{X_i=X}\left(O_{i,k}-u_k^X\right)^2 \tag{6}$$

对 $C_{2,0}$ 和 $C_{2,1}$ 中曲线逐点计算对应的统计量，其中，第 k 个点的结果为

$$t_k=\frac{\left|u_k^0-u_k^1\right|}{\sqrt{\frac{\delta_k^0}{n_0}+\frac{\delta_k^1}{n_1}}} \tag{7}$$

根据 TVLA 的判定方法，当 $t_k>4.5$ 时，则判定 u_k^0 和 u_k^1 有显著的统计差异. 从攻击的角度，该点存在泄露. 由此，定义曲线中的泄露特征点为：$\mathrm{POI}=\left\{k|t_k>4.5\right\}$.

对于 D_{L} 和 D_{E}，本文选择仅保留 POI 中的点，作为泄露特征提取的结果. 当 $t_k\leqslant 4.5$ 时，则意味着该点与曲线类型关联较小，甚至无关联. 若对无关点进行学习，那么可能会导致过学习，降低模板匹配的成功率. 为方便起见，下文中用 D_{L} 和 D_{E} 指代经泄露特征提取后的曲线集.

3.3.3 模板刻画

本文的模板攻击针对泄露特征提取后曲线进行，分为模板刻画和模板匹配两个阶段. 在模板刻画阶段中，首先通过 LDA 对训练集进行学习，确定出投影方向，并对曲线进行降维，然后再对降维后的曲线进行降维.

训练集 D_{L} 中按照配置模式可分为 4 类 $C_{1,i}$，其中，$i=0,1,2,3$. 根据式(1)和式(2)，分别计算 D_{L} 的类内距 $\boldsymbol{S}_0$ 和类间距 $\boldsymbol{S}_1$. 由于分析曲线的类数为 4，本文需确定出 2 个投影方向，将曲线投影到 2 维空间中. 根据 LDA 算法，投影方向为 $\left(\boldsymbol{S}_1\right)^{-1}\boldsymbol{S}_0$ 的最大两个特征值所对应的特征向量 $\boldsymbol{w}=(\boldsymbol{w}_1,\boldsymbol{w}_2)$. 进一步的，训练集中的曲线可被投影到 2 维空间中：$\boldsymbol{O}_j'=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{O}_j$.

利用投影后的数据，可为每种曲线模式建立对应的模板. 不妨假设，每种模式对应的投影数据服从二维正态分布. 类 $C_{1,i}$ 服从的概率分布为

$$P_i(\boldsymbol{x})=\frac{1}{2\pi\sqrt{\left\|\boldsymbol{\theta}_i\right\|}}\mathrm{e}^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{u}_i)^{\mathrm{T}}(\boldsymbol{\theta}_i)^{-1}(\boldsymbol{x}-\boldsymbol{u}_i)} \tag{8}$$

其中，$\boldsymbol{u}_i$ 和 $\boldsymbol{\theta}_i$ 为该类中曲线经投影后得到数据的均值和协方差. 由此，针对曲线类 $C_{1,i}$ 建立的模板包括两部分：最优投影方向 $\boldsymbol{w}$ 和投影后的概率分布 P_i.

3.3.4 模板匹配

匹配集 D_{E} 中的曲线来自于单次模幂运算，而其中曲线对应的模式未知. 受掩码随机数的影响，不同曲线互相独立，因此攻击中需对曲线逐条分析，目标是确定出曲线对应的模式.

模板匹配的过程包括降维和匹配两个阶段. 首先利用 LDA 所确定的投影方向对匹配集中的曲线降维，即将匹配集中曲线 $\boldsymbol{O}_j$ 投影到 2 维空间中，计算：$\boldsymbol{O}_j'=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{O}_j$. 然后对于投影后的数据 $\boldsymbol{O}_j'$，分别计算其与 4 个模板的匹配概率值 $P_i\left(\boldsymbol{O}_j'\right)$. 最后攻击将 $\boldsymbol{O}_j$ 识别为匹配概率最大的模板所对应的类，该类曲线的模式即为类的序号.

3.3.5 密钥恢复

匹配集中的曲线与模乘相对应，而由模乘对应的序列可以恢复出密钥. 对于曲线 $\boldsymbol{O}_i$，模板匹配的结果即曲线模式 Y_j. 由于 $X_j=\left\lfloor Y_j/2\right\rfloor$，因此，可确定出对应的曲线类型及模乘类型.

