出站报告

物理攻击密码系统的防御研究

--改进的防御差异能量攻击AES密码系统的定值遮盖法及其安全性能分析

A Study on Securing cryptography against Physical Analysis

-- A Modification of Fixed-Value Masking Method and Security analysis on Securing AES against DPA

顾晓东

合作导师：彭思龙研究员

中国科学院自动化研究所国家专用集成电路工程中心

2005.03.05

摘要

在未来的几年里智能卡将广泛地应用到交通、电子商务等领域，其主要的优势在于内部的数据只用于内部处理，只有运算的结果可以获取。然而智能卡的内部数据可以通过边际信道攻击得到。当用输入信息和密钥进行加密运算时，能量消耗和电磁辐射等信息会泄漏，在边际信道攻击中，这些信息被利用来得到密钥。有时攻击者会利用时间信息或者诱导的故障信息来攻击密钥。差异能量攻击（DPA）是一种利用能量消耗信息的边际信道攻击方法且容易实施，从而成为智能卡安全性的最大威胁。防御 DPA 攻击的方法分为两类，一种是软件防御法、一种是硬件防御法，现今的智能卡一般都装备硬件防御，但是只有结合软、硬件防御才能更好地防御差异能量攻击。

AES 是 NIST 于 2000 年宣布的用于替代 DES 的分组加密标准，也是目前最完善的密码体系，因此针对 AES 密码体系的差异能量攻击及其防御措施成为研究的热点。遮盖方法是一种能防御差异能量攻击的软件防御法，但现有的方法难以防御二阶差异能量攻击（SODPA）。2001 年，Messerges[31]提出随机遮盖法以保护 AES 系统免于 DPA 攻击，这种方法用随机产生的遮盖值，每次必须计算新的 S 盒，因此得占用很多 RAM 空间和要求很强的运算能力。为此 Akkar、Golic[34-36]提出乘积遮盖法，Itoh 等[33]提出定值遮盖法，定值遮盖法相比乘积遮盖法更具实用性，但文献[37] [40]表明现有的定值遮盖法无法对抗二阶差异能量攻击。

为了防御二阶差异能量攻击，我们必须使攻击者难以获取装载遮盖的时间和每轮选择的遮盖值，在本文中，我们针对 AES 密码体系提出了一种改进的简化定值遮盖法（Hwasun Chang）以对抗高阶差异能量攻击:事先装载两个遮盖在存储器中，随机地选用遮盖，使攻击者难以判断装载遮盖的点；同时准备了两个遮盖的明文，只有选上的明文被处理，使得攻击者无法知道遮盖的装载时间。最后我们进行了算法的性能分析，存储遮盖所需的内存是以前方法的 33%，应用遮盖的异或操作的数目是以前方法的 18%。在实际应用中，内存的减少不会优化整个算法太多但异或操作数目的减少能极大地改进算法的性能，对 32 位智能卡而言异或操作数目大约减少 10%。同时我们还讨论了产生遮盖所要求的特性、遮盖的数目、所要求的加法运算、完善防御方法所需的内存以及算法的安全性。

Abstract

Major credit card companies are planning to convert most of credit cards with magnetic stripe into smart cards within a few years. And usage of smart cards are increasing in such fields like transportation. electronic money, ID card, etc. Major advantage of smart cards is that internal data like secret key can be used for internal processing and only the result is open to the public access.

However, the internal data kept inside smart cards and used internally can be found out using side channel attack. When cryptographic processing is occurred using input message and secret key, information like power consumption or electromagnetic radiation may be leaked. In side channel attack, the information is used to find out the secret key. Sometimes, attackers utilize timing information or induced faults during computation. Differential Power Analysis (DPA) is a kind of side channel attacks that makes use of power consumption information. DPA is a real threat because attackers can mount DPA with relatively cheap equipments and without knowing the internal implementation. Countermeasures against DPA can be divided into two categories. One is by hardware and the other is by software. Smart card chips manufactured recently are equipped with hardware countermeasures. But it is generally recognized that DPA can be prevented effectively only by using both hardware and software countermeasures.

AES is the standard block cipher selected by NIST to replace DES in 2000. Masking methods were proposed as software countermeasure against DPA. But previous masking methods are vulnerable to Second Order DPA(SODPA) and can be made simpler in regard to memory and processing requirement.

In this thesis, a modification of Simple fixed-value masking method(Hwasun Chang) that is resistant to HODPA and more efficient than previous methods is proposed. We load two masks into memory that can be used for operand of xor instruction, and we can use one mask among them randomly, at the same time, we prepared two masked plaintext and only the selected one is processed , so the attacker will not be able to decide the mask loading time. In the end, the performance of the algorithm is analyzed, the required memory for storing mask is 33% of previous method and the number of XOR operation for applying mask is 18% of previous method. In practice, the reduction of memory usage will not affect the overall algorithm size much. But reducing the number of xor operations can improve the algorithm performance by about 10% in 32 bit smart cards optimized for each round. In analysis process, the required properties of the generated masks, the appropriate number of mask, the required additional processing and memory for implementing the proposed countermeasure, and the security of the proposed method are suggested.

1. 前言

由于在密码算法的实现过程中，会有一些边际信道没有被包含在其理论模型中，因此即使是号称安全的密码算法也会因边际信道泄漏而被攻击。智能卡是一种可防止入侵的设备，目前各种各样的智能卡广泛的应用在移动通信 、金融/银行业、社保/医疗、公交、加油、身份认证和具有一定社会应用规模的计量仪表等领域。

智能卡是电子商务的未来，其主要的优势在于内部的数据只用于内部处理，只有运算的结果可以获取。然而智能卡的内部数据可以通过边际信道攻击得到。当用输入信息和密钥进行加密运算时，能量消耗和电磁辐射等信息会泄漏，在边际信道攻击中，这些信息被利用来得到密钥。有时攻击者会利用时间信息或者诱导的故障信息来攻击密钥。

在传统的密码体系中，密码算法被转成数学模型，由分析数学模型可知不存在能破译密码的低复杂度的算法。然而这些数学模型并没有考虑到密码算法实现的问题，这就使得攻击者可以通过非正式方式来破解所谓安全的密码算法，它们包括电子探针[1]、反向工程、存储器读取技术[2]（扫描电子显微镜或超导量子图像显示设备）。近年来，许多新型的攻击技术被提出[3][4][5][6][7]，这些攻击技术是靠容易获得的低价设备实施攻击，被称为基于物理特征的密码攻击技术，基于物理特征的密码攻击技术包括故障攻击、侵入攻击、时间攻击、电压攻击、简单的能量攻击、差异能量攻击、电磁辐射攻击、高阶差分攻击和汉明差分攻击，其中最著名的攻击方法有： SPA（Simple Power Analysis/简单能量攻击法）和 DPA(Differential Power Analysis/差异能量攻击法)。DPA 由 Paul Kocher、Joshua Jaffe 和 Benjamin Jun 于 1998 年提出。它是在 Paul Kocher 提出的时间攻击法（Timing Attacks）基础上的改进，其原理是基于数据在加密算法的运行过程中呈现出的能量消耗/功率的变化，利用统计方法检测和分析这些变化来推断密钥。利用这些技术，攻击者可以在获得密码算法运行载体（计算机、保密机、加密盒、IC 卡等等）的情况下，快速地获得密钥，从而破译整个密码系统。

差异能量攻击（DPA）是一种利用能量消耗信息的边际信道攻击方法且容易实施，从而成为智能卡安全性的最大威胁。为了防止攻击者借助边际信道泄漏的信息，如能量消耗、执行时间、故障时的输入输出行为、辐射等攻击智能卡，必须采取防御措施。防御 DPA 攻击的方法分为两类，一种是软件防御法、一种是硬件防御法，现今的智能卡一般都装备硬件防御，但是只有结合软、硬件防御才能更好地防御差异能量攻击。本文先是概要地介绍了各种密码体系以及各种物理攻击技术，最后针对 AES 密码系统研究了能防御差异能量攻击的定值 Masking 方法，由于现有的各种定值 Masking 方法难以防御二阶差异能量攻击（SODPA）。我们提出了能防御二阶差异能量攻击的改进的定值遮盖法，并进行了算法的安全性能分析。

本文的组织如下：第一节是前言，第二节介绍了常见的密码算法，第三节介绍了各种攻击技术，包括故障攻击、时间攻击、简单能量攻击和差异能量攻击，第四节介绍了各种防御 DPA 攻击的遮盖方法并提出了我们的算法，第五节是算法的安全性能分析及结论。

2. 密码算法

2.1 DES 算法

1977 年 1 月 15 日美国正式公布实施的数据加密标准 DES 是一个众所周知的分组密码系统，其分组长度为 64 位，密钥中有 8 位是奇偶校验位，实际密钥的长度为 56 位，尽管 DES 目前已被高级加密标准 AES 所取代，但其设计思想仍然值得借鉴[8]。

DES 是一种 16 轮 Feistel 网络架构的对称式加解密系统，它使用 64 位的密钥 K，把 64 位的明文加密成 64 位的密文，或者是把 64 位的密文解密成 64 位的明文。在每一轮中，DES 使用轮函数 f 与子密钥 K_i 更正 2 个 32 位的缓存器 R_i 与 L_i，DES 加密流程如图 1 、2 所示。

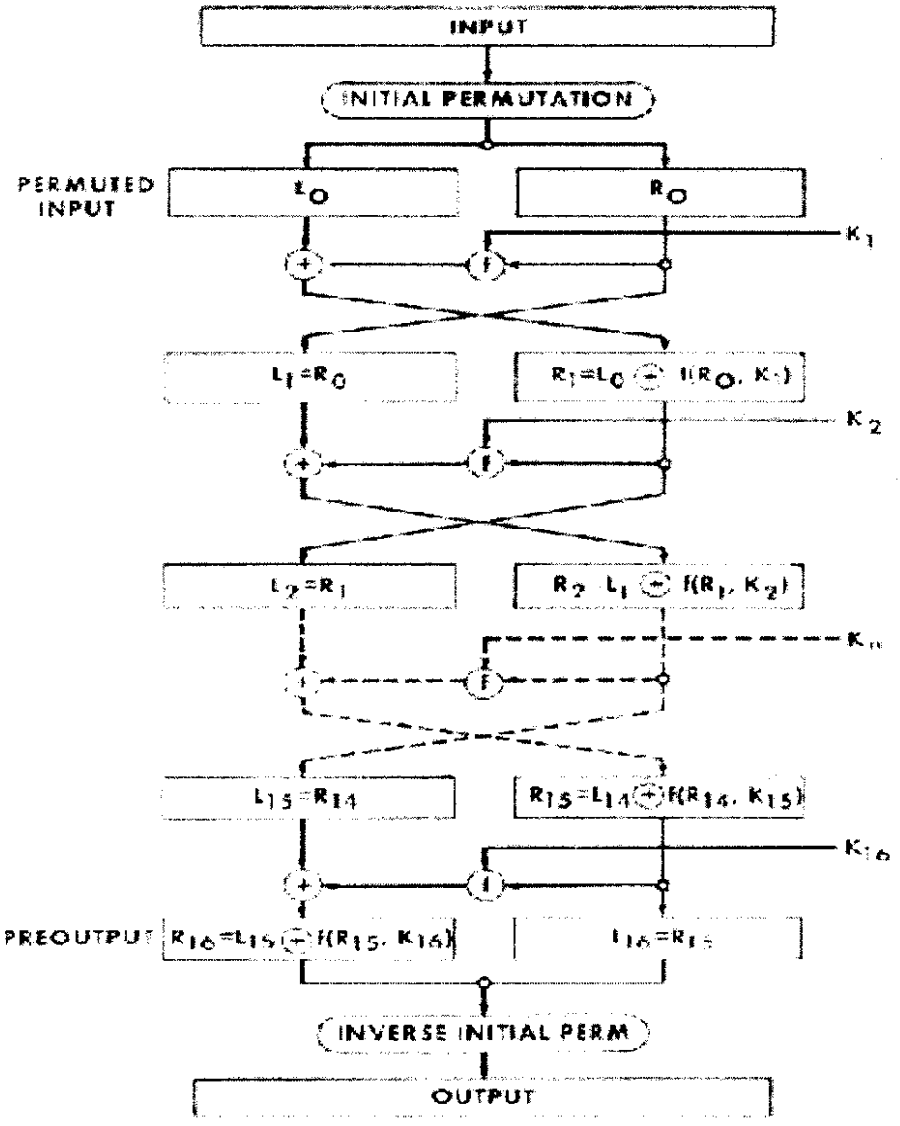

图 1：DES 的流程架构图

INPUT: INPUT, K

OUTPUT: OUTPUT=DES_K(INPUT)

1. 从密钥 K 中推导出 16 个子密钥 K_1、K_2、…、K_{16}（见图 13）
2. L_0R_0 = INITIAL PERMUTATION(INPUT)
3. for i = 1 to 16
4. $L_i = R_{i-1}$
5. $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$
6. endfor
7. OUTPUT = INVERSE INITIAL PERM($R_{16}L_{16}$)
8. return OUTPUT

图 2：DES 算法

DES 的主要流程为：明文 INPUT 经过 INITIAL PERMUTATION 后得到各 32 位的 L_0 与 R_0，L_0 为 64 位 L_0R_0 的左半边，R_0 为 64 位 L_0R_0 的右半边。接着进行 16 轮的运算，在每一轮中使用轮函数 f 与子密钥 K_i 更正 2 个 32 位的缓存器 R_i 与 L_i，每轮间的关系为

$$L_i = R_{i-1},$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i) \quad 。$$

每个轮所使用的子密钥 K_i 皆不相同，48 位的子密钥 K_i 是由 64 位的密钥推导得出（图 13）。函数 f 的输入为 32 位的 R_{i-1} 与 48 位的子密钥 K_i，输出为 32 位。16 轮运算后的结果 $L_{16}R_{16}$ 左右颠倒成 $R_{16}L_{16}$ 再经 INVERSE INITIAL PERM 后得到 DES 算法的输出结果 OUTPUT。输出 OUTPUT 是 INPUT 用密钥 K 经过 DES 密码算法加密后的结果。

由于 DES 密码算法与其使用的置换函数、函数皆为公开的，唯一不公开的资料只有密钥，因此 DES 算法的安全性就靠密钥的安全性来保证。

2.2 AES 算法

1997 年 4 月 15 日美国国家标准技术研究所 NIST（National Institute of Standard and Technology）发起征集 AES（Advanced Encryption Standard）算法的活动，目的是为了确定一个安全性能更好的分组密码算法用以取代 DES。AES 的基本要求是比三重 DES 快并且至少与三重 DES 一样安全，分组长度为 128 位，密钥长度为 128 位、192 位、或 256 位。1998 年 8 月 20 日，NIST 公布了满足要求的 15 个 AES 的候选算法。2000 年 10 月 2 日，美国商务部宣布 AES 的最终评选结果，比利时密码专家 Joan Daeman 和 Vincent Rijmen 提出的“Rijndael 数据加密算法”最终获胜。2001 年 11 月 6 日，NIST 正式公布高级加密标准 AES，并与 2002 年 5 月 6 日正式生效。AES 的安全性能是良好的，至今还没有发现 AES 的明显缺点，也没有找到明显的安全漏洞。

在 AES[9-11]中，各种运算是以字节为单位来进行处理的，AES 加密过程和解密过程中间各步的结果称为一个状态，每个状态也是 128 位。状态可由 4 行

Nb 列的字节矩阵来表示。同样密钥也由 4 行 Nk 列的字节矩阵来表示，Nk 为密钥的长度除以 32 得到，取 4、6、8。图 3 所示为状态与密钥，图 4 表示 AES 轮变换数目 Nr 与密钥长度 Nk 的关系。

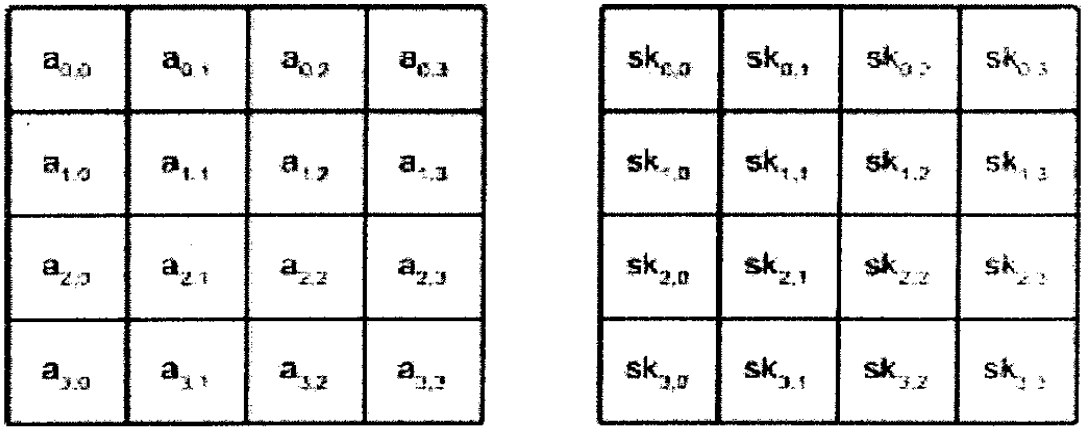

图 3：状态与密钥

Nr	$Nk = 4$	$Nk = 6$	$Nk = 8$
$Nb = 4$	10	12	14

Number of rounds (Nr)

图 4：AES 轮变换数目 Nr 与密钥长度 Nk 的关系。

AES 加密过程中的轮变换由字节替换变换（SubBytes Transformation）、行移位变换（Shift Rows Transformation）、列混合变换（Mix Columns Transformation）、轮密钥加法变换（AddRound Key Transformation）组成，其伪 C 代码如下：

```
Round(State, RoundKey){
    ByteSub(State);
    ShiftRow(State);
    MixColumn(State);
    AddRoundKey(State, RoundKey);
}
```

