

申请上海交通大学硕士学位论文

一种应用非线性混淆的基于查找表的白盒 AES 实现

论文作者 罗 睿

学 号 1120339105

指导教师 来学嘉

专 业 计算机技术

答辩日期 2015 年 1 月 14 日

Submitted in total fulfilment of the requirements for the degree of Master
in Computer Technology

A Table-based White-box AES Implementation with Non-linear Obfuscation

RUI LUO

Supervisor

Prof. XUEJIA LAI

DEPART OF COMPUTER SCIENCE AND ENGINEERING, SCHOOL OF ELECTRONIC,
INFORMATION AND ELECTRICAL ENGINEERING
SHANGHAI JIAO TONG UNIVERSITY
SHANGHAI, P.R.CHINA

Jan. 14th, 2015

一种应用非线性混淆的基于查找表的白盒 AES 实现

摘 要

白盒密码是一种等价转换技术，它将传统密码算法转换成一种在功能上与之等价但具备在不可信任的运行环境（即白盒环境）中的安全性的算法。它的出现源自于持续增长的对设计和实现能够部署于白盒环境的强密码算法的需求。在白盒环境中，攻击者不仅拥有密码算法的硬件/软件实现，还具有这些算法实现——特别是软件实现——及其运行环境的完全访问及控制的权限。而白盒密码的目的是使得密码算法的白盒实现即使在白盒环境下也能够像黑盒一样运行。也就是说，即使具有对密码算法实现及其运行环境的完全掌控，攻击者仍然无法获得相比于黑盒环境下更多的攻击优势。

自 2002 年，Chow 等人提出第一个白盒实现以来，许多研究者投身于这个领域，提出了许多富有创造性的方案。然而时至今日，大部分白盒实现，尤其对常用密码算法，例如高级加密标准 AES，最终都被攻破了。因此，设计新型白盒实现方案对我们而言不仅是一项挑战，更成为了一个机遇。

本文描述了一种全新的基于查找表的白盒 AES 实现，该实现对 BGE 攻击和 De Mulder 等人针对 Xiao-Lai 白盒 AES 实现的攻击都显示出了很好的防护性。新方案采用了更大的密钥相关查找表 nTMC——一个 16 to 32-bit 的映射——来实现三步密码操作 AddRoundKey、SubBytes 和 MixColumns 的组合。每个查找表 nTMC 都内置了两个轮密钥字节，并且经过随机混淆双射的编码。

除查找表 nTMC 之外，我们设计了 8 to 128-bit 的密钥无关查找表 TSR 来实现 ShiftRows 操作。而 nTMC 和 TSR 以一种立体连接模式进行安全的协同工作。TSR 的输出通过一个由查找表 TXOR 和 TXOR3 所构成的树形网络结构被合并成为一个 128-bit 的值。新方案中全体查找表都经过了非线性的随机编码。

随后，我们对新方案进行了详细的分析，包括性能方面和白盒安全性方面。在性能方面，新方案共计消耗 28444kB 存储空间且在整个加密/解密过程中共有 7776 次查表操作。而在安全性方面，我们首先计算了各类查找表的白盒多样性及白盒含混度，得到了较好的结果。其次，其次，我们阐述了我们认为 BGE 攻

击不适用于新方案的原因：**BGE** 攻击所依赖的最为核心的性质——输出字节对 (y_i, y_j) 之间唯一的线性相关性——在新方案上并不一定能够成立。此外，我们也说明了新方案能够抵御 **De Mulder** 等人的攻击的两点原因：其一，新方案中的所有查找表都加入了非线性输入输出编码，因此 **De Mulder** 等人的攻击所利用的“线性”等价算法无法直接应用在我们的方案中；其二，由于新方案采用了立体的查找表连接结构，攻击者在应用线性等价算法时得到的解集从 2^8 上升到了 2^{24} ，这使得攻击的时间复杂度至少为 2^{64} 。而这在实际应用层面已经能够满足安全性要求。

关键词： 白盒密码 白盒实现 白盒安全性 高级加密标准

A Table-based White-box AES Implementation with Non-linear Obfuscation

ABSTRACT

White-box cryptography is in fact a set of techniques for equivalent transformation, which converts conventional cryptographic algorithms into functionally equivalent alternatives which guarantee solid security under untrusted execution environments, i.e. white-box attack context. Its existence emanates from the pressing demand to design and implement strong cryptographic algorithms that are able to be deployed in white-box context. In such an environment, the attacker is presumed to be in possession of the hardware/software of the cryptographic algorithms, and furthermore have full access to its software implementation and full control over its execution platform. White-box cryptography aims at constructing software implementation behaving as “black box” even executed in white-box environment, making a white-box attacker impossible to gain additional advantages over a black-box attacker.