根据算法 2，可将模乘序列转化为密钥比特. 当相邻两个模乘为 S 和 M 时，那么其对应密钥比特为 1. 对于其它模乘，其对应的密钥比特为 0. 攻击得到的密钥为与初始密钥等价的掩码后的密钥，可直接用于解密运算中.

攻击中可能存在部分曲线被匹配到错误的配置模式中，此时需对攻击结果进行纠错. 考虑到当曲线模式识别错误时，不一定会导致曲线类型错误，因此攻击应选择对曲线类型纠错. 特别的，当曲线类型仅在少数位置出错时，通过遍历可直接恢复出密钥.

3.4 方法比较

Carbone 等提出的基于深度学习的水平侧信道分析[16]，是目前对模幂掩码实现的最有效分析方法之一. 与本文提出的攻击方法相比，二者均具有较强的分析能力，都可对采用了指数掩码、底数掩码的模幂掩码实现进行攻击. 表 2 为对两种攻击方法的比较.

表 2 与基于深度学习水平侧信道分析的比较

比较项	本文方法	Carbone 方法[16]
利用泄露类型	基于访存的泄露	基于操作数的泄露
指数掩码	不影响	被限制在单条曲线
底数掩码	不影响	泄露被降低
模幂实现方案	不影响	影响泄露大小
使用的底层技术	LDA、模板攻击	深度学习
模板匹配准确率	99.98%	99.31%

与 Carbone 方法相比，本文方法具有对攻击者要求低、使用底层技术简单，可解释性强等优点. 首先，本文方法主要利用模乘运算中基于访存操作的泄露，其不受掩码和模幂实现算法的影响；而 Carbone 的方法主要关注基于操作数的泄露，在掩码实现中此类泄露刻画和利用的难度明显变高. 其次，本文将 LDA 与模板攻击相结合，攻击涉及的底层技术较简单，可将攻击过程与泄露和实现相关联；而 Carbone 的方法则使用了相对复杂的深度学习技术，攻击过程中需要对大量参数进行训练，可解释性相对较低. 最后，两种方法的攻击能力相当，其对模乘分类的准确率均超过了 99%.

4　实验验证

4.1　实验对象

本文的实验对象为一款芯片,其通过32位CPU核ARM SC100调用硬件蒙哥马利模乘,实现了如算法2所示的模幂算法. 实验中在模幂算法运行期间记录功耗变化,对其进行分析以获取密钥. 实验环境如图1所示,其中采集主要使用了Riscure公司研制的功耗采集平台. 该平台在软件Inspector 4.12的控制下,基于读卡器Power Tracer与芯片通信,发送启动模幂命令;同时还提供了功耗测量接口,示波器则对功耗信号采样,得到本文实验曲线. 本文使用示波器为Lecroy WaveRunner 610Zi,采样频率设置为250 MHz.

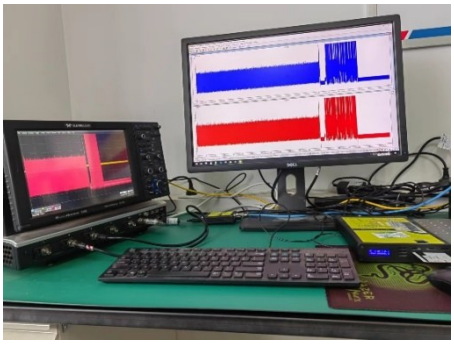


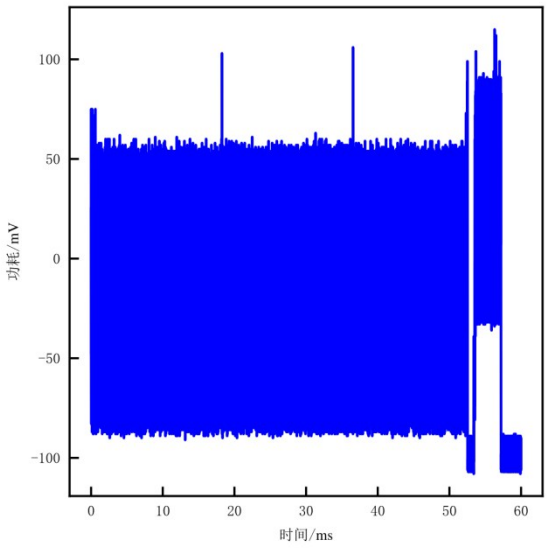
图1　功耗采集环境

模幂对应的功耗曲线具有清晰的功耗特征,如图2所示,其中图2(a)为1 024比特模幂运算对应的功耗曲线全貌图;图2(b)则为模乘对应的功耗曲线. 由于S和M两类模乘基于同一硬件模块实现,因而实验中未观测到二者对应功耗特征的区别.