在最后一轮中，省去 *MixColumn* 变换。其中字节替换变换是针对状态中的每个字节独立地进行非线性字节替换的操作，它可以通过表循环来处理，此表被称为 S

盒。在行移位变换中，状态中的行循环移位不同的偏移量，第零行不移位，第一行移动一个字节，第二行移动两个字节，第三行移动三个字节。在列混合变换中，状态中的列被认为是有限域$GF(2^8)$中的多项式，以x^4+1为模乘固定的多项式$c(x)=03'x^3+01'x^2+01'x+02'$，其矩阵表达为：

$$\begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix}。$$

在轮密钥加法变换中，轮密钥通过简单的逐位异或操作应用到状态中。轮密钥由密钥通过密钥分发得到，它由密钥扩展和轮密钥选择组成。扩展的密钥是四个字节字的线性排列表示为$W[Nb*(Nr+1)]$，最前面的Nk个字包含密钥。如果Nk小于等于6，密钥扩展可以由下面的伪C代码表示：

```
KeyExpansion(byte Key[4*Nk], word W[Nb*(Nr+1)]){
  for(i=0;i<Nk;i++)
    W[i]=(Key[4*i], Key[4*i+1], Key[4*i+2], Key[4*i+3]);
  for(i=Nk;i<Nb*(Nr+1);i++){
    temp=W[i-1];
    if(i%Nk==0)
    temp=SubByte(RotByte(temp))⊕Rcon[i/Nk];
    W[i]=W[i-Nk]⊕temp;
  }
}
```

AES解密过程由逆行移位变换（Inverse Shift Rows Transformation）、逆字节替换变换（Inverse Sub Bytes Transformation）、逆轮密钥加法变换（Inverse Add Round Key Transformation）、逆列混合变换（Inverse Mix Columns Transformation）组成。AES的加密算法流程图如图5所示。

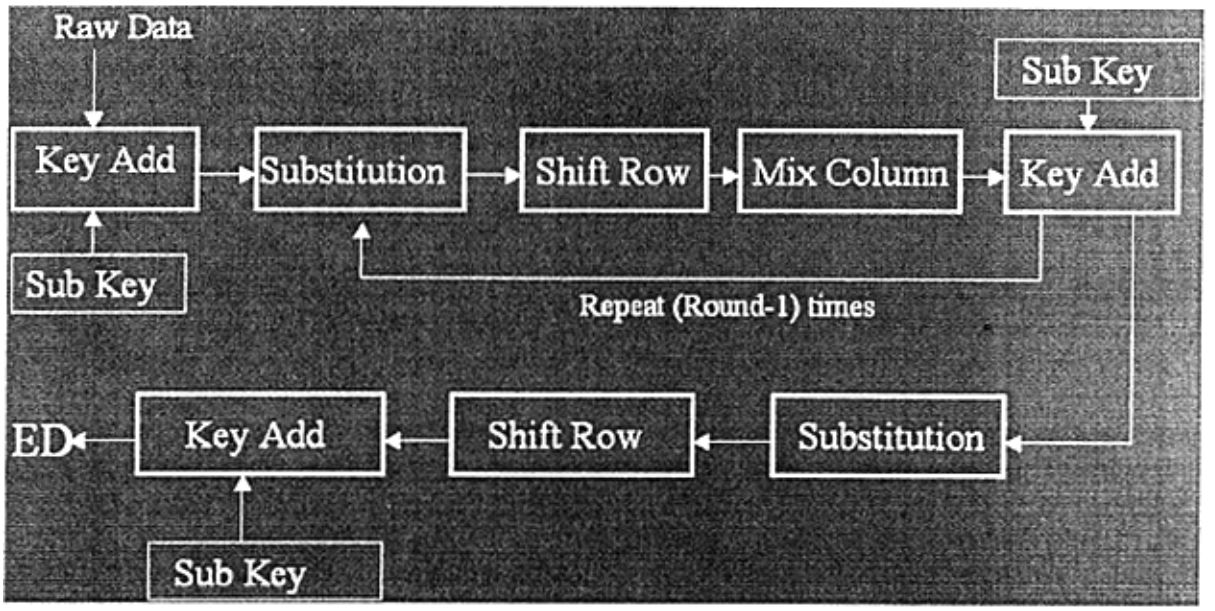

图 5：AES 加密算法流程图

2.3 RSA 算法

1976 年，W. Diffie 和 M. E. Hellman 发表了“密码学的新方向(New Directions in Cryptography)”疑问，提出了公钥密码学(public-key cryptography)的思想。在公钥密码体制(public-key cryptosystem)中，加密密钥和解密密钥是不一样的，加密密钥可以公开传播而不危及密码体制的安全性。RSA 密码系统是 R. L. Rivest、A. Shamir 与 L. M. Adelman 于 1978 年提出的公钥密码系统[12]，在公钥密码系统里，每一个使用者都有一对钥匙：（加密密钥简称公钥、解密密钥简称密钥），公钥顾名思义就是可公开的密钥，而密钥则需使用者保密。

在 RSA 密码系统中，n 是 RSA 密码系统中的模，它是两个很大的质数 p 与 q 的积 n=pq，公钥 e 为一个小于 $\phi(n)=(p-1)(q-1)$ 且满足 $\gcd(e,\phi(n))=1$ 的正整数，而密钥 d 为正整数且与 e 关于模φ(n)为逆，即

$$de \equiv 1 \pmod{\phi(n)}$$

对明文 M 加密时，计算

$$c = M^e \bmod \phi(n)$$

c 为密文。解密时，则计算

$$c^d \bmod n = (M^e)^d \bmod n = M$$

将明文 M 复原。若是做数字签名，则签名者用其密钥将要签署的明文 M 以密钥 d 作指数同余运算，成为签名 s：

$$M^d \bmod n = s$$

而验证签名时，则检查

$$s^e \bmod n = (M^d)^e \bmod n = M \quad 。$$

在 RSA 密码系统中信息 M 的大小被限制在 1 与 n 之间。RSA 密码系统的安全性是建立于模数 n 的素分解问题的难解性。

在 RSA 密码系统中最主要的计算为指数运算，即 M^e mod n，一般常用的算法如图 6 所示。

输入：M, n, e = $(e_{w-1}\cdots e_2\ e_1\ e_0)$
输出：S = M^e mod n

```
1    Let S = 1
2    FOR k = w-1 downto 0
3        Let S = S² mod n
4        IF (e_k is 1 ) then Let S = (S•M) mod n
5        ENDIF
6    ENDFOR
7    Return S
```

图 6：计算 M^e mod n 算法

图 6 是一个从左至右、平方与乘法的算法，其中 M 为需要加密的信息，n 为 RSA 系统的模，指数 e 为 RSA 系统的密钥，假设指数 e 有 w 位$(e_{w-1}\cdots e_2\ e_1\ e_0)$，算法的输出 S 是计算 S = M^e mod n 的结果。

2.4 RSA with CRT 算法

为了有效的计算，一般会用中国余数定理(Chinese Remainder Theorem, CRT)来计算，即所谓 RSA with CRT 密码系统。RSA with CRT 密码系统是以下列方式计算：

首先找出 a 与 b 两个数，其中 a 满足

$$a = 1 \bmod p \text{ 且 } a = 0 \bmod q$$

的条件，而 b 满足

$$b = 0 \bmod p \text{ 且 } b = 1 \bmod q$$

的条件。则 S 的计算方式为

$$S = (a\ S_1 + b\ S_2) \bmod n$$

其中 $S_1 = M^d \bmod P$ 且 $S_2 = M^d \bmod q$ 。

由于 S 是 S_1 与 S_2 的线性组合，而且模比较小，因此计算比较有效。

3. 智能卡物理攻击技术

3.1 故障分析攻击技术

1996 年 9 月份，三位 Bellcore 的信息安全专家 Boneh、DeMillo 与 Lipton 提出一种被称为“故障分析”(hardware fault cryptanalysis)的攻击技术[3]，它是利用智能卡的物理特性与被攻击的密码系统本身的特性，设计出的一种极具效率的密码攻击技术。继三位 Bellcore 的信息安全专家发现故障攻

击法后，许多与故障攻击相关的研究成果陆续于 1996 年底到 1997 年初被提出[13-18]，这些攻击技术都有其特殊的硬件故障假设，依其不同的故障假设不同所具有的攻击能力亦相距甚远。

故障攻击技术是指攻击者利用智能卡因故障而产生的错误输出，比较正确输出与错误输出而推导出密钥的技术。

故障攻击可分为暂时性故障模型与永久性故障模型。在暂时性故障模型里，攻击者可以利用不规则时钟脉冲、幅射、瞬间高电压等技术改变缓存器或挥发性储存器上的某个位。在永久性故障模型里，攻击者可以利用紫外线或 X 射线改变清除 EEPROM 上的某个位；或使用电子探针设定或清除 EEPROM 上的某个位，或使用 laser cutter 破坏或建构 ROM 上的某个位，或者使用 cutting wire 切除智能卡内的线路。

我们以 RSA with CRT 密码系统来说明故障攻击技术。 文献[3] 描述了如何用暂时性故障模式攻击 RSA with CRT 密码系统：攻击者以相同的信息产生两次签名，一次为正确签名，另一次为不正确的签名，攻击者依据正确与不正确的签名推导出模 n 的一个质因子，也就将模数 n 因子分解，从而破解了 RSA with CRT 密码系统。

假设信息 M 及其使用 RSA with CRT 密码系统的签名为

$$S = (a\ S_1 + b\ S_2)\ \bmod n$$

另外假设信息 M 在使用 RSA with CRT 密码系统计算签名时，在计算 S_1 有一些情况发生，产生暂时性故障，而计算出错误结果 $\hat{S}_1$，并导致错误的签名

$$\hat{S} = (a\ \hat{S}_1 + b\ S_2)\ \bmod n$$

观察正确的签名 S 与不正确的签名 $\hat{S}$，由于

$$S - \hat{S} = (a\ S_1 + b\ S_2) - (a\ \hat{S}_1 + b\ S_2) = a\ (S_1 - \hat{S}_1),$$

假设 $(S_1 - \hat{S}_1)$ 不可被 p 除尽，那么可以得出

$$\gcd((S - \hat{S}),\ n) = \gcd(a(S_1 - \hat{S}_1), n) = q,$$

从而成功地将模数 n 因子分解。上述因子分解的成功是建立在 $(S_1 - \hat{S}_1)$ 不可能被 p 整除的条件上，文献[3]中还指出通过如下的推导不需要上述条件也可将模数 n 因子分解，

$$\gcd(M - \hat{S}^e,\ n) = q$$

其中 e 为验证签名时用的公钥。也就是说，只需知道信息 M 和得到不正确的签名，就可以将模数 n 因子分解了。

总之，根据正确与不正确的签名就可推导出模数 n 的其中一个质因子，也就将模数 n 因子分解了，从而破解了 RSA with CRT 密码系统。必须注意的是，前述的错误攻击技术是不管什么错误发生，不管发生错误几次，只需要在计算 S_1 或 S_2 时发生错误并导致不正确的签名即可。

前述的例子里，由于一个小错误通常会破坏整个系统，因此一般还是认为此类故障攻击仅属于理论推导范围。但在文献[1][2]中指出，现实中的黑客团体就是使用属于暂时性故障模型的瞬时脉冲攻击(glitch attack)技术破解付费电视频道，而免费收看付费频道。所谓瞬时脉冲攻击技术是指暂时性的加快时钟脉冲或增强电压，使智能卡产生暂时性的执行错误。这些错误可使智能卡泄露出一些信息出来。我们举例说明瞬时脉冲攻击技术。参考如下的片段程序，这是印出一小段储存器内容的程序。

```
1    b = answer_address
2    a = answer_length
3    if ( a == 0 ) goto 8
4    transmit( *b )
5    b = b + 1
6    a = a - 1
7    goto 3
8    …
```

假如在执行第 3 行指令时，时钟脉冲突然暂时性的增快，使得在尚未获得比较结果前就开始执行第 4 行指令，因此不管 a 值是多少，用此技巧接着都会执行第 4 行指令，重复此技巧，整个储存器的内容即被泄露。

由于不同信号路径所需的延迟时间不同，藉由巧妙的控制瞬时脉冲，攻击者就可使用瞬时脉冲攻击技术读出信息，跳过安全检查而破解密码算法。

继三位 Bellcore 的信息安全专家于 1996 年发表故障攻击技术后，Biham 与 Shamir 于 1996 年发表了使用暂时性故障模型破解 DES 算法[4]，紧接着 Shamir 于 1996 年底发表了使用暂时性故障模型破解 DES 算法[13]，随后陆续发表了使用故障攻击技术于其它的密码算法上[14-18]。[14] 是针对随机数产生器，[15] 是针对 RSA with CRT 算法攻击，[16] 是攻击使用 Montgomery 算法的 RSA 密码系统，[17] 提出故障的模型并实现了 RSA 密码系统的攻击，[18] 实现了非对称式椭圆曲线加解密系统的攻击。

3.2 时间攻击技术

1995 年 9 月 29 日，Paul Kocher 首次提出了密码系统的执行时间与密钥相互关联的思想，并进一步在 1996 年的 Crypto 会议上发表了时间攻击的论文[5]，该文献指出攻击者如何分析时间特性以推导出密钥。继 Kocher 发现时间攻击后，在 UCL 的 Crypto 研究团体，于 1998 年发表了在自行研制的 CASCADE 智能卡上实现时间攻击的论文 [19]，同年 Handschuh 与 Heys 发表了实现时间攻击于 RC5 算法上的论文[20]，随后陆续发表的实现时间攻击于其它算法上[21-25]。自 Kocher 发表时间攻击后，人们才开始注意到目前使用的产品与协议(如 SSL)存在被攻击的危险。

密码算法的执行时间常会因输出资料的不同而有差异，这主要是因为密码算法会依不同的输入资料执行不同路径所致，这也使得攻击者可以借助分析收集到的执行时间信息以推导出密钥。

RSA 密码系统的基本运算为指数与模运算，且是被用来对信息做加密或签名。一般而言，实现 RSA 密码系统是使用图 6 的算法。图 6 的算法是一个从左至右、平方与乘法的算法，其中 M 为需要加密的信息，n 为 RSA 系统的模，指数 e 为 RSA 系统的密钥，假设密钥 e 有 w 位($e_{w-1}\cdots e_2\ e_1\ e_0$)，其算法的输出 S 是计算 $S = M^e \bmod n$ 的结果。

从图 6 的算法中，Kocher 观察到一个重要现象。在该算法中第 4 行的条件指令会影响执行路径。在循环的第 k 个轮次，假设密钥的第 k 位 e_k 为 1，则会执行第 3 行与第 4 行程序。类似的，假设密钥的第 k 位 e_k 为 0，则仅会执行第 3 行程序。也就是说该算法中第 4 行条件指令的成立与否，与密钥的值有关。因此，计算 $M^e \bmod n$ 的总时间，即算法执行时间，会因密钥的值的不

同而有所不同。

假设攻击者依据图 6 的算法，能够观察到该算法不同轮次的执行时间，且能比较该算法不同轮次执行时间的差异，该攻击者就可能推导出指数 e。当 RSA 密码系统应用于签名时，上述的技术就可能推导出签名者的密钥。

在本节中我们首先介绍 Kocher 的时间攻击的环境，接下来描述时间攻击的先决条件，最后介绍时间攻击的算法。

实施时间攻击技术的环境如图 7 所示，攻击者的计算机与被攻击的智能卡通过读卡机相连通。假设在智能卡上执行图 6 的算法。攻击者以其计算机用时间攻击技术破解智能卡上的密钥----M^e mod n 的指数 e。

图 7 ： 实施时间攻击技术的环境

依据图 7 的攻击环境，Kocher 的时间攻击技术必须具备以下条件：

1. 攻击者熟悉智能卡的软硬件设备，特别是熟悉智能卡实现 RSA 密码系统算法的细节。攻击者须构建一套与智能卡几乎一样的软硬件设备（一般卖智能卡的公司都会卖该智能卡的运行环境，该运行环境是可以仿真的）。
2. 攻击者能精确的记录从送出欲加密信息到接收到已加密信息的总交易时间。
3. 攻击者能精确的记录 RSA 密码算法每个轮次的执行时间。

依据图 7 的攻击环境与上述的前题条件，Kocher 攻击图 6 算法的思想如下：

1、收集整个执行时间

2、收集累积的轮次执行时间

3、决策。

在第 1 步，攻击者在计算机上产生 L 个信息(M_1、M_2、M_3、•••、M_L)。接着把此 L 个信息一个一个的传送到智能卡上做加密，智能卡会把加完密后的信息传回到计算机上。攻击者记录每个信息加密时间，从送出信息至接收到加密后信息的总时间。假设信息 M_i 的总加密时间为 T_i。因为攻击者是记录从送出信息至接收到加密后信息的时间，因此

T_i = 读卡机送收时间 (e) + 智能卡上的加密时间(tl_i)。

第 2 步，攻击者在智能卡仿真环境上执行图 6 所示的算法，开始猜测密钥 e，每次猜一位，一直至整个密钥 e 被猜出来为止。假设对于信息 M_i ，执行循环的第 k 个轮次算法的执行时间为 $tl_{k,i}$，那么整个算法的执行时间为

$$\sum_{k=0}^{k=w-1}(tl_{k,i})$$

假设已知指数的(w-f-1)位(e_{w-1}•••e_{f+2} e_{f+1})，现在要猜测 e_f 。攻击者针对每个信息 M_i，记录计算执行语句

FOR j = w-1 downto f

所需的时间 $\sum_{k=f}^{k=w-1}(tl_{k,j})$。

最后一步是判断 e_f 到底是 0 还是 1？首先计算 Y_0 与 Y_1 两个集合，这两个集合的元素是针对每个信息的全部执行时间减去该信息的累积轮次执行时间，亦即

$$e + \Sigma_{k=0}^{k=w-1}(tl_{k,i}) - \Sigma_{k=f}^{k=w-1}(tl_{k,i}) = e + \Sigma_{k=0}^{k=f-1}(tl_{k,i})$$

其间的差异为 Y_0 集合的假设条件是 e_f 为 0，而 Y_1 集合的假设条件是 e_f 为 1。由于每个信息是相互独立，因此 Y_0 与 Y_1 集合的变异数为

$$Var(e) + (w-f)\ Var(tl)$$

假设 (w-f-1) 个位中有 c 个位猜错，其变异数为

$$Var(e) + (f+2c)Var(tl)$$

正确的猜测会依 Var(tl) 的大小减少变异数，而不正确的猜测会依 Var(tl) 的大小增加变异数。据此来判断 e_{f-1} 是 1 还是 0。换句话说，若 Y_0 集合的变异数减小则猜测 e_f 为 0，若 Y_1 集合的变异数减小则猜测 e_f 为 1。

由于测量的计算时间并不很精确，导致计算 Y_0 与 Y_1 的变异数会有一些出入，因而每次对 e_f 的猜测并不一定正确，因此必须修正错误。若猜错 e_f 的值时，则会有依 Var(tl) 的大小增加变异数的特性。因此当发现 Y_0 与 Y_1 的变异数持续扩大时，即可推论前面的猜测是错误的，此时即必须后退几个循环重新猜测，这被称为错误修正技术。虽然错误修正技术会增加储存器与处理的需求，但可大大降低攻击样本数。