Starting from 2002, as Chow et al. proposed the first white-box implementation for AES, lots of researchers have dedicated themselves into this field, presenting plenty of creative ideas as well as innovative approaches. However, most of their contributions, especially for those popular algorithm such as AES, are proved to be insecure. Therefore, the design of secure white-box implementations, which is not only challenging but promising, remains to be solved.

This paper describes in detail a new table-based white-box AES implementation, which obtains a sound resistance against the BGE attack and De Mulder et al.’s cryptanalysis against Xiao-Lai scheme. The new scheme exploits larger key-dependent lookup tables $nTMC - 16$ - to 32-bit mappings – to implement the composition of AddRoundKey, SubBytes and MixColumns. Each such table integrates two bytes of the round keys and is blended with random mixing bijections.

Apart of the tables nTMC, 8- to 128-bit key-independent tables TSR are designed to implement the ShiftRows operation and cooperate with nTMC via a sophisticated three-dimensional binding form. The outputs of TSR are merged via the network of tables TXOR and TXOR3. Moreover, random non-linear encodings are applied to all types of tables for better obfuscation.

We then analyse the performance of the new scheme and white-box security against different types of attack and measure two security metrics: the *white-box diversity* and *ambiguity*. From the perspective of performance, the new scheme consumes 28444kB memory for each encryption/decryption procedure, with 7776 table lookups. While from the aspect of security, we get solid results from the calculation of both security metrics. We also believe that the new scheme can withstand the BGE attack due to the lack of the essential property: the unique linear dependency between the output pair (y_i, y_j) . We further illustrate that the scheme can also resist the cryptanalysis of De Mulder et al. since (i) the non-linear encodings are introduced to every table to prevent from the application of “linear” equivalence algorithm and (ii) the well-defined binding method between nTMC and TSR is irreducible and enlarges the size of solution set from 2^8 to 2^{24} , resulting in the complexity of at least 2^{64} , which is satisfactory for applications.

KEY WORDS: white-box cryptography white-box implementation white-box security AES

目 录

插图索引	vii
表格索引	ix
主要符号对照表	xi
第一章 绪论	1
1.1 研究的背景以及意义	1
1.1.1 密码学的发展以及演进	2
1.1.2 分组密码白盒实现及其最新进展	4
1.2 本文的研究内容及目标	5
1.3 文章结构	6
第二章 分组密码和白盒密码及其安全性	7
2.1 分组密码	7
2.1.1 高级加密标准 AES	7
2.2 密码算法的安全性	11
2.2.1 无条件安全性	11
2.2.2 计算安全性	12
2.3 Kerckhoffs 假设	13
2.4 攻击模型	13
2.4.1 黑盒模型	14
2.4.2 灰盒模型	15
2.4.3 白盒模型	15
2.5 白盒密码	16
2.6 白盒安全性	17
2.7 本章总结	17

第三章 基于查找表的白盒 AES 实现及其相应的攻击	19
3.1 Chow 等人的白盒 AES 实现	19
3.2 BGE 攻击	22
3.3 Xiao-Lai 白盒 AES 实现	23
3.4 De Mulder 等人的攻击	26
3.5 本章总结	27
第四章 一种新的基于 16 to 32-bit 查找表的非线性混淆方案	29
4.1 步骤 3 中使用的查找表 nTMC	29
4.2 步骤 1 中使用的查找表 TSR	30
4.3 步骤 2 中使用的查找表 TXOR 和 TXOR3	33
4.4 解密过程	36
4.5 本章总结	37
第五章 新方案的性能及白盒安全性分析	39
5.1 算法实现的性能	39
5.2 算法实现的白盒安全性衡量指标	40
5.2.1 算法实现在 BGE 攻击下的安全性	41
5.2.2 算法实现在 De Mulder 等人提出的攻击下的安全性	42
5.3 本章总结	43
全文总结	45
参考文献	47
致谢	51
攻读学位期间发表的学术论文目录	53
攻读学位期间参与的项目	55