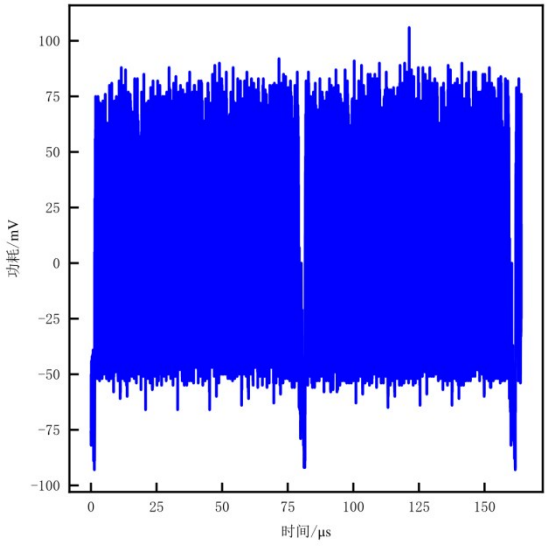
本文实验中采集了两组功耗曲线,分别对应于训练集和匹配集. 首先,采集1 000次RSA加密对应的曲线,其中幂指数设置为固定的1 024比特,输入则每次随机选取. 每条曲线中包括1 023次S和516次M,从中提取出模乘对应的曲线作为训练集. 其次,采集100次RSA解密对应的曲线,将其作为匹配集. 由于掩码后幂指数的汉明重量并不固定,因而模幂运算中包括的模乘个数也是变化的. 经统计,单条曲线中提取出的曲线个数约为1 600个.

4.2　泄露分析

本节以训练集中的曲线作为分析对象,基于TVLA对模乘中的基于操作数的泄露与基于访存的泄露进行分析和比较. 首先,将训练集按照曲线模式分为4类,检测曲线中是否存在基于访存的泄露. 为避免基于操作数的泄露对测试结果的影响,从每类曲线中随机选取5 000条作为测试对象,以使得模乘对应的操作数随机分布. 对4类曲线两两进行TVLA检测,其中模式为0和1、1和3两次测试对应的t值曲线分别如图3(a)和图3(b)所示. 说明:图中红色直线表示阈值t=4.5.



(a) RSA模幂运算功耗曲线全貌图



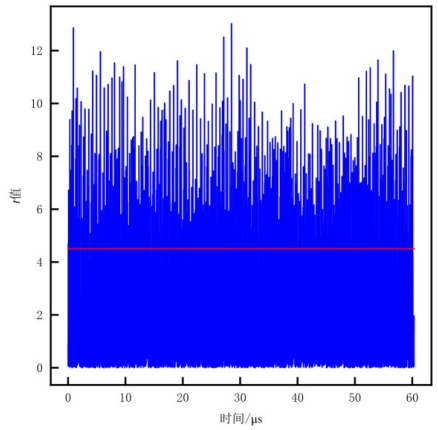
(b) 功耗曲线局部放大图

图2　模幂及模乘对应的功耗特征

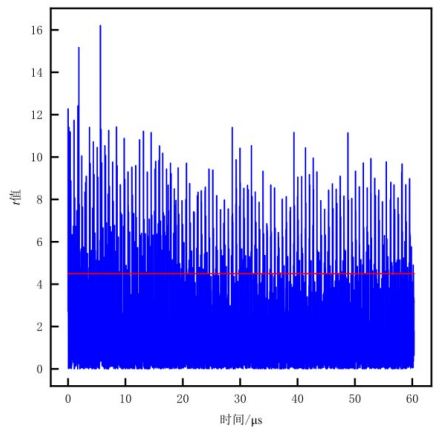
由图3可知,满足$t \geqslant 4.5$的点分布在整个模乘对应的曲线中,点数约为800~1 800个. 根据TVLA的判定准则,不同配置模式模乘对应曲线在这些点上有显著性差异,即基于访存的泄露分布在整个模乘运算中.