依据 Kocher 所提出的时间攻击的观念，许多人针对不同的算法实现了攻击[20-25]。[20] 是针对对称式加解密算法 RC5 实施攻击，[21] 是针对对称式加解密算法 DES 进行安全设计，[22] 是攻击使用中国余数定理以及 Montgomery 算法的 RSA 密码系统，[24] 攻击 AES ，[25] 是攻击 RSAREF 2.0 密码系统。

3.3 简单能量攻击

大多数现代的智能卡是由晶体管半导体来制成。电子束穿越硅基底至晶体管栅极时，会消耗能量并产生电磁幅射。简单能量攻击是一种通过直接观察密码运算时的能量消耗来获取密钥的技术。

1997 年，在 Cryptography Research 公司的 Paul Kocher、Joshua Jaffe 与 Benjamin Jun 首次提出能量攻击的概念，进一步在 1998 年把其研究结果公布于其公司网站上[6]，并于 1999 年发表在 Crypto 会议上[7]。下面将以文献[7]中的例子来说明简单能量攻击。例如一毫秒采样一次，若频率为 5MHz，那么每秒就可以产生 5000 个采样值。图 8 为智能卡执行 DES 运算的能量消耗图，从图 8 中可以清楚地看到 DES 的 16 个轮次。图 9 所示的是 DES 的第 2 轮与第 3 轮运算更详细的能量消耗图。从图 9 中可以清楚的看到在第 2 轮 28 位的 DES 密钥寄存器 C、D 有一次旋转的发生（如左边箭头所示），而在第 3 轮 28 位的 DES 密钥寄存器 C、D 有二次旋转的发生（如右边箭头所示）。

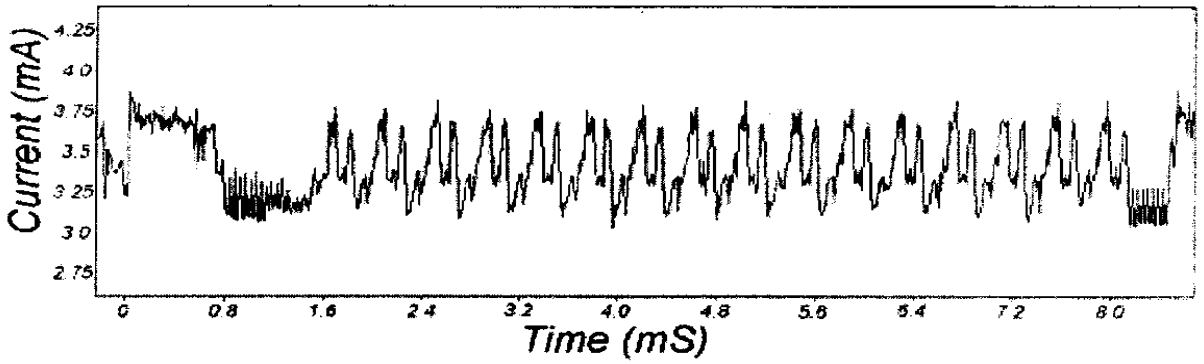

图 8：整个 DES 运算的能量消耗图[7]

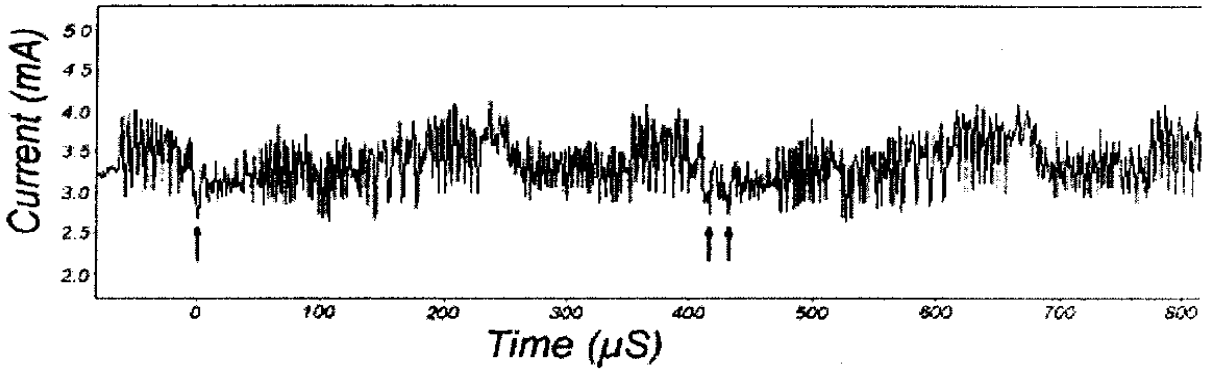

图 9：第 2 轮与第 3 轮 DES 运算的能量消耗剖面图[7]

若采样频率更高则能获得更详细的 DES 运算信息，图 10 所示的是更高分辨率的两个能量消耗图，每个包含 7 个时钟周期，频率为 3.5714MHz。不同时钟周期间的差异主要来自于不同的微处理指令能量消耗的差异，值得注意的是，图 10 的上图与下图均为执行相同一段程序的能量消耗图，不同是在第 6 个时钟周期的条件转移指令结果为上图真而下图假，这个结果也显示在第 6 个时钟周期的能量消耗有明显的不同。简单能量攻击技术就是依靠直接观察能量消耗图而推导出密钥的。

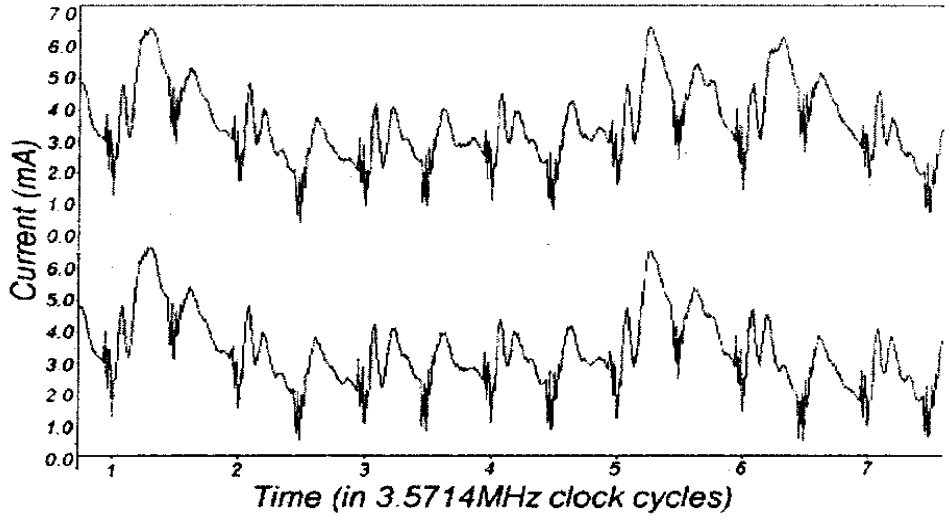

图 10：单个时钟脉冲的能量消耗剖面图[7]

简单能量攻击技术也可以用于指数的计算上面，图 11 为计算 RSA 签名的部份能量消耗剖面图，图 11 中的 9 个尖峰为平方运算与乘法运算的开始点。由于乘法运算比平方运算多一个加载的动作，表现出来就是尖峰比较宽。根据图 6 所示的 RSA 算法，若密钥的第 i 位为 0 则执行平方运算，反的则执行平方与乘法运算。依据平方较窄尖峰、乘法较宽尖峰的推论，因此获得若密钥的第 i 位为 0 则出现较窄尖峰，而若密钥的第 i 位为 1 则出现较窄尖峰后紧随较宽尖峰的结论，根据图 11 的能量消耗图可推出密钥的 5 位为：00111。

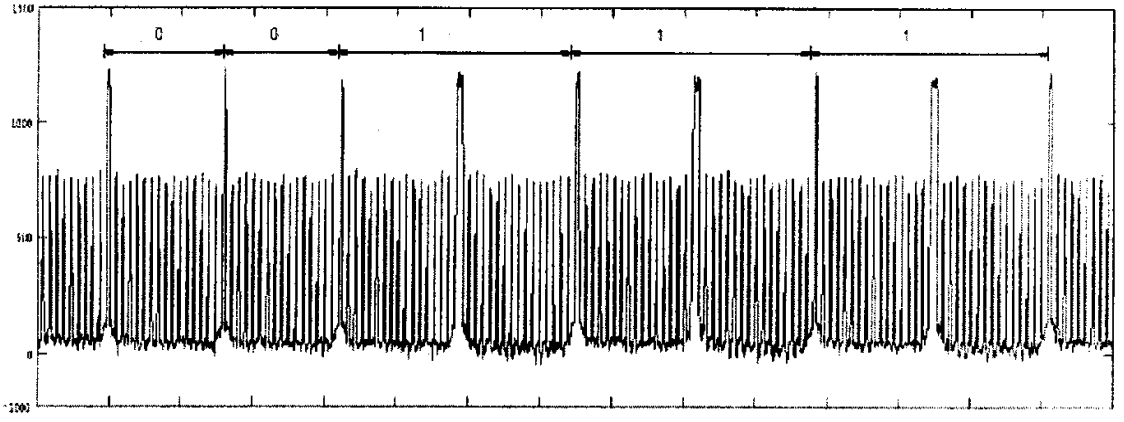

图 11：部份的 RSA 签名能量消耗图[22]

要实现简单能量攻击，首先必须执行密码算法，在执行时同步记录其能量消耗。攻击者如果知道执行指令及其相对应能量消耗图的所在位置，就可以很快的获得有用的信息。换言之，简单能量攻击技术是建立在特定指令及其相对应的能量消耗图所在位置的基础上，因此攻击者需要清楚密码算法的实现。

由于简单能量攻击技术中使用的能量消耗图能显示指令的执行顺序，因此密码算法中的密钥如果与执行路径有关，攻击者就可使用此信息破解密码。密码算法中的步骤包括：

(1) DES key schedule
(2) DES permutation
(3) comparison
(4) multiplier
(5) exponentiation

它们通常与密钥相关且会因此而决定不同的执行路径，是实施简单能量攻击的最佳攻击点，需要引起注意。

继 Kocher 的发现简单能量攻击技术后，在 Motorola 的 Messerges 使用简单能量攻击于数据总线上并得到 Hamming weight 与 transition count 信息，并使用 Hamming weight 与 transition count 信息破解了 DES[26]。

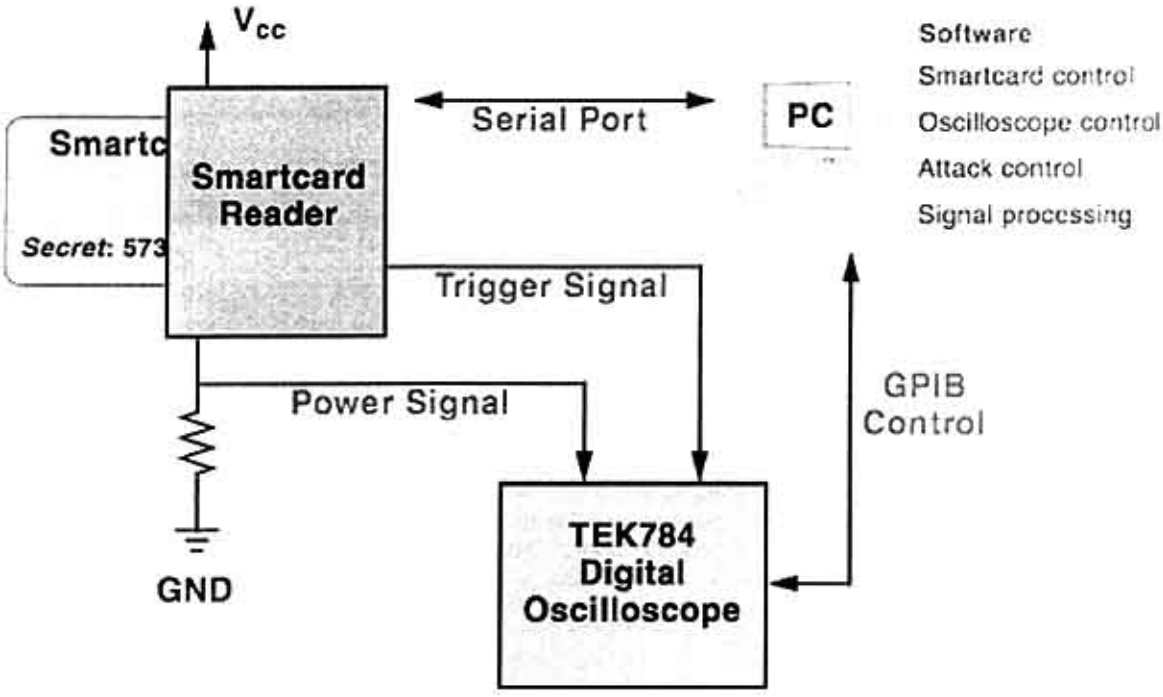

图 12：差异能量攻击原理图

除了指令的执行顺序会影响能量消耗外，指令执行时所处理的数据值也会影响能量消耗，这种影响很小，有时会被噪声或者错误的观测所覆盖，但依靠统计方法还是能看出端倪。差异能量攻击技术是一种以能量消耗图为基础利用统计方法获取密钥的攻击技术。1997 年，Cryptography Research 公司的 Paul Kocher、Joshua Jaffe 与 Benjamin Ju 首次提出差异能量攻击的概念，1998 年将其研究成果公布在公司网站上，并于 1999 年发表在 Crypto 会议上。图 12 为差异能量攻击技术的原理图，下面我们将以 DES 为例说明差异能量攻击的概念。

针对 DES 的 DPA

在尚未介绍差异能量攻击技术之前，先来分析 DES 的一些流程。图 13 为 DES 的密钥与 16 轮次每轮所使用的子密钥的关系图，64 位的密钥(KEY)经过 PERMUTED CHOICE 1 的重新排列后得到各 32 位的 C_0 与 D_0，C_0 与 D_0 各自循环左移一位得到 C_1 与 D_1，C_1 与 D_1 组合成 64 位再经过 PERMUTED CHOICE 2 的重新排列后得到第 1 轮使用的但只有 48 位的子密钥 K_1。C_1 与 D_1 各自循环左移 1 位后得到 C_2 与 D_2，C_2 与 D_2 组合成 64 位再经过 PERMUTED CHOICE 3 的重新排列后得到第 2 轮使用的 48 位子密钥 K_2。依此类推，可以得到各轮使用的子

密钥。注意到 C_i 与 D_i 要循环左移几位后得到 C_{i+1} 与 D_{i+1}，如表 1 所示。

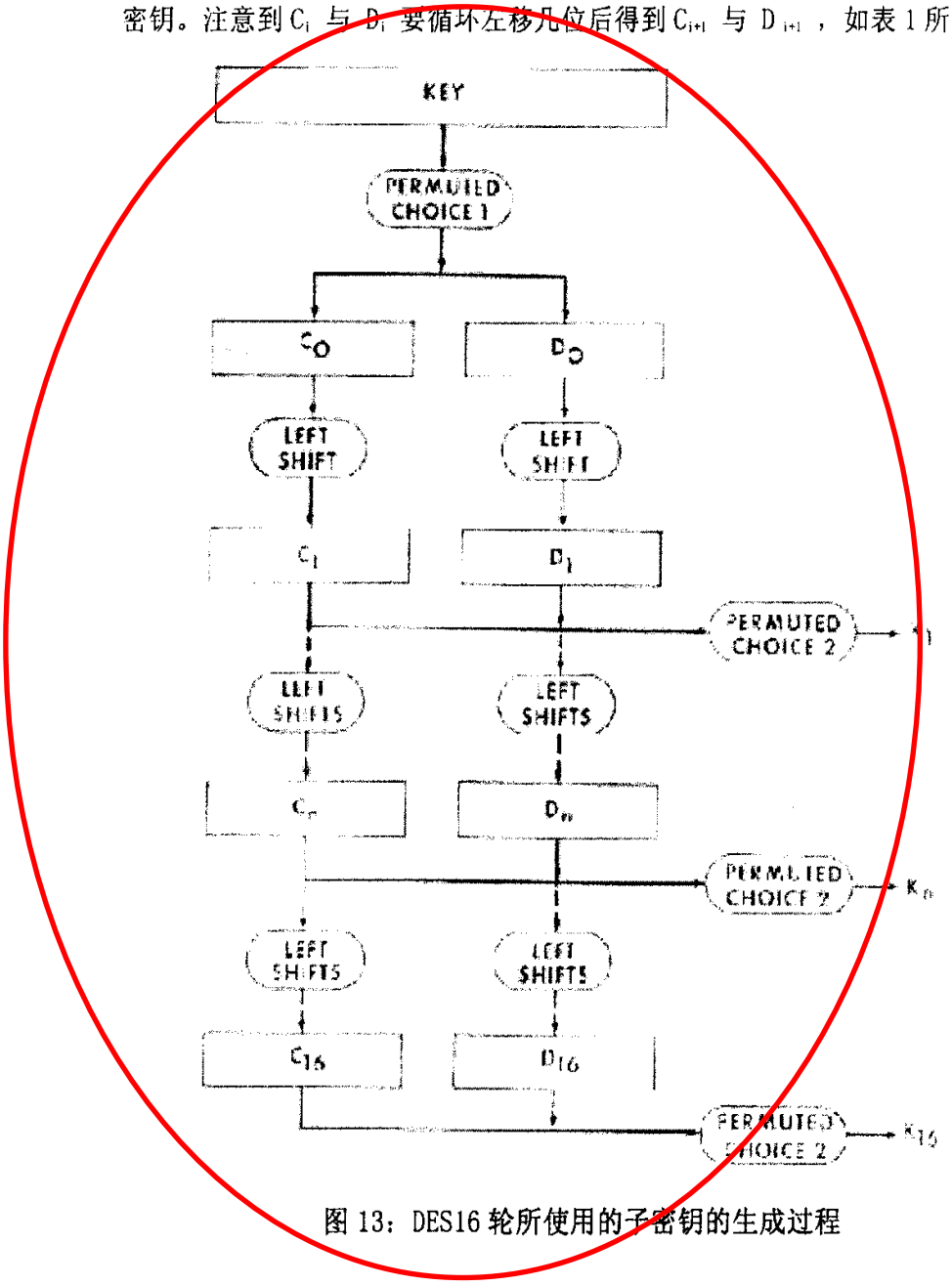

图 13：DES16 轮所使用的子密钥的生成过程

轮	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
循环左移	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