插图索引

2-1 SubBytes 操作示意图	8
2-2 ShiftRows 操作示意图	9
2-3 MixColumns 操作示意图	9
2-4 AddRoundKey 操作示意图	10
2-5 标准的 AES 加密过程	11
2-6 标准的 AES 解密过程	12
3-1 一个等价的 AES 加密过程	21
3-2 查找表 TMC 的结构	25
3-3 Xiao-Lai 白盒 AES 实现的结构	26
3-4 查找表 TMC 的结构	27
4-1 加密过程的流程结构	30
4-2 表 nTMC 的结构	31
4-3 表 TSR 的结构	34
4-4 nTMC 和 TSR 的连接方式	35
4-5 TXOR 和 TXOR3 的结构	35
4-6 一个等价的 AES 解密过程	36

表格索引

5-1 新方案所包含的查找表的数量和空间占用量	40
-----------------------------------	----

主要符号对照表

\cdot	矩阵乘法
\circ	级联, 定义为 $Y \circ X = Y(X(\cdot))$
\otimes	在 $\mathbb{GF}(2^8)$ 上的乘法运算
\oplus	在 $\mathbb{GF}(2^8)$ 上的加法运算以及异或运算
$\oplus_c(x)$	异或函数 $\oplus_c(x) = x \oplus c$
\wedge	逻辑运算: 与 (AND)
$ $	数据值并联
\parallel	查找表并联
$\text{diag}(A, \dots, C)$	以 A, \dots, C 构成的分块对角矩阵
$\mathbb{GF}(p^n)$	伽罗瓦域 (Galois Field), 即有限域, 阶数 p^n
$ \mathcal{S} $	集合 \mathcal{S} 的大小
S	AES 中使用的 Rijndael S-box
MC	有限域 $\mathbb{GF}(2^8)$ 上的 4×4 -byte 非奇异矩阵, 用来表示 MixColumns 操作
$mc_{i,j}$	矩阵 MC 第 i 行 j 列的系数
SR	有限域 $\mathbb{GF}(2)$ 上的 128×128 -bit 非奇异矩阵, 用来表示 ShiftRows 操作
sr	ShiftRows 操作, 定义为 $sr(i, j) = (j - i) \bmod 4$
isr	InvShiftRows 操作, 定义为 $isr(i, j) = (j + i) \bmod 4$
$\bigoplus_{k=0}^n$	从下表 0 到 n 的连续异或: $\bigoplus_{k=0}^n val_n = val_0 \oplus val_1 \cdots val_n$

第一章 绪论

1.1 研究的背景以及意义

我们身处的世界自始至终都充斥着大量的信息，而人类的活动也始终是以这些信息为核心的。早年间，人们已经在信息处理以及信息交换的过程中充分的认识到了信息安全的重要性。起初，信息安全是作为政府工作和军事层面的需求而得到关注的，这是因为政府和军队需要保护保密文件以及敏感信息的内容使其免于泄露。最有名的例子莫过于著名的“凯撒密码”（Caesar cipher），它是由罗马帝国皇帝朱利乌斯·凯撒发明的用于保护军事机密安全性的方案。

自从人类迈入信息时代，信息的总量已经呈现出了爆炸式的增长。现如今，我们可以毫不犹豫的说，信息之于我们的生活就好像空气、水、阳光之于生命，它是信息时代的现代文明这个“生命体”赖以生存的要素之一。得益于最近几十年以来信息技术的飞速发展，个人电脑完成了对人们日常生活的广泛渗透，互联网业已将全世界人类无时差的连接到了一起。我们已经能够通过数字化的方式来进行信息处理和信息交换，而这无疑极大地提高了效率，使得我们的生活变得更加美好。也正因为信息处理和信息交换的环境发生了改变，人们对信息安全的需求以及确保信息安全的方法也相应的产生了巨大的变化。除此之外，得益于通信领域大量的技术进步，例如互联网和数字移动网络的快速拓展，全球范围内数量庞大的参与者都能够通过通信设备互联互通。数以亿计的终端用户通过他们的手中的已经连入网络的设备完成日常工作：接打电话、收发电子邮件和短信、在线金融交易等，而巨量用户之间的互相关联让许多机构意识到了此间存在的巨大商业价值。数字化的娱乐服务，例如数字电视、能够购买音乐或者租赁电影的虚拟多媒体商店，都纷纷出现。这些在数字化信息基础架构之上产生的新生事务和潮流在现如今我们的市场生活中已经扮演了举足轻重的角色，并且其重要性仍然日益增加。