为避免基于访存泄露对检测结果的影响,本文选择以模式为2的曲线类对基于操作数的泄露进行测试. 该类曲线对应的模乘类型为M,在单次模幂运算中该类模乘有一个相同的操作数. 因此,利用两次模幂对应曲线进行TVLA检测,结果如图4所示.

由图4知,基于操作数的泄露在模乘执行过程中同样存在,但其仅在少数点中存在. 图4中的两个t值曲

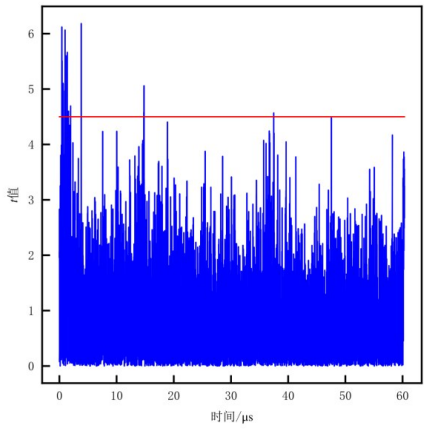


(*a*) 模式0和1对应的TVLA测试结果

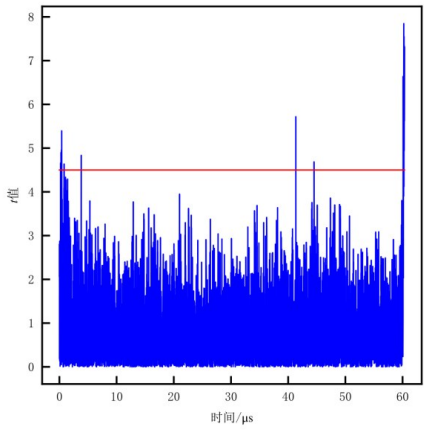


(*b*) 模式1和3对应的TVLA测试结果

图3 模乘中与访存泄露相关的TVLA测试



(*a*) 第一组曲线对应的TVLA测试结果



(*b*) 第二组曲线对应的TVLA测试结果

图4 模乘中与操作数相关泄露的TVLA测试

线并不完全一致,表明基于操作数的泄露与测试所针对的操作数相关,不同操作数将导致不同的泄露. 特别的,对图3与图4中的t值曲线进行比较,易知基于操作数的泄露远小于基于访存的泄露.

针对模幂攻击的目标是将模乘分为S和M,因此下面将曲线按照类型分为S和M两类. 从每类曲线中随机选择10 000条曲线,进行TVLA检测,结果如图5所示.

根据实验结果,图5中满足$t \geqslant 4.5$的点有942个,其均有助于曲线分类. 从曲线中保留这些点,将其作为下一步分析对象.

4.3 经典模板攻击

本节对曲线进行模板攻击,分为模板刻画和模板匹配两步. 首先,将训练集中的曲线分为S和M两类,从每类中各随机选取10 000条曲线;然后计算每类曲线对应的均值向量和协方差,作为该类对应的模板.

在模板匹配阶段,从训练集和匹配集中各随机选取10 000条曲线进行攻击. 对于每条曲线,基于多元高斯分布计算其与两个模板的匹配概率,选择其中匹配概率最大对应的类作为攻击的结果,匹配准确率如表3所示.

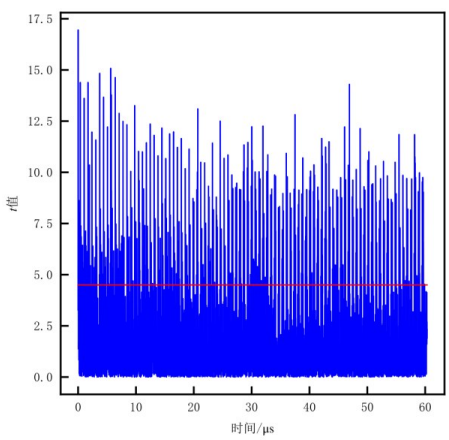


图5 针对S和M的TVLA测试

显然,经典模板攻击并未恢复出密钥. 其原因在于

表3　经典模板攻击匹配准确率

比较项	训练集	匹配集
整体匹配准确率	68.8%	63.9%
S匹配准确率	71.8%	71.5%
M匹配准确率	65.8%	56.3%

本次实验中主要利用基于操作数泄露,而在不同次模幂运算中操作数的分布区别较小,训练集与匹配集中M对应的泄露分布并不一致,从而导致模板匹配的准确率相对较低.