表 1：各轮的循环左移位

由于 PERMUTED CHOICE 2 为 64 位转换至 48 位的固定表格，因此若子密钥 K_i 为已知，反推可知 C_iD_i 64 位中的 48 位。而 C_i 与 D_i 是由 C_0 与 D_0 各自循环左移 1 位后得到，以相反方向循环移位后反推，C_i 与 D_i 各自循环右移 1

位即可得到 C_0 与 D_0，因此假设 C_i 与 D_i 已知，就可轻易地知道 C_0 与 D_0。既然 C_0 与 D_0 为已知，而 PERMUTED CHOICE 1 为固定表格，也就得到密钥。因此若 48 位的子密钥 K_i 为已知，经反推 PERMUTED CHOICE 2 、循环右移、反推 PERMUTED CHOICE 1 后，至少可得到 64 位密钥中的 48 位。获得 64 位密钥中的 48 位的后，其它的位可由穷举法得到。差异能量攻击技术正是藉由破解 DES 的子密钥 K_i 来破解 DES 密钥的。

在 16 轮的每一轮中，DES 数据加密算法执行 8 个 S 盒查找操作，每个 S 盒的输入为 6 位密钥与 R 寄存器中的 6 位的异或值，产生 4 位输出值，共得到 32 位输出值再与 L 寄存器的值异或。图 14 所示的是轮函数 f ，轮函数的输入为 32 位的 R_i 与 48 位的子密钥 K_i，32 位的 R_i 经过 E 扩展变换后得到 48 位的 T1，48 位的 T1 与 48 位的子密钥 K_i 做异或运算后得到 48 位的 T2，48 位的 T2 以 6 位为一组成分成 8 组：

T2={T2S1、T2S2、T2S3、T2S4、T2S5、T2S6、T2S7、T2S8}

每一组 T2Si 寻找对应的盒 Si，映射出 4 位的 T3Si，8 个 4 位的 T3Si 再重新组合成 32 位的 T3：

T3={T3S1、T3S2、T3S3、T3S4、T3S5、T3S6、T3S7、T3S8}

32 位的 T3 经过置换函数 P 换位处理后得到 32 位的轮函数的输出。

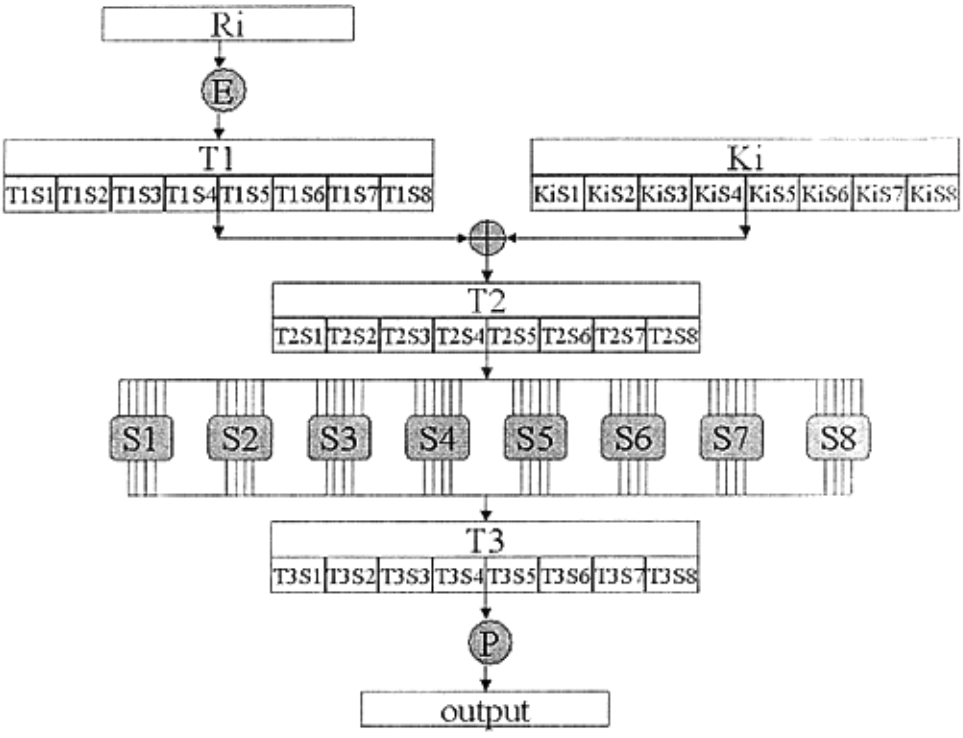

图 14：DES 运算中的轮函数

差异能量攻击技术正是通过破解 DES 的子密钥 K_i 而破解 DES 密钥的。一般是通过破解子密钥 K_1 或子密钥 K_{16} 来破解 DES 密钥，由于道理相似，在此以

破解子密钥 K_{16} 为例来说明差异能量攻击。差异能量攻击技术分为资料的收集阶段、分类和分析阶段。

首先谈资料的收集阶段：设 PT_i（Power Consumption Trace）为明文 PTI_i 经过 DES 加密后得到密文 CTO_i 的整个流程的能量消耗图。收集 CTO_i 及其相对应的能量消耗图 PT_i。

由于 48 位的子密钥 K_{16} 有 2^{48} 种组合，很难破解，因此把 48 位的子密钥 K_{16} 以 6 位一组分成 8 组：

K_{16}={ K16S1、K16S2、K16S3、K16S4、K16S5、K16S6、K16S7、K16S8}

若一次能破解一组，那么一次仅有 2^6=64 种组合，要破解就简单多了，差异能量攻击技术就是先破解 K16S1，再破 K16S2，以此类推，最后把子密钥 K_{16} 完全破解。

其次是资料的分类阶段，如图 15。首先假设 K16S1=000000 ，接下来确定观察位元 D，假设 D 为 T3S1 的最左边的位，注意到观察位元 D 与要猜测的子密钥 K16S1 有关联，将前一阶段收集的资料依观察位元 D 的值分成两类：

P1(D=1) 与 P0(D=0)

以统计方式分析这两类资料以确定假设 K16S1=000000 是否正确，若是则找到 K16S1 子密钥，若不是则重新假设 K16S1 新值，重复分析一遍。由于 K16S1 仅有 64 种组合，因此可以很快地找到 K16S1。

接下来看如何依据观察位元的值将收集资料分成 P1 与 P0 两集合。假设观察位 D 为 T3S1 最左边的位，假设 CTO_i 为第一阶段收集的密文，PT_i 为明文 PTI_i 经过 DES 加密得到 CTO_i 的整个流程的能量消耗图。参考图 1 中 的 DES 流程图，注意到第 16 轮经过运算

$$L_{16}=R_{15}$$
$$R_{16}=L_{15} \quad f(R_{15}, K_{16})$$

why???

得到 PREOUTPUT $(R_{16}L_{16})$，PREOUTPUT 再经过 INVERSE INITIAL PERM 的重新排列后得到密文 OUTPUT。了解 DES 加密的第 16 轮得到密文的流程后，我们可以用已知的密文 CTO_i 与假设已知的子密钥 K16S1=000000 推算出观察位 D 是 1 还是 0，从而将能量消耗图 PTi 归入 P1 集合或 P0 集合。

由于 CTO_i 为已知，而且 INVERSE INITIAL PERM 为固定表格，因此可以轻易地反推出 PREOUTPUT 而使 R_{16} 与 L_{16} 也为已知。由于 $L_{16}=R_{15}$ ，因此 R_{15} 也为已知。轮函数 $f(R_{15}, K_{16})$ 的输入 R_{15} 为已知，子密钥 K_{16} 的部份 K16S1=000000 也为已知。由于 32 位的 R_{15} 经过 E 表格的重新排列后得到 48 位的 T1，因此 T1 也为已知。接下来是 48 位的 T1 与 48 位的子密钥 K_i 做异或运算得到 48 位的 T2，由于异或运算是位运算，T1 与子密钥 K_i 的最左边 的 6 位 K16S1 均为已知，因此 T2 的最左边的 6 位 T2S1 可知。由于观察位元 D 为 T3S1 的最左边的位，T3S1 是 T2S1 经 S 盒 S1 映射出的 4 位，T2S1 已知，T3S1 也就已知。既然 T3S1 已知，也就知道观察位元 D 是 1 还是 0。观察位元 D 是 0 ，就将能量消耗图 PT_i 归入 P0 集合；观察位元 D 是 1，就将能量消耗图 PT_i 归入 P1 集合。重复上述过程，把第一阶段收集到的所有密文 CTO_i 依上述反推程序得到观察位元的值，然后依据观察位元的值将能量消耗图 PT_i 归入相应的集合。

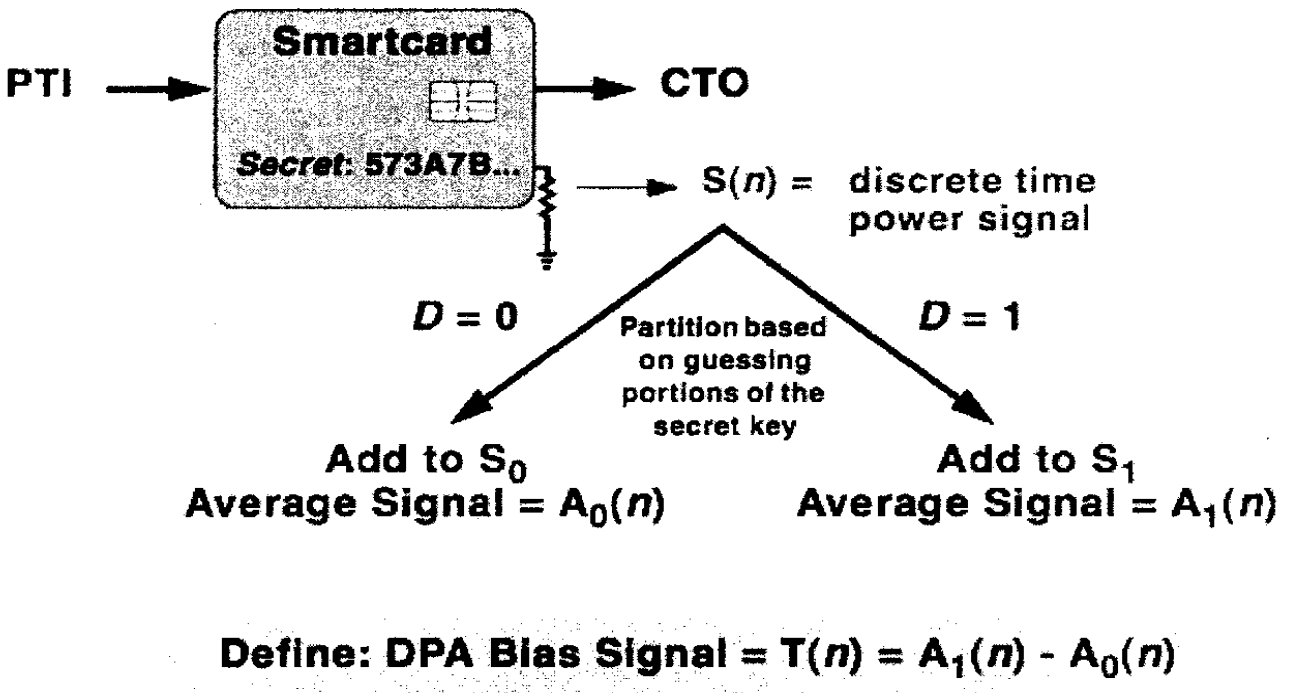

图 15：DPA 的资料分类阶段

最后是资料分析阶段，从资料分类阶段获得的两个能量消耗图集合：*P0* 集合与 *P1* 集合。首先计算 *P0* 集合能量消耗图的平均值，得到在观察位元 D 为 0、子密钥 K16S1 为 000000 的条件下 DES 加密整个流程的平均能量消耗图 P0 。同样的计算可得 *P1* 集合能量消耗图的平均值，得到在观察位元 D 为 1 、子密钥 K16S1 为 000000 的条件下 DES 加密整个流程的平均能量消耗图 P1。接下来计算平均能量消耗图 P0 与平均能量消耗图 P1 的差异，从而得到在子密钥 K16S1 为 000000 的条件下 DES 加密整个流程的差异能量消耗图 P0P1 000000。

假设不同的子密钥值重复分类阶段与分析阶段，得到相应的差异能量消耗图。设子密钥 K16S1 为 i 时得到的差异能量消耗图为 P0P1 I，由于子密钥 K16S1 仅有 64 (0 到 63)种组合，因此可得到 P0P1 000000、P0P1 000001、…、与 P0P1 000063 共 64 个差异能量消耗图：

子密钥 K16S1 = 000000 其相对应的差异能量消耗图 P0P1 000000，
子密钥 K16S1 = 000001 其相对应的差异能量消耗图 P0P1 000001，
…
…
子密钥 K16S1 = 000063 其相对应的差异能量消耗图 P0P1 000063。

观察这 64 个差异能量消耗图，如果是正确的子密钥 K16S1，那么差异能量消耗图在计算观察位元的地方会有比较突出的差异表现出来，所谓比较突出是相比较而言。图 16 为在文献[7]中所举的例子，最上一条为执行 DES 加密整个流程的平均能量消耗图，其余 3 条为各个不同的子密钥条件下的差异能量消耗图。比较第 2、第 3 与第 4 条差异能量消耗图，在第 2 条约 70~80 us 的地方有明显的突出，因此与该图相对应的子密钥即为正确的子密钥。

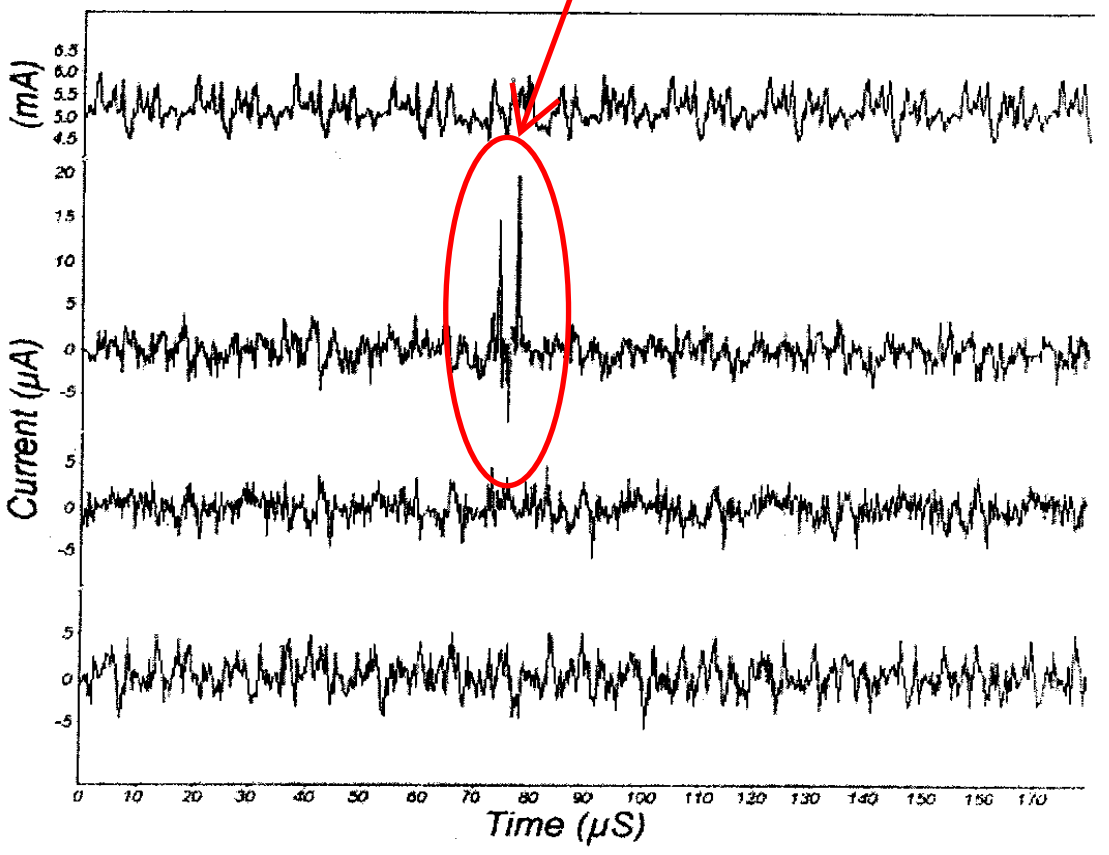

图 16：能量消耗剖面图[7]

继 Kocher 发表差异能量攻击技术后，许多人针对不同的加密算法做了实验[27-29]。[27] 是攻击 AES 的 key schedule，[28] 是针对非对称式加解密算法的指数运算做攻击，[29] 是攻击椭圆曲线非对称式加密算法。Kocher 等针对 DES 加密的智能卡实施 DPA 攻击作了一系列的实验 (采样点 N = 1,300, 时间 ≈ 1 hour, 设备代价< $10K.)。滤波技术可以使 DPA 攻击更有效。图 17、18 分别表示了 DPA 攻击的信噪比及其所需的采样率。

$$\text{Theoretical Voltage SNR} = \frac{\sqrt{N}\,\varepsilon}{\sqrt{8\sigma^2 + \varepsilon^2(\alpha m + m - 1)}}$$

N = Number of signals used for an attack

ε = Size of the DPA bias signal

σ = Noise variance in a single power signal

m = Bus width of the processor

α = Percentage of "algorithmic" noise

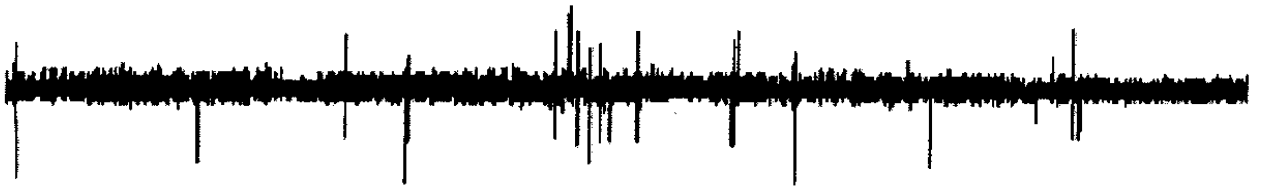

图 17： DPA 攻击的信噪比

1. Solve for *N*:

$$N = \frac{8\sigma^2 + \varepsilon^2(\alpha m + m - 1)}{\varepsilon^2 \cdot SNR^2}$$