虽然将世界各地的许许多多参与者连接起来显然是大有裨益的，但我们仍然不能无视这其中的风险。举例而言，很多数字信息，比如经由无线连接传输的银行交易信息，是极易被截获而进一步被篡改的。而参与通信的各方，有可能是可信的，例如政府机关或者合法组织或机构，也有可能是不可信任的，如

大部分的终端用户，因为我们无法预测这些用户是否会实施有害行为。正因如此，保护数字信息的安全就成为了至关重要的问题。对于这个问题的研究我们称为密码学。现代密码学表明对信息安全的需求已经不再局限于政府工作和军事，而是已经扩展到人类生产生活的方方面面。

1.1.1 密码学的发展以及演进

通常，密码学包含两个互补的分支：密码编码学和密码分析学。密码编码学家研究如何设计能够保护在不安全信道上传输的信息的方法；而密码分析学家则致力于分析这些保密方法的安全性水平。建立这两个分支之间的紧密关联是非常重要的，因为密码算法的安全性经常取决于多年来在密码分析中取得的成果。

密码编码学是设计用于在攻击者存在的情况下保护数据及通信安全的密码算法的一门学科 [1]。通信的双方，即消息的发送者和接收者，都不得不在一个不安全的信道上交换机密信息。他们希望这些机密在任何通过窃听信道而获取消息的第三方（即攻击者）看来都是毫无意义或者说不理解的。为了达到这样一个目的，通信的双方共同选定一套密码算法，协商好一些仅有双方知道的秘密信息，由发送方将原始的机密信息——即“明文”（plaintext）——经过特定的变换——即“加密”（encrypt）——转换成为攻击者不理解的形式——即“密文”（ciphertext）——再经由不安全的信道传输至接收方，而接收方在收取密文之后便将密文经过逆变换——即“解密”（decrypt）——转换回明文。实际上，在不安全的信道上传输的是密文而不是明文。而对于被截获的密文，我们假定若不知道通信双方协商的秘密信息，任何人都不可能从中获得明文。这些通信双方共享的秘密信息就被称为密钥。如果用于加密和解密过程的密钥是相同或能够通过一系列简单变换相互推导的，那么这一套密码算法就被称为对称密码算法。对称密码算法的加密和解密过程可以用如下公式表示：

$$\text{Encrypt}_{key}(\text{plaintext}) = \text{ciphertext}$$

$$\text{Decrypt}_{key}(\text{ciphertext}) = \text{plaintext}$$

在本文中，我们所研究的“高级加密标准”（Advanced Encryption Standard, AES）算法就是使用相同的加密解密密钥的对称密码算法。

在设计密码算法的时候，尽可能真实的估计攻击者的能力是非常重要的。攻击者的能力一般会被归纳和总结为几种不同的攻击模式。在上世纪九十年代

之前，现代密码学都假设信道的端点是可信的，这意味着密码算法始终会在一个安全的环境中运行，这个假设被称为“黑盒模型” (black-box model)。在黑盒模型中，攻击者仅仅能够窃听或者修改在可信的通信各方之间的不安全的信道上传输的信息，故至多能够掌握密码算法的输入和输出特性。因此，直到 1990 年代，算法设计的核心思想都是在可信平台上部署已公布的高安全性密码算法，而将信息的安全寄托于密钥的安全性上。

然而，自 1990 年代下半叶以来，由于计算机、通信设备取得的突破以及互联网、移动网络对数以亿计的用户的相互连接，攻击方式和攻击场景都已发成了巨大的变化，传统的黑盒模型已经无法准确的刻画攻击者的能力。在不安全的设备（如受到恶意软件、木马、病毒感染的计算机）上部署使用密码组件的应用的情况已经被愈来愈多普遍。而对于了解了现状的终端用户，很多已经有能力利用不安全的终端实施恶意行为，即他们是潜在的攻击者，而我们并没有办法对此作出预测。在这种情况下，黑盒模型假设中可信的信道端点已经不复存在，它们甚至有可能是经过攻击者设计的、专门用于窃听的设备。至此，我们可以宣告传统的黑盒模型已经不再满足新时期中安全模型的要求。在新形势下，我们迫切的需要能够刻画能力更强大的攻击者的实际攻击模型。