4.4　基于LDA的模板攻击

本节对曲线进行基于LDA的模板攻击实验. 首先对于每种模式,从训练集中随机选取10 000条曲线,共得到40 000条曲线. 曲线按照配置模式可分为4类,那么在LDA算法中通过两个投影方向,将曲线降维到二维空间中. 对训练集曲线进行LDA降维后,形成的散点图如图6所示,不同模式用颜色进行标识.

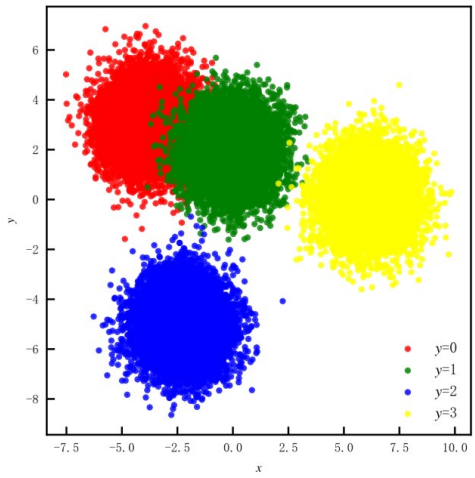


图6　训练集中曲线经LDA后的散点图

如图6所示,每条曲线降维后即对用于图中的一个点. 对这些点进行模板攻击,计算不同配置模式对应的均值和协方差,将其作为模板. 经测试,训练集中曲线模式识别的正确率约为99.89%,而匹配集中曲线模式识别的正确率则约为99.86%. 更进一步的,由曲线模式可确定出对应的模乘类型. 经测试,训练集中模乘类型匹配的准确率为99.98%. 匹配集对应于100次模幂,其中每个模幂包括约1 600个模乘. 经测试,匹配集中共有74次模幂运算,其模乘类型匹配的准确率达到了100%,可直接恢复出模幂密钥.

5　总结

模乘运算中存在多种类型泄露,新的泄露类型将为模幂安全实现引入新的威胁. 本文发现模乘中存在基于访存的泄露,其不受掩码的影响,并通过实验证明了此类泄露分布在整个模乘运算中. 此外,本文提出了基于线性判别分析的模板攻击方法,充分利用了LDA和模板攻击各自的优势. 特别的,基于线性判别分析对数据进行降维,解决了高维数据分析难度高的问题. 实验结果表明,本文方法可有效的恢复出模幂掩码实现的密钥.

参考文献

[1] MESSERGES T S, DABBISH E A, SLOAN R H. Power analysis attacks of modular exponentiation in smartcards[C]// Cryptographic Hardware and Embedded Systems. Berlin: Springer, 1999: 144-157.

[2] SCHINDLER W, ITOH K. Exponent blinding does not always lift (partial) spa resistance to higher-level security[C]//Applied Cryptography and Network Security. Berlin: Springer, 2011: 73-90.

[3] HOMMA N, MIYAMOTO A, AOKI T, et al. Collision-based power analysis of modular exponentiation using chosen-message pairs[C]//Cryptographic Hardware and Embedded Systems—CHES 2008. Berlin: Springer, 2008: 15-29.

[4] AMIEL F, FEIX B, TUNSTALL M, et al. Distinguishing multiplications from squaring operations[C]//Selected Areas in Cryptography. Berlin: Springer, 2009: 346-360.

[5] WITTEMAN M F, VAN WOUDENBERG J G J, MENARINI F. Defeating RSA multiply-always and message blinding countermeasures[C]//Topics in Cryptology—CT-RSA 2011. Berlin: Springer, 2011: 77-88.

[6] CLAVIER C, FEIX B, GAGNEROT G, et al. ROSETTA for single trace analysis[C]//Progress in Cryptology-INDOCRYPT 2012: 13th International Conference on Cryptology in India. Berlin: Springer, 2012: 140-155.

[7] CLAVIER C, FEIX B, GAGNEROT G, et al. Horizontal correlation analysis on exponentiation[C]//Information and Communications Security. Berlin: Springer, 2010: 46-61.

[8] COURRÈGE JC, FEIX B, ROUSSELLET M. Simple power analysis on exponentiation revisited[C]//Smart Card Research and Advanced Application: 9th IFIP WG 8.8/11.2 International Conference, CARDIS 2010. Berlin: Springer, 2010: 65-79.