2. Determine parameters for a specific smartcard:

$\sigma = 7.5$ mV $\quad \varepsilon = 6.5$ mV $\quad m = 8 \quad \alpha = 0$

3. Assign *SNR*:

$SNR = 0.67$ ← Median for Gaussian Distributed Noise

4. Calculate *N*:

Theoretical Minimal Number of Samples: *N* = 40

图 18：所需的采样点数

针对 AES 的差异能量攻击技术

针对 AES 的 DPA 攻击由针对 S 盒的攻击以及针对轮密钥加法变换的攻击构成。针对 S 盒的攻击方法由 Kocher 等[7]提出，用以攻击 DES 的 S 盒。攻击者先找第一轮子密钥的八位，重复 16 次即可得到所有的第一轮密钥。对于大多数 AES 系统而言，密钥长度为 128 位且第一轮子密钥与密钥相同，如果密钥的长度大于 128 位则第二轮子密钥包含密钥的剩余部分，第二轮子密钥可通过第二个字节替换变化用如下的程序由第一轮子密钥得到，攻击第一轮子密钥的最初 8 位的程序如图 19。

DS1. 攻击者随机产生许多明文。

DS2. 用未知的密钥加密明文并记录相应的能量消耗曲线。

DS3. 攻击者选择第一次字节替换变换输出的一位作为观察位元，它是明文的第一个字节值和第一轮子密钥的第一个字节值的函数。

DS4. 攻击者假定第一轮子密钥的第一个字节的值，从密钥的假定部分和明文，攻击者推断观察位元的值。重复此过程直至取遍第一轮子密钥的第一个字节的可能值。

DS5. 根据观察位元的值，将能量消耗曲线分成两类。

DS6. 攻击者计算每类的平均能量消耗曲线，得到两平均能量消耗曲线的差异能量消耗曲线。如果假定的部分密钥值以及相应的分类是正确的，则当选定的观察位元被处理时两平均能量消耗曲线会显出差异，即在此点处差异能量消耗曲线存在尖峰。

图 19：DPA 攻击第一轮子密钥的最初 8 位的程序

针对轮密钥加法变换的攻击由 chari 等[39]提出，用以攻击 Twofish 加密算法的隐藏操作(whitening)，隐藏操作是一种不改进算法而扩展密钥长度的技术。执行隐藏操作，密钥在算法执行前、后与数据做异或运算。当攻击者实施攻击时，必须考虑到密钥中的“1”翻转了数据中相应位的值，而密钥中的“0”使数据中相应位的值在异或运算后保持不变。如果我们根据与密钥异或运算后数据的位值将能量消耗曲线分成两类，分别计算每类的平均，我们可以得到两类平均曲线的差异。当密钥位为 0 时我们可在差异能量曲线上看到两个同向的尖峰，这是因为数据相应位的差异存在于差异能量曲线上的两点且数据的相应位值并没有改变。如果密钥的位值是 1，则数据的相应位值在与密钥异或前后有不同的值，我们根据与密钥异或运算后数据的相应位值划分能量消耗曲线为两类，则平均能量消耗的差异曲线有两个相互反向的尖峰。通过攻击轮密钥加法变换，攻击者可以找到

第一、第二轮子密钥。如此，攻击者可以找到整个密钥，寻找第一轮子密钥位值的具体程序如图 20。

DK1. 攻击者随机产生许多明文。

DK2. 用未知的密钥加密明文并记录相应的能量消耗曲线。

DK3. 攻击者选择第一轮子密钥的一位，攻击者根据与选择的密钥位元异或运算后数据的位值将明文和相应的能量消耗曲线对分成两类。

DK4. 攻击者分别计算每类的平均，得到两类平均能量消耗曲线以及两类平均能量消耗曲线的差异曲线，如果密钥的位值是“1”，则在差异能量消耗曲线上存在相反方向的两个尖峰；如果密钥的位值是“0”，差异能量消耗曲线上存在相同方向的两个尖峰。

图 20：DPA 攻击寻找第一轮子密钥位值的具体程序

3.5 二阶差异能量攻击

能有效防御一阶差异能量分析的方法未必对高阶差异能量分析有效，将能量泄漏可以描述为基于汉明加权的简单模型，[37]证明了一阶差异能量分析对数据隐藏能实施有效攻击！改进算法使之能有效防御一阶差异能量分析，但二阶差异能量分析对改进算法仍然能实施有效攻击！

定义：n 阶差异能量攻击是利用相应于算法执行过程中得到的 n 个不同的中间变量的能量消耗曲线上的 n 个不同的采样点来进行。

定义：当攻击者通过能量消耗信息理论上能得到密钥的各个位值时我们说 DPA 攻击是有效的！

为了更好地理解二阶差异能量攻击的概念，我们来看一简单的例子，算法 W_1、 W_2 的伪代码如图 21。这些算法起始于结合输入明文 PTI 和密钥，此结合步被称为“隐藏操作”，常用在一些算法的第一步。输入数据的隐藏操作是用 XOR 操作来实现的。算法 W_1 在行 A 执行此 XOR 操作，而在行 A 的异或操作能泄漏密钥的信息因此算法 W_1 对一阶差异能量攻击很脆弱，为了能有效地防御 DPA 攻击，算法 W_2 采用间接方法实现数据隐藏，首先在行 B 产生随机遮盖，然后计算输入明文数据 PTI 和随机遮盖的异或值 mPTI 作为中间结果，最后在行 C 计算 mPTI 和密钥的异或值。如此，随机遮盖在算法的内部产生对攻击者不可见，分别攻击行 B 和行 C，泄漏的仅是随机信息，因此算法 W_2 能有效的防御 DPA 攻击；然而当考虑行 B 和行 C 的联合攻击时，算法 W_2 不能有效的防御二阶 DPA 攻击。

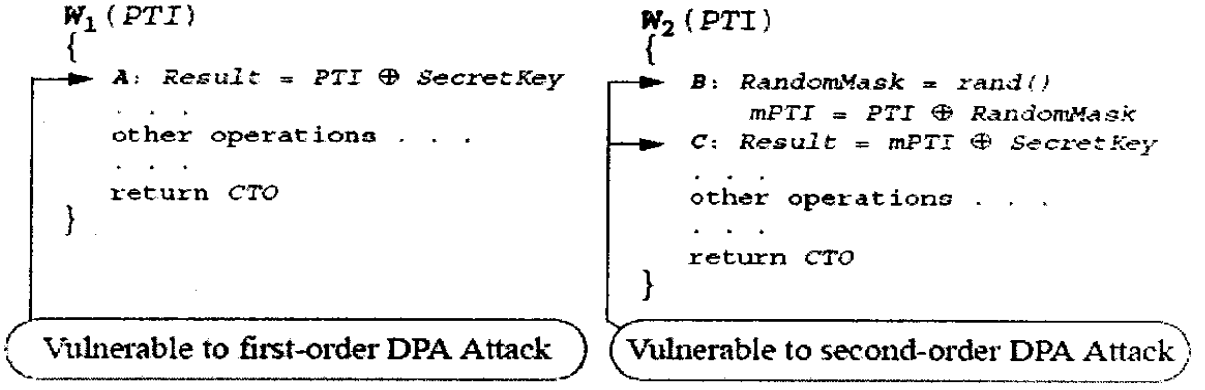

图 21：一阶差异能量分析对左边的程序的 A 行能实施有效攻击，右边的程序能有效防御一阶差异能量攻击，但对二阶差异能量攻击很脆弱，攻击者可以通过算法执行 B 行、C 行时的能量消耗的联合统计来实施有效的二阶差异能量攻击。

一阶差异能量攻击与二阶差异能量攻击的比较

命题 1：当 W_1 算法进行 N 位处理时，假定瞬时能量消耗线性依赖于被处理数据的汉明加权，下面的 DPA 攻击是有效的：

for(i=0, i<=N-1, i++){

反复令 b=0 或 b=1 {

令明文输入的第 i 位为 b, 其余的明文位值为随机值，收集算法的能量消耗计算平均能量信号 $A_b[j]$}

得到 DPA 差异统计量 $T[j]=A_0[j]-A_1[j]$.

当 T[j]有正的偏差峰值时，密钥的第 i 位为 1，当 T[j]有负的偏差峰值时，密钥的第 i 位为 0。

}

考虑图 21 中的算法 W_2，算法先产生 RandomMask 遮盖明文数据 PTI 得到 mPTI，行 C 计算 mPTI 和密钥的异或值，结果的汉明加权是随机的，如此，行 C 的能量消耗与密钥或明文 PTI 值无关，算法 W_2 能有效防御 DPA 攻击，但下面的命题表明算法 W_2 对二阶能量差异攻击很脆弱。

命题 2：当 W_2 法进行 N 位处理时，假定瞬时能量消耗线性依赖于被处理数据的汉明加权，下面的二阶 DPA 攻击是成功的：

for(i=0, i<=N-1, i++){

反复令 b=0 或 b=1{

令明文输入的第 i 位为 b, 其余的明文位值为随机值，收集算法的能量消耗计算平均能量信号 $A_b[j]$

收集算法进行到行 B、行 C 时的瞬时能量消耗，分别令其值为 P_B 和 P_C

计算平均统计量 $\overline{S}_b=|P_B-P_C|$}

得到 DPA 差异统计量 $T=\overline{S}_0-\overline{S}_1$

当 T 大于零时，密钥的第 i 位为 1，当 T 小于零时，密钥的第 i 位为 0。

}

在一阶差异能量攻击中我们不要求知道算法的信息，然而在二阶差异能量分析中，攻击者我们必须知道算法的代码及其实际运作，没有这些知识，攻击者就无法知道能量消耗信号曲线上那些点是重要的。例如在 W_2 算法中这些重要的点对应程序中行 B、行 C 的运行！不知道观察点的攻击者就得借助更进一步的统计分析来找这些点，使攻击很难实施。因此我们假定攻击者事先知道能量消耗曲线上的监控点。

以 ST Microelectronics 公司的 ST16 智能卡的 EEPROM 运行图 21 所示的算法 W_1、 W_2，实验用接地电阻为 18ohm，能量信号通过采样接地电阻两边的电压来监控，智能卡时钟 3.57Mhz，设置数字示波器的采样率为 1.0G 采样点/秒以保证能异步采样！进行 DPA 攻击，收集能量消耗信号，在每次收集到能量信号后更新 T 的值，攻击的强度依收集到的能量信号的增多而增多，随着能量信号的增加，T 值收敛。命题 1 中一阶差异能量攻击的结果如图 22 所示，图中的曲线表明对攻击的每一位 T 值收敛，被攻击字节是 0X6B，对 bit#0，T 应该收敛到正值；对 bit#1，T 应该收敛到正值；对 bit#2，T 应该收敛到负值等。实验证明攻击是有效的！在所有的情况下收集的能量信号不多于 50 个时 T 值收敛！命题 2 中的二阶差异能量攻击的结果如图 23 所示，实验证明攻击是有效的！有趣的是对不同位，T 值收敛率不同，对有些位 T 收敛的很快，对另一些位 T 收敛得很慢。

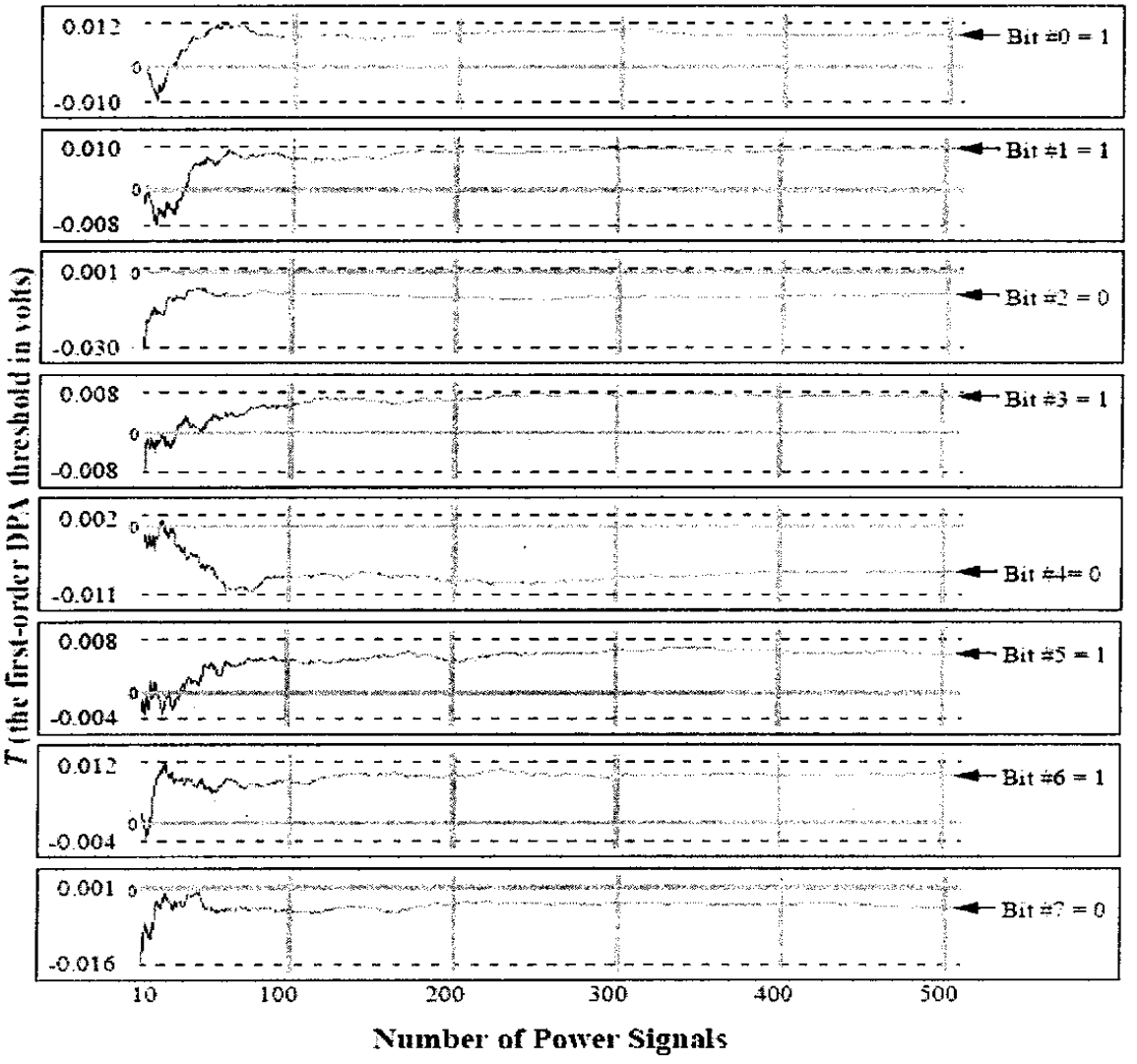

图 22：一阶差异能量分析的阈值与能量信号的数目之间的关系

图表示了 T 随能量信号的增加而收敛，被攻击的字节是 0X6B，对此字节的每位进行攻击的收敛图。水平的阴影线表示 T=0。T 取正值表明相应的位值为 1，T 取负值表明相应的位值为 0。在所有的情况下当收集的能量信号不多于 50 个时 T 值收敛！

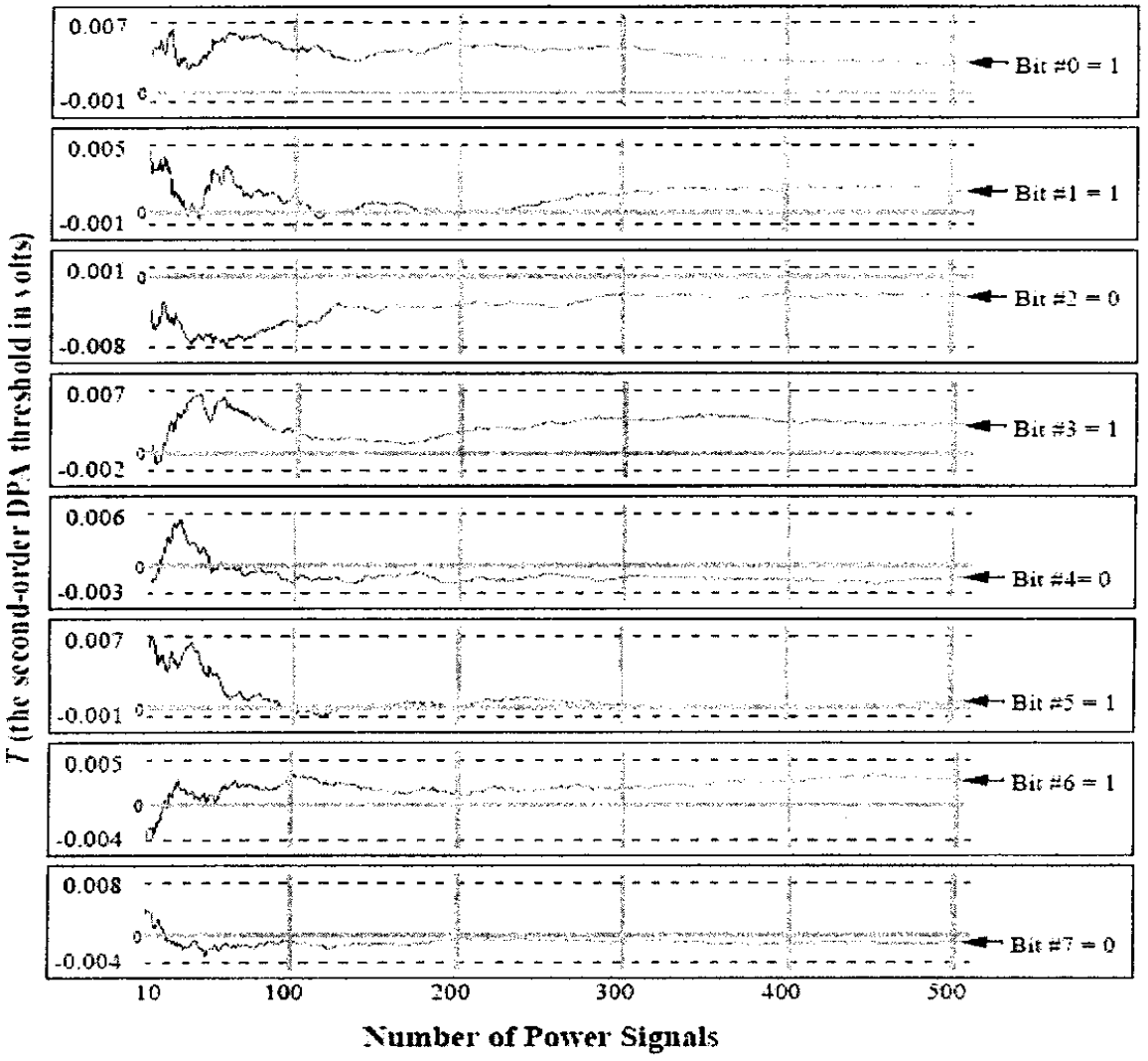

图 23：二阶差异能量分析的阈值与能量信号的数目之间的关系

图表示了 T 随能量信号的增加而收敛，被攻击的字节是 0X6B，对此字节的每位进行攻击的收敛图。水平的阴影线表示 T=0。T 取正值表明相应的位值为 1，T 取负值表明相应的位值为 0。在大多数情况下不少于 50 个能量信号 T 值收敛，对 bit#5, T 直到 2500 能量信号才收敛！

4. 防御方法

前面介绍了智能卡的各种物理攻击技术，包括（1）故障攻击，（2）时间攻击，（3）简单能量攻击与（4）差异能量攻击，这些攻击技术的成本仅有数万或数十万元而已，在一般的实验室就有足够的设备来做这些攻击实验。

最近，VISA 对其芯片式信用卡中，把其所使用的产品分成 3 个技术层次。简单而言，第 1 层明确定义要防止时间攻击，第 2 层明确定义要防止简单能量(simple power analysis)攻击，第 3 层明确定义要防止差异能量(differential power analysis)攻击，详细的技术层次定义参考[27]。技术层次愈高基本上表示该产品愈安全，VISA 认为符合第 1 层的产品，保护安全能力不够，并不适用

于当下的产品。芯片式的信用卡(VSDC 卡)必须符合第 2 层的规格，而可发卡后下载应用程序的 VOP 卡必须符合第 3 层的规格。由此可见，能防止各类的攻击已成为设计智能卡的基本常识，因此在研发智能卡的同时，应熟悉每一种或合并的多种攻击方式，讨论相应的对策，而研发出安全的智能卡来。智能卡的微控制器包括 CPU、交互界面、ROM、RAM、EEPROM 和加密协处理器，结构相对简单且须从外界获得能量，因此使得能量攻击者有隙可乘。自 Kocher[7]等提出能量攻击以来，人们研究了许多防御措施[31-39]，大体上这些防御法分为硬件防御法与软件防御法。硬件防御法有：

1. 采用先进的 0.6 微米工艺技术，使芯片的能量消耗以及智能卡计算过程中的能量消耗相对变化极大地降低！
2. 通过增大电容来产生额外的噪声和干扰信号，或通过增加滤波电路来消除噪声。
3. 基于硬件设计非线性密码更新系统，使得追踪能量消耗变得不可能！比如：采用硬件随机数发生器使每次芯片与外界数据传输中，产生的随机数可以保证数据不会重复。
4. 加保护层，或采用特殊的材料（对电子束敏感的材料）等技术，使监测芯片内执行的指令序列不可能实现。
5. 利用“量子纠缠（quantum entanglement)”来解决能量攻击。

Clavier 等[32]表明即使智能卡已有硬件保护，软件防御也是必需的。同时结合软、硬件防御措施能进一步保证智能卡内部数据的安全。普通的软件防御法是通过增加程序中的 “无效指令”来平衡能量消耗(增加能量消耗曲线的峰数)，使能量消耗的相对变化不明显，但这种方法被证明是无法对抗有效攻击的[7]。AES 是 NIST 于 2000 年宣布的用于替代 DES 的分组加密标准，也是目前最完善的密码体系，因此针对 AES 密码体系的差异能量攻击及其防御措施成为研究的热点。2001 年，Messerges[31]提出随机遮盖法以保护 AES 系统免于 DPA 攻击，这种方法用随机产生的遮盖值，每次必须计算新的 S 盒，因此得占用很多 RAM 空间和要求很强的运算能力；为此，Akkar、Golic[34-36]提出乘积遮盖法；Itoh 等[33]提出定值遮盖法；定值遮盖法相比乘积遮盖法更具实用性，但文献[37]-[40]表明现有的定值遮盖法无法对抗高阶差异能量攻击，本文提出了一种改进的定值遮盖法以对抗高阶差异能量攻击并进行了算法的安全性能分析。

4.1 **随机遮盖法**：Messerges[31]提出用随机遮盖法加固 AES 系统以对抗 DPA 攻击，通过改变原始操作实现遮盖，其改变如下：

表查找操作

x 作为输入，y 作为输出的表查找操作 S 可以表示为：

$$y = S[x]$$

当输入遮盖为 m_{in} 输出遮盖为 m_{out} 时，遮盖表为 S' 可由如下方程定义：

$$S'[x] = S[x \oplus m_{in}] \oplus m_{out}$$

遮盖表 S' 的输入被 m_{in} 遮盖，输出被 m_{out} 遮盖。

位布尔函数

为了计算两个遮盖的操作数的异或值，可以分别计算操作数的异或值和它们相

应遮盖的异或值，如果被遮盖的操作数分别为 x'、y'，相应的遮盖值分别为 m_x、m_y，那么被遮盖的输出 $z' = x' \oplus y'$，新的遮盖值为 $m_z = m_x \oplus m_y$。

移位操作

在此操作中，遮盖沿数据逐位移动。

有限域 GF（2^8）中的多项式乘

当用表查找操作、移位和异或操作实现乘积运算时，相应的方法用于这些操作可以使乘积运算免于攻击。用上述改变的原始操作，遮盖的数据被加密的同时须计算相应的遮盖值，最后，再次应用遮盖得到去遮盖的秘文。对每个轮次和对状态中的每个字节，此方法均随机产生新的遮盖值，因此要求更多的处理时间和更大的内存；每次加密都伴随着相应的遮盖的计算，对每个遮盖都得临时生成新的字节替换变换表。考虑到这个问题，Messerges 注释说对不同变量和轮次可以用同样的遮盖值，建议用定值遮盖法，但他本人并没有将此思想付诸实现。

4.2 乘积遮盖法：

由于 S 盒的重新计算要求更多的处理时间和 RAM，因此完善遮盖方法的主要问题是算法的非线性部分。为了解决此问题，Akkar 等[34]提出乘积遮盖法。对 AES，由于字节替换变换由 GF（2^8）中的求逆和图 24 中所示的仿射变换组成，因此在求逆操作之前布尔遮盖被变换为乘积遮盖。当状态 $a_{i,j}$ 中的字节被遮盖值 m_r 遮盖时，用图 25 所示的流程进行时，在求逆运算之后我们可以获得同样的遮盖值。但 Golic 等[35]表明乘积遮盖法对 DPA 很脆弱，乘积遮盖法最基本的问题是它并不能遮盖数据中所有的零值字节。

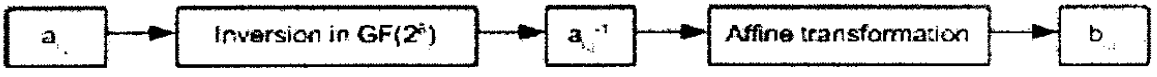

图 24：字节替换变换

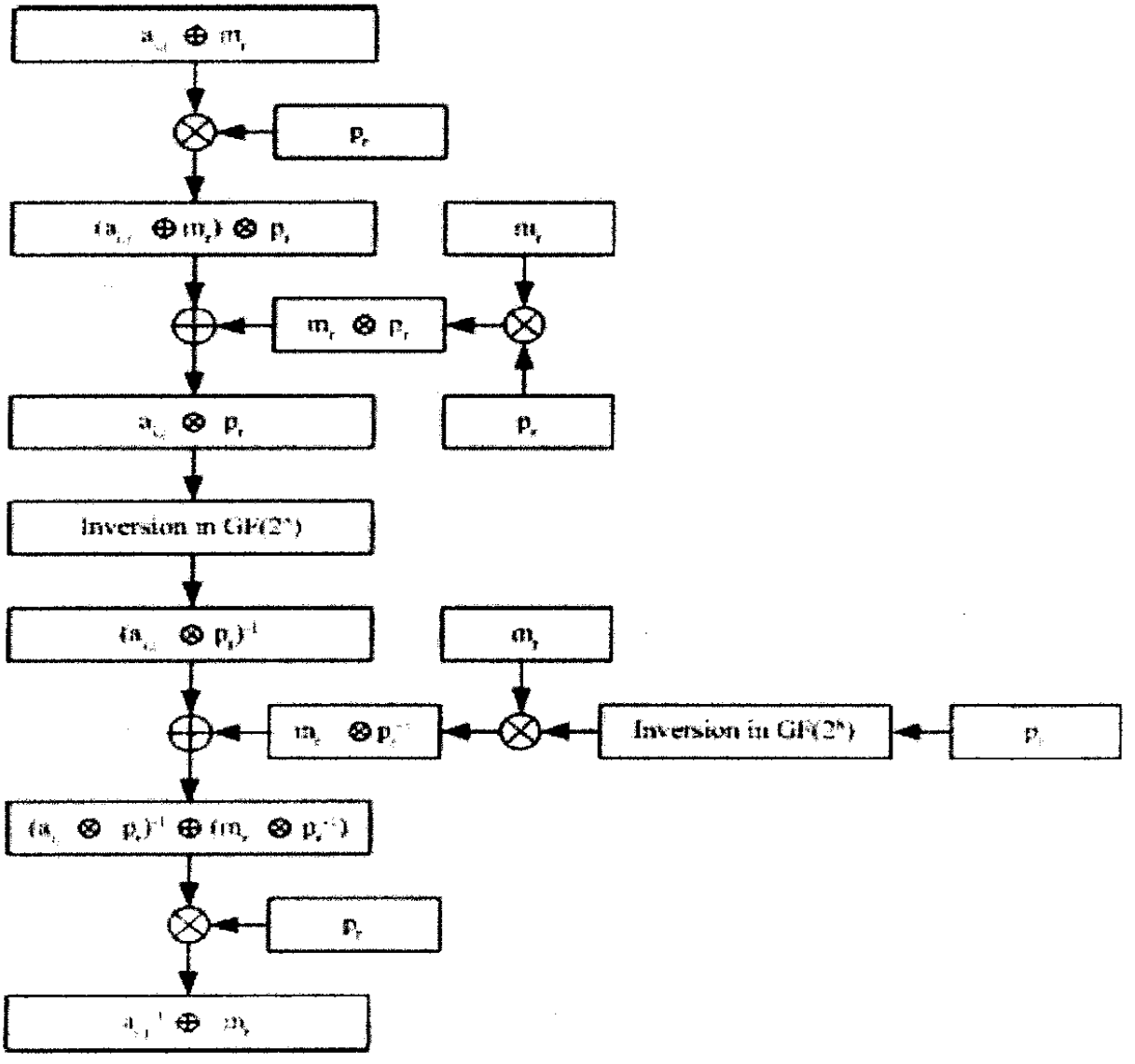

图 25：乘积遮盖法

4.3 定值遮盖法

在这种方法中，事先计算 q 套遮盖值和相应的改进的 S 盒，并存储在 ROM 中。K. Itoh 等[33]提出两种类型的定值遮盖方法，在第一种方法中，每轮所用的遮盖值是相同的但对状态中不同的字节用不同的遮盖值；在第二种方法中，每轮和状态中的每个字节都用相同的遮盖值。当执行加密运算时，一套遮盖值和改进的 S 盒被随机选用，由于第二种方法只占用少量的 ROM，即使对便宜的智能卡也是可行的。第二种方法所用的 AES 算法如图 26 和算法 1 所示，算法 1.1 是 *ByteSub _ FM* 变换的改进版，用的是存储在 ROM 中改进的 S 盒，*ShiftRow* 变换和 *MixColumn* 变换和原来的变换是相同的。有三种类型的遮盖值 $FK_{i,r}$

（$i=0, i=1,...Nr-1$和$i=Nr$， Nr 是 AES 加密系统中的轮数）用于遮盖轮密钥，这些遮盖值可以由 FIN 和 FOUT 导出：

$$FK_{i,r}=\begin{cases} FIN_r, & i=0 \\ MixColumn(ShiftRow(FOUT_r))\oplus FIN_r, & i=1...Nr-1 \\ ShiftRow(FOUT_r), & i=Nr \end{cases}$$

式中，FIN 是状态中的字节在 *ByteSub _ FM* 变换之前所用的遮盖值，FOUT 是状态中的字节在 *ByteSub _ FM* 变换之后所用的遮盖值，FIN 和 FOUT 被随机产生以使遮盖值任意一位为零的概率是 1/2。算法 1.2 计算了改进的 S 盒。

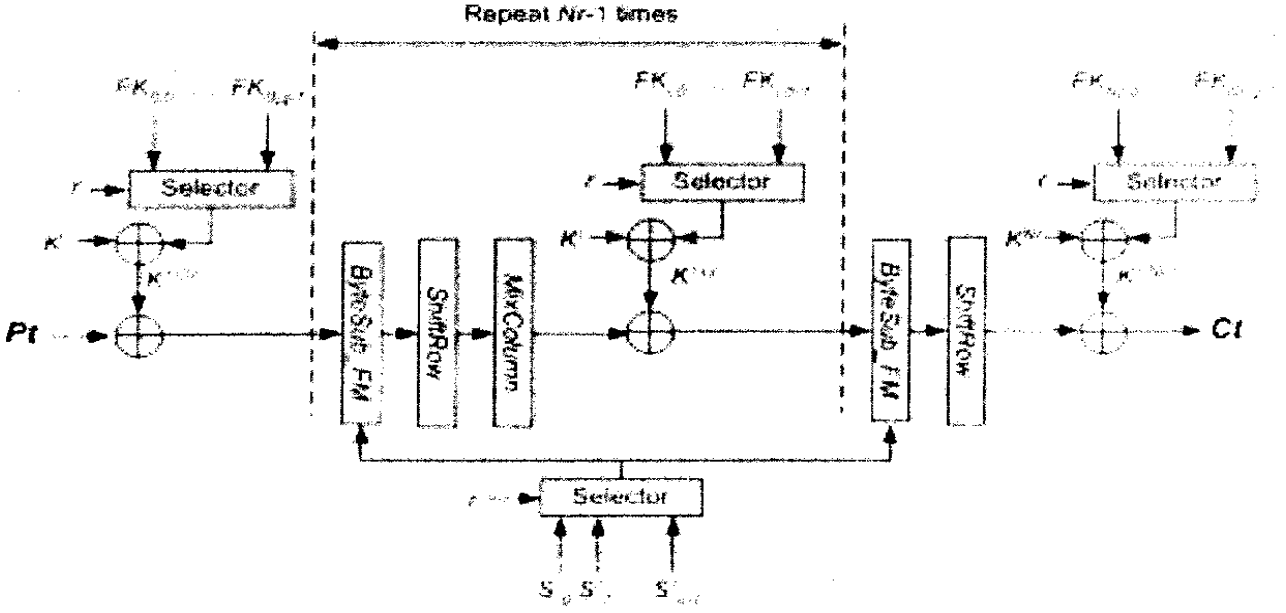

图 26：定值遮盖法

算法 1：定值遮盖法

$FixedValueMasking(Pt)$

1. /* $K''^{,r}$: $maskedroundkey(i=0,\ldots Nr, r=0,\ldots q-1)$ */
2. $r = GenerateRandomNumber()$; /* $chooser = 0,\ldots q-1$ */
3. $T' = Pt$;
4. $for(i=0; i<Nr-1; i++)\{$
5. $\quad T' = T' \oplus K''^{i,r}$;
6. $\quad T' = ByteSub\_FM(T', r)$;
7. $\quad T' = ShiftRow(T')$;
8. $\quad T' = MixColumn(T')$;
9. $\}$
10. $T' = T' \oplus K'^{Nr-1,r}$;
11. $T' = ByteSub\_FM(T', r)$;
12. $T' = ShiftRow(T')$;
13. $T = T' \oplus K'^{Nr,r}$;
14. $outputT$;

算法 1.1：字节替换操作

$ByteSub\_FM(X, r)$

1. $(x_{15}, x_{14}, \cdots x_0) = X$;
2. $for\ (j=0; j<16; j++) x_j = S'_r[x_j]$;
3. $X = (x_{15}, x_{14}, \cdots x_0)$;
4. $output\ X$;

算法 1.2：S 盒更新操作

$SboxUpdate\ (S, r)$

1. $for(x=0; x<256; x++)\ S'[x] = S[x \oplus FIN_r] \oplus FOUT_r$;
2. $output\ S'$;

4.4 简化的定值遮盖法

在这种方法中，q 套单字节的遮盖值和相应的改进的 S 盒被事先计算，并存储在 ROM 中。改进的 S 盒是用相应的遮盖值计算的（算法 2.2）。在加密的开始，随机选择一对遮盖值和改进的 S 盒，明文的所有字节被同样的单字节遮盖值所遮盖，然后用改进的 S 盒加密被遮盖的明文，在算法的最后，再次应用遮盖得到密文。图 27 和下面的算法中，$m_r\ (r=0,1,\ldots q-1)$ 是选择的遮盖，r 是选择的遮盖和改进的 S 盒的索引，S' 是改进的 S 盒，$ByteSub\_FM$ 变换与定值遮盖法相同。