[9] BAUER A, JAULMES E, PROUFF E, et al. Horizontal and vertical side-channel attacks against secure RSA implementations[C]//Topics in Cryptology—CT-RSA 2013. Berlin: Springer, 2013: 1-17.

[10] HANLEY N, KIM H, TUNSTALL M. Exploiting collisions in addition chain-based exponentiation algorithms using a single trace[C]//Lecture Notes in Computer Science. Cham:

Springer International Publishing, 2015: 431-448.
[11] BATINA L, CHMIELEWSKI Ł, PAPACHRISTODOULOU L, et al. Online template attacks[J]. Journal of Cryptographic Engineering, 2019, 9(1): 21-36.
[12] DUGARDIN M, PAPACHRISTODOULOU L, NAJM Z, et al. Dismantling real-world ECC with horizontal and vertical template attacks[C]//Constructive Side-Channel Analysis and Secure Design: 7th International Workshop, COSADE 2016. Cham: Springer International Publishing, 2016: 88-108.
[13] HHEYSZL J, IBING A, MANGARD S, et al. Clustering algorithms for non-profiled single-execution attacks on exponentiations[M]//Smart Card Research and Advanced Applications. Cham: Springer International Publishing, 2014: 79-93.
[14] PERIN G, CHMIELEWSKI Ł. A semi-parametric approach for side-channel attacks on protected RSA implementations [M]//Smart Card Research and Advanced Applications. Cham: Springer International Publishing, 2016: 34-53.
[15] MAGHREBI H, PORTIGLIATTI T, PROUFF E. Breaking cryptographic implementations using deep learning techniques[M]//Security, Privacy, and Applied Cryptography Engineering. Cham: Springer International Publishing, 2016: 3-26.
[16] CARBONE M, CONIN V, CORNÉLIE M A, et al. Deep learning to evaluate secure RSA implementations[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2019, 2019(2): 132-161.
[17] ZAID G, BOSSUET L, HABRARD A, et al. Efficiency through diversity in ensemble models applied to side-channel attacks[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2021, 2021(3): 60-96.
[18] PERIN G, CHMIELEWSKI Ł, BATINA L, et al. Keep it unsupervised: Horizontal attacks meet deep learning[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2020, 2021(1): 343-372.
[19] SAITO K, ITO A, UENO R, et al. One truth prevails: A deep-learning based single-trace power analysis on RSA—CRT with windowed exponentiation[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2022, 2022(4): 490-526.
[20] 周志华. 机器学习[M]. 北京: 清华大学出版社, 2016: 60-63.
ZHOU Z H. Machine Learning[M]. Beijing: Tsinghua University Press, 2016: 60-63. (in Chinese)
[21] GOODWILL G, JUN B, JAFFE J, et al. A testing methodology for side-channel resistance validation[C]//NIST Non-Invasive Attack Testing Workshop (NIAT2011). Gaithersburg: NIST, 2011: 115-136.
[22] KAYA KOC C, ACAR T, KALISKI B S. Analyzing and comparing Montgomery multiplication algorithms[J]. IEEE Micro, 1996, 16(3): 26-33.

作者简介

韩绪仓 男,1987年10月出生于陕西省西安市,现为中国科学院软件研究所博士研究生,研究方向为密码算法侧信道分析与防护.
E-mail: xucang2020@isccas.ac.cn

陈波涛 男,1976年12月出生于四川省汉源县,高级工程师,现为北京中电华大电子设计有限责任公司副总经理,主要研究方向为集成电路设计、芯片硬件安全攻防. 中国电子学会会员编号:E190029014M.
E-mail: chenbt@hed.com.cn

曹伟琼(通讯作者) 女,1986年1月出生于桂林市. 现为中国科学院软件研究所助理研究员. 主要研究方向为公钥算法的侧信道分析与防护.
E-mail: caoweiqiong@iscas.ac.cn

陈 华 女,1976年10月生于山东省日照市,现为中国科学院软件研究所正高级工程师,博士生导师,研究方向为侧信道分析与防护、密码检测.
E-mail: chenhua@iscas.ac.cn

李昊远 男,1995年11月出生于山东省,现为中国科学院软件研究所博士研究生,主要研究方向为密码算法的侧信道分析与防护.
E-mail: haoyuan2019@isccas.ac.cn