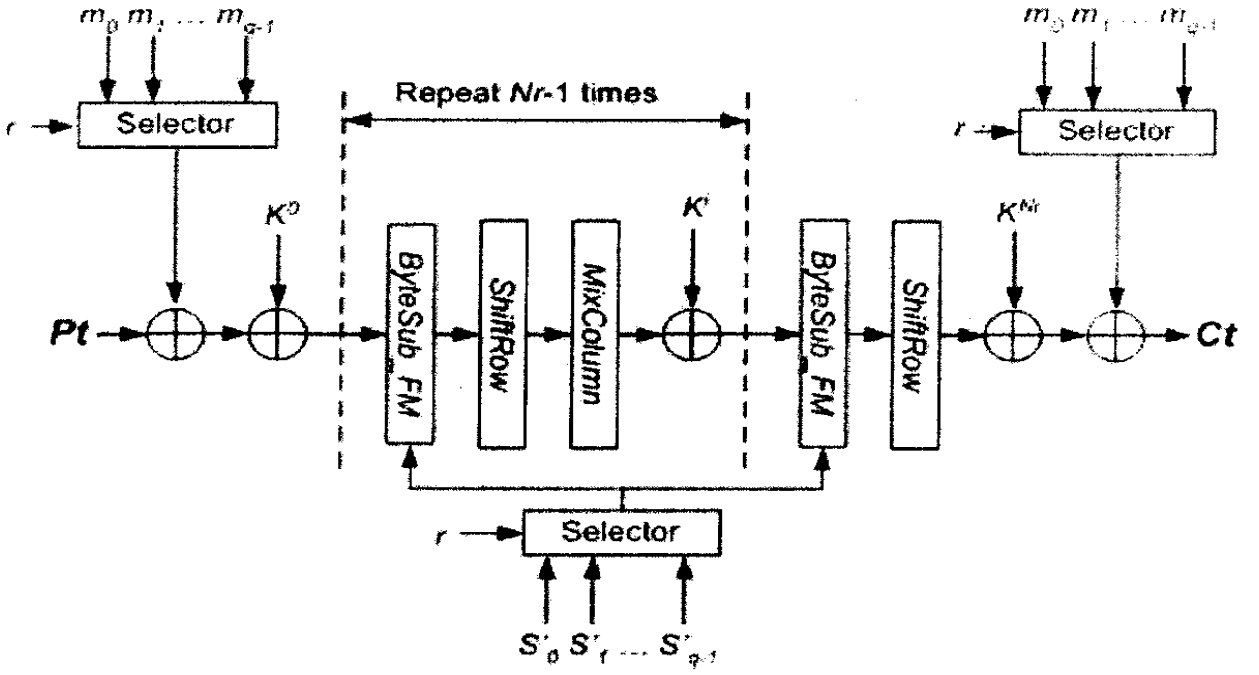

图 27：简化的定值遮盖法

算法 2：简化的定值遮盖法

SimpleFixedValueMasking(*Pt*)

1. $r = GenerateRandomNumber();$ /* $r = 0, \ldots q-1$ */
2. $T' = Pt;$
3. $T' = ApplyMask(T', r);$
4. $T' = T' \oplus K^0;$
5. $for(i = 1; i < Nr; i++)\{$
6. $\quad T' = ByteSub\_FM(T', r);$
7. $\quad T' = ShiftRow(T');$
8. $\quad T' = MixColumn(T');$
9. $\quad T' = T' \oplus K^i;$
10. $\}$
11. $T' = ByteSub\_FM(T', r);$
12. $T' = ShiftRow(T');$
13. $T' = T' \oplus K^{Nr};$
14. $T = Applymask(T', r);$
15. *output* $T;$

算法 2.1：ApplyMask

Applymask(T', r)

1. $(t'_{15}, t'_{14}, \ldots t'_0) = T'$;

2 *for*$(j = 0; j < 16; j++)$ $t'_j = t'_j \oplus m_r$;

3. $T' = (t'_{15}, t'_{14}, \ldots t'_0)$;

4. *output* T';

算法 2.2：S 盒更新操作 2

SboxUpdate 2(S, r)

1. *for*$(x = 0; x < 256; x++)$ $S'[x] = S[x \oplus m_r] \oplus m_r$;

2. *output* S';

我们在算法起始处运用遮盖，最终能得到期望的密文的原因是遮盖值在 AES 所有的变换过程中一直能保持不变，由于所用的是改进的 S 盒且同样的遮盖值应用到状态中的所有字节，证明如下：

ByteSub 变换：在字节替换变换中，由于是用改进的 S 盒，遮盖值得到保持，所以算法 2.2 并不会改变遮盖值。

ShiftRow 变换：行移位变换不会改变遮盖值，因为状态中所有的字节都用相同的遮盖值，ShiftRow 变换改变的仅仅是位置。

MixColumn 变换：我们考虑状态中的第 j 列，列中的每个字节表示为 $a_{i,j}$ $(i = 0,1,2,3)$，列混合变换由下式表示：

$$\begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix}$$

如果列中的每个字节都用 m_r 遮盖则有：

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_{0,j} \oplus m_r \\ a_{1,j} \oplus m_r \\ a_{2,j} \oplus m_r \\ a_{3,j} \oplus m_r \end{bmatrix}$$

$$= \begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} \oplus m_r \begin{bmatrix} 02 \oplus 03 \oplus 01 \oplus 01 \\ 01 \oplus 02 \oplus 03 \oplus 01 \\ 01 \oplus 01 \oplus 02 \oplus 03 \\ 03 \oplus 01 \oplus 01 \oplus 02 \end{bmatrix} = \begin{bmatrix} b_{0,j} \oplus m_r \\ b_{1,j} \oplus m_r \\ b_{2,j} \oplus m_r \\ b_{3,j} \oplus m_r \end{bmatrix}$$

故在列混合变换之间的遮盖值在变换后仍然能得到保持。

AddRoundKey 变换：

$$(a_{i,j} \oplus m_r) \oplus k^l_{i,j} = (a_{i,j} \oplus k^l_{i,j}) \oplus m_r \quad (r = 0,1,\ldots q-1; i = 0,1,2,3; j = 0,1,2,3)$$

表明通过轮密钥加法变换遮盖值得到保持。式中，$a_{i,j}$ 为状态中的第 i 行第 j 列的字节，m_r 是选定的遮盖值，$k^l_{i,j}$ 是第 l 轮子密钥第 i 行第 j 列的字节。

4.5 改进简化的定值遮盖法以对抗高阶差异能量攻击

简化的定值遮盖法可以对抗 DPA 攻击，但它却不能有效地防御二阶差异能量攻击，为了对抗二阶差异能量攻击，须使攻击者难以判断装载遮盖的点。为此我们事先装载两个以上的遮盖在存储器中用作异或指令的操作数，随机地选用遮盖，就得到了可以对抗二阶差异能量攻击（SODPA）的改进的简化定值遮盖法。

算法 3：改进的简化定值遮盖法防御 SODPA

$SODPA\,\mathrm{Re}\,sis\,\tan\,tSFVM(Pt)$

1. $r[0] = GenerateRandomNumber();$ /* $r[0] = 0,\ldots q-1$ */
2. $r[1] = GenerateRandomNumber();$ /* $r[1] = 0,\ldots q-1$ */
3. $lm[0] = m_{r[0]};$
4. $lm[1] = m_{r[1]};$
5. $s = GenerateRandomNumber();$ /* 选择$s = 0$或1 */
6. $T' = Pt;$
7. $T' = ApplyMask2(T', s);$
8. $T' = T' \oplus K^0;$
9. $for(i = 1, i < Nr; i++)\{$
10. $\quad T' = ByteSub\_FM(T', r[s]);$
11. $\quad T' = ShiftRow(T');$
12. $\quad T' = MixColumn(T');$
13. $\quad T' = T' \oplus K^i;$
14. $\}$
15. $T' = ByteSub\_FM(T', r[l]);$
16. $T' = ShiftRow(T');$
17. $T' = T' \oplus K^{Nr};$
18. $T' = ApplyMask\ 2(T', s);$
19. $output\ T;$

算法 3.1：ApplyMask2

$ApplyMask\ 2(T', s)$

1. $(t'_{15}, t'_{14}, \ldots t'_0) = T';$
2. $for(j = 0; j < 16; j++)t'_j = t'_j \oplus lm[s];$
3. $T' = (t'_{15}, t'_{14}, \ldots t'_0);$
4. $output\ T';$

像 ST72101 这样的 8 位微控制器有异或指令，源操作数可以被累加并存储地址，目的操作数可以被累加。在实现 ApplyMask2 算法时，必须注意到遮盖值并没有从算法 3 中第 3、4 行的存储位置中移出，如果它移到其他的存储器中，此瞬间将成为 SODPA 攻击的目标。如果遮盖没有移动且未用在用遮盖的地址的地方，攻击者就无法找到实施 SODPA 攻击的点。在算法 3 中，攻击者有可能攻击遮盖被移动到微控制器的 ALU 中所对应的点。不同的指令对 DPA 有不同程度的脆弱性，异或指令还相对坚强些。在算法 4 中，我们事先准备了两个遮盖的明文，只有选上的明文被处理，这样攻击者无法知道遮盖的装载时间，因而相比算法 3 更安全。

算法 4：改进的简化定值遮盖法 2 防御 SODPA

$SODPA\,\mathrm{Re}\,sis\,\tan\,tSFVM2(Pt)$

1. $r[0] = GenerateRandomNumber();\ /*\ r[0] = 0,\ldots q-1\ */$
2. $r[1] = GenerateRandomNumber();\ /*\ r[1] = 0,\ldots q-1\ */$
3. $lm[0] = m_{r[0]};$
4. $lm[1] = m_{r[1]};$
5. $s = GenerateRandomNumber();\ /*$ 选择$s = 0$或$1\ */$
6. $Pt[0] = Applymask2(Pt,0);$
7. $Pt[1] = Applymask2(Pt,1);$
8. $T' = Pt[s];$
9. $T' = T' \oplus K^0;$
10. $for(i = 1, i < Nr; i++)\{$
11. $\quad T' = ByteSub\_FM(T', r[s]);$
12. $\quad T' = ShiftRow(T');$
13. $\quad T' = MixColumn(T');$
14. $\quad T' = T' \oplus K^i;$
15. $\}$
16. $T' = ByteSub\_FM(T', r[l]);$
17. $T' = ShiftRow(T');$
18. $T' = T' \oplus K^{Nr};$
19. $Ct[0] = ApplyMask\ 2(T',0);$
20. $Ct[1] = ApplyMask\ 2(T',1);$
21. $output\ Ct[s];$

算法 4 能有效地防御二阶差异能量攻击，但对更高阶（高于二阶）的差异能量攻击未必有效，为此我们结合算法 3、4：事先装载两个遮盖在存储器中，随机地选用遮盖，使攻击者难以判断装载遮盖的点；同时准备了两个遮盖的明文，只有选上的明文被处理，使得攻击者无法知道遮盖的装载时间，因而相比算法 3、4 更安全，可以防御高阶差异能量攻击。

算法 5：改进的简化定值遮盖法防御高阶差异能量攻击

$HODPA\,\mathrm{Re}\,sis\,\tan\,tSFVM(Pt)$

1. $r[0] = GenerateRandomNumber();$ /* $r[0] = 0,\ldots q-1$ */
2. $r[1] = GenerateRandomNumber();$ /* $r[1] = 0,\ldots q-1$ */
3. $lm[0] = m_{r[0]};$
4. $lm[1] = m_{r[1]};$
5. $s = GenerateRandomNumber();$ /* 选择$s = 0$或1 */
6. $Pt[0] = Applymask2(Pt,0);$
7. $Pt[1] = Applymask2(Pt,1);$
8. $T' = Pt[s];$
9. $T' = ApplyMask2(T', s);$
10. $T' = T' \oplus K^0;$
11. $for(i = 1, i < Nr; i++)\{$
12. $\quad T' = ByteSub\_FM(T', r[s]);$
13. $\quad T' = ShiftRow(T');$
14. $\quad T' = MixColumn(T');$
15. $\quad T' = T' \oplus K^i;$
16. $\}$
17. $T' = ByteSub\_FM(T', r[l]);$
18. $T' = ShiftRow(T');$
19. $T' = T' \oplus K^{Nr};$
20. $T' = ApplyMask2(T', s);$
21. $Ct[0] = ApplyMask\ 2(T',0);$
22. $Ct[1] = ApplyMask\ 2(T',1);$
23. $output\ Ct[s];$

5. 安全性能分析及结论

为了实施 DPA 攻击，必须确定一作为明文或密文与部分密钥的函数的中间值，如果此条件满足，攻击者就能逐步得到密钥。但在简化的定值遮盖法中，状态中所有字节被随机选择的遮盖值所遮盖，攻击者无法知道用于遮盖的具体遮盖值，从而无法由明文和部分密钥获取状态中的字节。由于遮盖被保持直到最后的轮密钥加法变换完成，所以攻击者无法从密文中估计状态中的字节。如果我们选择任意位为零的概率是 0.5 的遮盖，简化的定值遮盖法能抵抗文献[33]中的概率 DPA 攻击。仿照文献[33]中的证明，我们的证明如下：

5.1 防御 DPA 攻击 S 盒的证明

令 x 的第 d 位为零的概率是 $\beta_d(x)$，则改进的 S 盒的输出和第一轮原来的 S

盒的关系为：

$$S'\left[a_{i,j} \oplus m_r \oplus k^0_{i,j}\right] = S\left[a_{i,j} \oplus m_r \oplus k^0_{i,j} \oplus m_r\right] \oplus m_r = S\left[a_{i,j} \oplus k^0_{i,j}\right] \oplus m_r$$

假设攻击者估计密钥为 $k^0_{i,j}$，并用从 $f = S\left[a_{i,j} \oplus k^0_{i,j}\right]$ 选择的位元计算差异能量消耗曲线 Δ_d，有：

$$\varepsilon_{S'} = \frac{2}{N}\left\{\sum_{V_c \in \delta 1_d\left(S\left[a_{i,j} \oplus k^0_{i,j}\right]\right)} V_c\left(t_{load\ S'}\right) - \sum_{V_c \in \delta 0_d\left(S\left[a_{i,j} \oplus k^0_{i,j}\right]\right)} V_c\left(t_{load\ S'}\right)\right\}$$

上式中，$\varepsilon_{S'}$ 是相应于 S 盒的一字节输出，差异能量消耗曲线上尖峰的数值。

由于 $S\left[a_{i,j} \oplus k^0_{i,j}\right]$ 的第 d 位以概率 $\beta_d(m_r)$ 相等于 $S\left[a_{i,j} \oplus k^0_{i,j}\right] \oplus m_r$ 的第 d 位，我们可以得到下面的等式：

$$\begin{aligned}
\varepsilon_{S'} = \frac{2}{N}\Bigg\{ & \beta_d(m_r) \sum_{V_c \in \delta 1_d\left(S\left[a_{i,j} \oplus k^0_{i,j}\right] \oplus m_r\right)} V_c\left(t_{load\ S'}\right) \\
& + \left(1 - \beta_d(m_r)\right) \sum_{V_c \in \delta 0_d\left(S\left[a_{i,j} \oplus k^0_{i,j}\right] \oplus m_r\right)} V_c\left(t_{load\ S'}\right) \\
& - \beta_d(m_r) \sum_{V_c \in \delta 0_d\left(S\left[a_{i,j} \oplus k^0_{i,j}\right] \oplus m_r\right)} V_c\left(t_{load\ S'}\right) \\
& - \left(1 - \beta_d(m_r)\right) \sum_{V_c \in \delta 1_d\left(S\left[a_{i,j} \oplus k^0_{i,j}\right] \oplus m_r\right)} V_c\left(t_{load\ S'}\right)\Bigg\} \\
= \frac{2(2\beta_d(m_r) - 1)}{N}\Bigg\{ & \sum_{V_c \in \delta 1_d\left(S\left[a_{i,j} \oplus m \oplus k^0_{i,j}\right]_*\right)} V_c\left(t_{load\ S'}\right) \\
& - \sum_{V_c \in \delta 0_d\left(S\left[a_{i,j} \oplus m \oplus k^0_{i,j}\right]_*\right)} V_c\left(t_{load\ S'}\right)\Bigg\}
\end{aligned}$$

由上式，我们可以看到如果 $\beta_d(m_r)$ 为 0.5 则差异能量消耗曲线上尖峰的数值为零，这意味着遮盖值的任意位为零的概率是 0.5。

5.2 防御 DPA 攻击轮密钥加法变换的证明

攻击者用从 $f = a_{i,j}$ 选择的位元计算差异能量消耗曲线 Δ_d，我们假定应用轮密钥的操作序列如下：

load $a_{i,j}$

load $k^0_{i,j}$

xor $t_{i,j}, a_{i,j}, k^0_{i,j}$

store $t_{i,j}$

load $t_{i,j}$

装载$t_{i,j}$时差异能量消耗曲线上尖峰的数值为：

$$\varepsilon_{load\ t_{i,j}} = \frac{2}{N}\left\{\sum_{V_c \in \delta 1_d(a_{i,j})} V_c\left(t_{load\ t_{i,j}}\right) - \sum_{V_c \in \delta 0_d(a_{i,j})} V_c\left(t_{load\ t_{i,j}}\right)\right\}$$

考虑到

由于$a_{i,j}$的第 d 位以概率$\beta_d(m_r)$相等于$a_{i,j} \oplus k_{i,j}^0$的第 d 位，差异能量消耗曲线上尖峰的数值为：

$$\begin{aligned}\varepsilon_{load\ t_{i,j}} = \frac{2}{N}\Bigg\{ & \beta_d(m_r) \sum_{V_c \in \delta 1_d(a_{i,j} \oplus m_r)} V_c\left(t_{load\ t_{i,j}}\right) \\ & + (1-\beta_d(m_r)) \sum_{V_c \in \delta 0_d(a_{i,j} \oplus m_r)} V_c\left(t_{load\ t_{i,j}}\right) \\ & - \beta_d(m_r) \sum_{V_c \in \delta 0_d(a_{i,j} \oplus m_r)} V_c\left(t_{load\ t_{i,j}}\right) \\ & - (1-\beta_d(m_r)) \sum_{V_c \in \delta 1_d(a_{i,j} \oplus m_r)} V_c\left(t_{load\ t_{i,j}}\right)\Bigg\} \\ = \frac{2(2\beta_d(m_r)-1)}{N}\Bigg\{ & \sum_{V_c \in \delta 1_d(a_{i,j} \oplus m_r)} V_c\left(t_{load\ t_{i,j}}\right) \\ & - \sum_{V_c \in \delta 0_d(a_{i,j} \oplus m_r)} V_c\left(t_{load\ t_{i,j}}\right)\Bigg\}\end{aligned}$$

由上式，我们可以看到如果$\beta_d(m_r)$为 0.5 则盖值的任意位为零的概率是 0.5，差异能量消耗曲线上尖峰的数值为零，攻击者无法猜测密钥。简化的定值遮盖法可以对抗 DPA 攻击，但它却不能有效地防御二阶差异能量攻击。

5.3 防御 SODPA 攻击的证明

在我们提出的防御 SODPA 的改进的定值遮盖方法中，攻击者无法确定遮盖装载的时间，从而无法实施二阶差异能量攻击[37]。如果同样的遮盖用一轮以上，攻击者可以通过互相关来实施二阶差异能量攻击，为了避免这种攻击，每轮的遮盖值应该重新选择。在遮盖改变的点，旧遮盖和新遮盖的异或值应用到状态中的每个字节。

5.4 所需遮盖的数目

为了将我们提出的方法应用到实际的智能卡上，必须事先确定遮盖的数目 q。如果我们只用一个遮盖，遮盖值应该是 0xFF 以满足对抗概率 DPA 攻击的条件。但是 Akkar[41]等表明 PODPA 能攻击此类情况，为了实施 PODPA，攻击者应该知道目标设备的准确的能量消耗特性而这对通常的攻击者而言是不容易做的。对低端智能卡来说遮盖的数目取 1，但对高端智能卡而言 q 应该大于一，为了满足对抗概率 DPA 攻击的条件 q 应该是偶数，同时考虑到安全性一般 q 取大于等于 4 的数。

5.5 性能分析

为了对抗 DPA 攻击，遮盖和改进的 S 盒必须用在整个加密过程中，在这种情况下额外的操作和内存要求如下：1.得产生选择遮盖所用的随机数，2.遮盖应该与状态中的每个字节做异或运算，选择改进的 S 盒代替原来的盒做加密运算 3.在加密结束后再次运用遮盖得到最终的密文。在此流程中额外的处理操作由随机数产生、初始遮盖的应用和最终遮盖的应用组成，额外的内存是寄存器中的一个或两个字节额外的处理是随机数产生和 32 次异或操作。

为了对抗 SODPA 攻击，就需要更多的资源：要隐藏遮盖的装载时间，得产生三个随机数、事先得准备好两个已遮盖的明文还需计算两个未遮盖的密文，总共需要 40 个字节的额外内存以及 64 次额外的异或运算；为了在不同轮次改变遮盖，新的遮盖得应用 Nr 次而且需要额外的 $Nr*(1+16)*2$ 次异或操作来计算旧遮盖与新遮盖的两个异或值并应用计算得到的值到状态中的每个字节中，数字 2 表示两个已遮盖的状态。

5.6 对比结果

我们所提出的改进的定值遮盖方法有节省内存和加速处理两方面的优势。以前的定值遮盖法需要三个字节来存储一种类型的遮盖，一是对初始的轮密钥、一是对中间的轮密钥还有一个是对最终的轮密钥。但我们所提出的方法只需要一个字节的遮盖。我们考虑对抗 SODPA 的情况：我们的方法由于只在两点（初始的轮密钥加法变换之前和最终的轮密钥加法变换之后）上用遮盖从而加快了处理速度，而以前的定值遮盖法在 $(Nr+1)$ 个点上用遮盖（一是用在最初的轮密钥加法变换，$(Nr-1)$ 个用在中间的轮密钥加法变换还有一个用在最终的轮密钥加法变换）。另外如果密钥的长度大于 128 位则在一个点所需的异或操作数还会增加。对 AES 系统来说，先前的定值遮盖法所需的异或操作数为 11points*16bytes=176 次,而我们提出的方法所需的异或操作数为 2points*16bytes=32 次操作。

在一些应用场合（例如交通卡），智能卡的处理速度至关重要，交易时间必须足够短以确保不会影响顾客；而智能卡的处理能力一般来说比较差且加密过程会占用整个交易时间的一大部分，因此]加密算法的优化势在必行！如果我们针对 AES-128 密码体系在 32 微处理器上优化加密算法：对每轮状态中的每一列需 4 个表查找操作和 4 个异或操作，故每轮需 16 个表查找操作和 16 个异或操作，整个加密过程中初始的轮密钥加法变换需要 4words 尺寸的异或操作，其它部分需要 320 次操作，共 324 次异或操作。如果我们用定值遮盖法，整个加密过程中每轮只需额外的 4words 尺寸的异或操作共 4*11=44 次异或操作，这增加了加密的操作数大约 13.6%。而用我们提出的改进的定值遮盖法，只需额外的 4*2=8words 尺寸的异或操作遮盖状态中的列，这增加了加密的操作数只有 2.5%，由此可见我们提出的改进的定值遮盖法的优越性。

5.7 结论

本文针对 AES 密码系统研究了防御智能卡差异能量攻击的定值遮盖法及其安全性能分析。我们先回顾了 AES 和 DPA 以及以前的各种遮盖方法，其后我们针对 AES 密码体系提出了能防御高阶差异能量攻击的改进的简化定值遮盖法（Hwasun Chang）：事先装载两个遮盖在存储器中，随机地选用遮盖，使攻击者难以判断装载遮盖的点；同时准备了两个遮盖的明文，只有选上的明文被处理，

使攻击者无法知道遮盖的装载时间，从而能防御高阶差异能量攻击。在算法分析中，我们分析了产生遮盖所要求的特性、遮盖的数目、所要求的加法运算、完善防御方法所需的内存以及提出方法的安全性。

6. 参考文献

1. R. J. Anderson and M. G. Kuhn, Tamper resistance - a cautionary note, Proceedings of Second USENIX Workshop on Electronic Commerce (Oakland, California), 1996, pp. 1~11.
2. R. J. Anderson and M. G. Kuhn, Low cost attacks on tamper resistant devices, Proceedings of International Workshop on Security Protocols 1997 (Paris, France) (M. Lomas et. al., ed.), Lecture Notes in Computer Science, vol. 1361, Springer-Verlag, 1997, pp. 125~136.
3. D. Noneh, R. A. Demillo, and R. J. Lipton, On the importance of checking cryptographic protocols for fault, Proceedings of EUROCRYPTO ' 97, Lecture Notes in Computer Science, vol. 1233, Springer-Verlag, 1997, pp. 37-51.
4. E. Biham and A. Shamir, Differential fault analysis of secret key cryptosystems, Proceedings of CRYPTO '97 (Burton S. Kaliski Jr., ed.), Lecture Notes in Computer Science, vol. 1294, Springer-Verlag, 1997, pp. 513~525.
5. P. C. Kocher, Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems, Proceedings of Crypto' 96, 1996.
6. P. Kocher, J. Jaffe, and B. Jun, Introduction to differential power analysis and related attacks, http://www.cryptography.com/dpa/technical/, 1998.
7. P. Kocher, J. Jaffe, and B. Jun, Differential power analysis, Proceedings of CRYPTO '99 (M. J. Wiener, ed.), Lecture Notes in Computer Science, vol. 1666, Springer-Verlag, 1999, pp. 388~397.
8. FIPS 46-3. Data Encryption Standard. Federal Information Processing Standard, National Institute of Standards and Technology, 25 October 1999.
9. Draft of AES - Federal Information Processing Standards Publication, Washington D.C.
10. Kuo, Henry and Ingrid Verbauwhede- Architectural Optimization for a 1.82Gbits/sec VLSI implementation of the AES Rijndael Algorithm
11. Rankl and W. Effing- Smart Card Handbook, Second Edition, Chichester, England, John Wiley & Sons Ltd., 2000
12. R.L. Rivest, A. Shamir, and L.M. Adelman, "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM, 21(1978), 120-126.
13. E. Biham and A. Shamir, ''A new cryptanalytic attack on DES: Differential fault analysis,'' October 1996.

14. Y. Zheng and T. Matsumoto, ''Breaking real-world implementations of cryptosystems by manipulating their random number generation,'' In Pre-proceedings of the 1997 Symposium on Cryptography and Information Security, Fukuoka, Japan, 1997.

15. M. Joye, A.K. Lenstra, and J.-J. Quisquater, ''Chinese remaindering based cryptosystems in the presence of faults,'' Journal of Cryptology, 1999.

16. F. Bao, R.H. Deng, Y. Han, A. Jeng, A.D. Narasimbalu, and T. Ngair, ''Breaking public key cryptosystems on tamper resistant devices in the presence of transient faults,'' In Pre-proceedings of the 1997 Security Protocols Workshop, Paris, France, 1997.

17. S.M. Yen and M. Joye, "Checking Before Output May Not Be Enough Against Fault-Based Cryptanalysis," IEEE Trans. on Computers, Vol.49, No.9, pp.967-970, Sept. 2000.

18. I. Biehl, B. Meyer, '' Differential fault attacks on elliptic curve cryptosystems,'' Advances in Cryptology - CRYPTO 2000, LNCS, Springer-Verlag, 2000.

19. J.F. Dhem, F. Koeune, P.A. Leroux, P. Mestre, J.J. Quisquater, and J.L. Willems, A Practical Implementation of the Timing Attack, Technical Report CG-1998/1, Universite catholique de Louvain, 1998.

20. H. Handschuh and H. Heys, A Timing Attack on RC5, In Workshop Record of Selected Areas of Cryptography - SAC' 98, pages 318-329, Queen' s University, 1998.

21. A. Hevia and M. Kiwi, Strength of Two Data Encryption Standard Implementation Under Timing Attacks, ACM Transactions on Information and System Security, 2(4), pp. 416-437, November 1999.

22. W. Schindler, A Timing Attack against RSA with the Chinese Remainder Theorem, Cryptographic Hardware and Embedded Systems - CHES 2000, pages 109-124, Springer-Verlag, August 2000.

23. J. Kelsey, B. Schneier, D. Wagner, and C. Hall, Side Channel Cryptanalysis of Product Ciphers, Journal of Computer Security, 8(2-3):141-158, 2000.

24. F. Koeune and J.J. Quisquater, A Timing Attack Against Rijndael, Technical Report CG-1999/1, Universite catholique de Louvain, 1999.

25. James Alexander Muir, "Techniques of Side Channel Cryptanalysis", Master Thesis, Department of Mathematics, University of Waterloo, Canada, 2001.

26. T.S. Messerges, E.A. Dabbish, and R.H. Sloan, Investigations of power analysis attacks on smartcards, Proceedings of USENIX Workshop on Smartcard Technology, 1999, pp. 151-161.

27. Eli Biham and Adi Shamir, "Power Analysis of the Key Scheduling of the AES Candidates," in Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference, March 1999. http://csrc.nist.gov/encryption/aes/round1/Conf2/aes2conf.htm

28. Thomas S. Messerges, Ezzy A. Dabbish and Robert H. Sloan, "Power Analysis Attacks of Modular Exponentiation in Smartcards," Workshop on Cryptographic Hardware and Embedded Systems, Springer-Verlag LNCS 1717, pp. 144-157.

29. Jean-Sebastien Coron, "Resistance Against Differential Power Analysis for Elliptic Curve Cryptosystems," Workshop on Cryptographic Hardware and Embedded Systems, Springer-Verlag LNCS 1717, pp. 292-302.

30. Jean McKenna, "VISA Chip Card Technology Level Definition", Visa Asia - Pacific Vendor Conference 'Building Partnerships for Success, 25 May 2001.

31. T. Messerges, "securing the AES Finalists Against Power Analysis Attacks," Fast Software Encryption Workshop-FSE 2000, LNCS 1978, pp. 150-164, Springer-Verlag , 2001.

32. C. Clavier, J. Corron, and N. Torh, "Differential Power Analysis in the Presence of Hardware Countermeasures, Workshop on Cryptographic Hardware and Embedded Systems, CHES 2000, LNCS 1965, pp. 252-263 Springer-Verlag 2000.

33. K. Itoh, M. Takenaka, and N. Torh, "DPA Countermeasure based on the masking Method ," International Conference on Information, Communications and Signal Processing ICICS 2001, LNCS 2288, pp. 440-456, Springer-Verlag 2002.

34. M. Akkar and C. giraud, An Implementation of DES and AES Secure against some Attacks, Workshop on Cryptographic Hardware and Embedded Systems, CHES 2001, LNCS 2162, pp. 309-318 Springer-Verlag 2001.

35. J. Golic and C. tymen, Multiplicative Masking and power Analysis of AES, Workshop on Cryptographic Hardware and Embedded Systems, CHES 2002, LNCS 2523, pp. 198-212 Springer-Verlag 2003.

36. E. Trichina, D. Seta, and L. Germani, Simplified Adaptive Multiplicative Masking for AES, Workshop on Cryptographic Hardware and Embedded Systems, CHES 2002, LNCS 2523, pp. 187-197 Springer-Verlag 2003.

37. T. Messerges, "using Second -order Power Analysis to Attack DPA Resistant Software, Workshop on Cryptographic Hardware and Embedded Systems, CHES 2000, LNCS 1965, pp. 238-251 Springer-Verlag 2000.

38. S. Yen, Amplified Differential Power Cryptanalysis on Rijudael Implementations with Exponentential Fewer Power

Traces ," Information Security and Privacy Australasian cinference-ACISP 2003, LNCS 2727. pp. 106-117, Springer-Verlag 2003.

39. S. Chari, C. Jutla, J. Rao, and P. Rohatgi, Towards Sound approaches to counter power analysis attacks, Advances in cryptology-crypto 1999, LNCS 1666, pp398-412, springer-verlag

40. Hwasun Chang , A Study on Securing AES against Differential Power Analysis, A thesis for the degree of Master, School of engineering Information and communications university, Dec. 29. 2003

41. M. Akkar, R. Bevan, P. Dischamp , and D. Moyart, Power Analysis, What Is Possible…, Advances in Cryptology ASIACRYPT 2000, LNCS 1976, pp489-502, Springer-Verlag, 2000.

致 谢

首先，作者衷心地感谢导师——彭思龙研究员两年来的悉心指导、热情关怀和谆谆教诲。本论文研究工作的各个环节，自始至终都是在彭老师的悉心指导下完成的。彭老师渊博的学识、敏锐的洞察力和深邃独到的见解给我极大的启迪。他严谨的治学态度和对我的严格要求将使我终生受益。

作者感谢国家专用集成电路中心主任王东琳研究员的关怀和帮助，他的谦虚和平易近人的品德令人难忘。成德信老师和陈海华老师也在很多方面为作者提供了帮助。

作者感谢国家专用集成电路中心的许多同学，他们是：文伟、肖志云、刘佳璐、崔峰、刘忠轩、沈兵、顾建平、杨明辉、石广建、岳永娟、李长青、张富彬、轩波、李振伟和马鸿等。和他们一起学习和讨论，度过了两年难忘的时间，使作者受益匪浅。

衷心感谢研究生部的李磊老师、邸凌老师、卜树云老师、毛磊老师对作者学习和生活上的关心。

最后，作者要特别感谢父母的养育之恩和妻子的理解和支持。

物理攻击密码系统的防御研究——改进的防御差异能量攻击AES密码系统的定值遮盖法及其安全性能分析

作者：顾晓东
学位授予单位：中国科学院自动化研究所

参考文献(43条)

1. 参考文献

2. R J Anderson. M G Kuhn Tamper resistance - a cautionary note 1996

3. R J Anderson. M G Kuhn Low cost attacks on tamper resistant devices 1997

4. D Noneh. R A Demillo. R J Lipton On the importance of checking cryptographic protocols for fault 1997

5. E Biham. A Shamir Differential fault analysis of secret key cryptosystems 1997

6. P C Kocher Timing Attacks on Implementations ofDiffie-Hellman,RSA,DSS,and Other Systems 1996

7. P Kocher. J Jaffe. B Jun Introduction to differential power analysis and related attacks 1998

8. P Kocher. J Jaffe. B Jun Differential power analysis 1999

9. FIPS 46-3.Data Encryption Standard.Federal Information Processing Standard,National Institute of Standards and Technology 1999

10. Draft of AES - Federal Information Processing Standards Publication

11. Kuo Henry. Ingrid Verbauwhede Architectural Optimization for a

12. 82Gbits/sec VLSI implementation of the AES Rijndael Algorithm

13. Rankland W Effing-Smart Card Handbook 2000

14. R L Rivest. A Shamir. L M Adelman A method for obtaining digitalsignatures and public-key cryptosystems 1978

15. E Biham. A Shamir A new cryptanalytic attack on DES: Differentialfault analysis 1996

16. Y Zheng. T Matsumoto Breaking real-world implementations of cryptosystems by manipulating their random number generation 1997

17. M Joye. A K Lenstra. J -J Quisquater Chinese remaindering based cryptosystems in the presence of faults 1999

18. F Bao. R H Deng. Y Han. A.Jeng,A.D.Narasimbalu, T.Ngair Breaking public key cryptosystems on tamper resistant devices in the presence of transient faults 1997

19. S M Yen. M Joye Checking Before Output May Not Be Enough Against Fault-Based Cryptanalysis 2000(09)

20. I Biehl. B Meyer Differential fault attacks on elliptic curve cryptosystems 2000

21. J F Dhem. F Koeune. P A Leroux. P.Mestre,J.J.Quisquater, J.L.Willems A Practical Implementation of the Timing Attack 1998

22. H Handschuh. H Heys A Timing Attack on RC5 1998

23. A Hevia. M Kiwi Strength of Two Data Encryption Standard Implementation Under Timing Attacks 1999(04)

24. W Schindler ATimingAttack against RSA with the Chinese Remainder Theorem 2000

25. J Kelsey. B Schneier. D Wagner. C.Hall Side Channel Cryptanalysis of Product Ciphers 2000(2-3)

26. F Koeune. J J Quisquater A Timing Attack Against Ri jndael 1999

27. James Alexander Muir Techniques of Side Channel Cryptanalysis 2001

28. T S Messerges. E A Dabbish. R H Sloan Investigations of power analysis attacks on smartcards 1999

29. Eli Biham. Adi Shamir Power Analysis of the Key Scheduling of the AES Candidates 1999

30. Thomas S Messerges. Ezzy A Dabbish. Robert H Sloan Power Analysis Attacks of Modular Exponentiation in Smartcards

31. Jean-Sebastien Coron Resistance Against Differential Power Analysis for Elliptic Curve Cryptosystems

32. Jean McKenna VISA Chip Card Technology Level Definition 2001

33. T Messerges securing the AES Finalists Against Power Analysis Attacks 2001

34. C Clavier. J Corron. N Torh Differential Power Analysisin the Presence of Hardware Countermeasures 2000

35. K Itoh. M Takenaka. N Torh DPA Countermeasure based on the masking Method 2002

36. M Akkar. C giraud An Implementation of DES and AES Secureagainst some Attacks 2001

37. J Golic. C tymen Multiplicative Masking and power Analysis of AES 2003

38. E Trichina. D Seta. L Germani Simplified Adaptive Multiplicative Masking for AES 2003

39. T Messerges using Second -order Power Analysis to Attack DPA Resistant Software 2000

40. S Yen Amplified Differential Power Cryptanalysis on Rijudael Implementations with Exponentential Fewer Power Traces 2003

41. S Chari. C Jutla. J Rao. P. Rohatgi Towards Sound approaches to counter power analysis attacks

42. Hwasun Chang A Study on Securing AES against Differential Power Analysis 2003

43. M Akkar. R Bevan. P Dischamp. D. Moyart Power Analysis 2000