在实践中，一套密码算法会使用硬件或者软件方法来实现，并经常运行在不可信任的开放平台上，例如算法实现运行在某个终端用户所拥有并且由他控制的电子设备上，在这种情况下，终端用户具有对密码算法的硬件/软件实现的完全/部分控制权限。这显然超出了黑盒模型中攻击者的能力范畴。实际上，黑盒模型的缺点早在 1996 年就已经被 Kocher 通过所谓的“算法实现攻击 (implementation attacks)”所强调 [2]。算法实现攻击利用某种“算法实现特有信息” (implementation-specific) 来对特定的算法实现进行分析，如果这些信息是不经意间从正在运行的密码算法实现中泄露出来的，那么它们又被称为“边信道信息” (side-channel information)。边信道的例子有很多，典型的就执行时间和功耗。利用这些算法实现特有信息或者边信道信息的攻击模型被称为“灰盒模型” (gray-box model)。

随着各式各样的分析手段的发展，日益增长的攻击者的能力已经不再局限于获取和利用类似于边信道信息这样的边缘信息，而是逐渐拓展到针对算法实现中的核心信息。因此，一个新的攻击模型就应运而生，我们将其称为“白盒攻击环境” (white-box attack context) [3] 或简称为“白盒模型” (white-box

model)。该模型相比灰盒模型更强，被认为是从攻击者角度而言的最强的攻击模型，它关注运行在不可信任的开放平台上的密码算法的软件实现。这些平台被潜在的恶意终端用户完全控制，它们可能是笔记本电脑、智能手机或机顶盒。在白盒攻击环境下，攻击者对于密码算法的软件实现及其运行环境都具有完全的访问及控制权限。这意味着他可以利用具有断点插入功能的调试器来观察和修改软件实现运行过程中的中间值，也可以在内存中搜索储存的密钥 [4]，甚至还可以通过急速冷却的方法保留内存的状态 [5]。对于在设计时仅考虑部署在黑盒模型中的传统密码软件实现而言，白盒模型中攻击者的能力是真正的威胁。由于除密钥之外密码算法的其他内部细节在大部分情况下都是已知的，如何用软件方法实现在白盒模型中仍然安全的密码算法就变得至关重要了，而“白盒密码”（white-box cryptography）作为一个研究课题就关注这个问题。

白盒密码旨在以一种面对白盒模型中的攻击依然能够保证足够高安全性水平的方法来构造密码算法的软件实现，即保护密钥的保密性使其不致泄露。而设计白盒密码的最终目标是让白盒模型中的攻击者相比黑盒模型中的攻击者在提取密钥方面不具有任何优势，也即是说，不论攻击者是强大到具有对密码软件实现的完全访问及控制权限还是仅仅只能观察算法实现的输入输出特性，攻击的难度都没有差异。

灰盒模型和白盒模型这两个在新时期更为真实的攻击模型的出现说明我们必须把安全的密码算法实现，不管是硬件的还是软件的，放在与密码算法的黑盒安全特性同等重要的地位上来考虑，因为攻击者必然会试图利用整个安全性链条上最薄弱的一环。一旦攻击者对密码硬件/软件具备了完全权限，最薄弱的一环将很有可能出现在具体的算法实现的环节上。也正因此，白盒密码的研究在现阶段以及未来相当长的时间内都将是重要的并且意义深远的研究课题。

1.1.2 分组密码白盒实现及其最新进展

分组密码作为对称密码算法的一种，与另一种流密码一样，一直以来被广泛研究和应用。在过去的许多年中，数据加密标准 DES (Data Encryption Standard) [6] 和其后继者高级加密标准 AES [7]，作分组密码中的两个最常用的方案得到了非常广泛的使用。现今各种加密软件或者包含了密码组件的应用无不内置这两种算法。而应用白盒密码的研究成果，通过软件实现的方法构造的加密算法实现被称为“白盒实现”（white-box implementation）。

2002 年, Chow、Eisen、Johnson 和 van Oorschot 引入了白盒密码这个研究课题 [3, 8] 并且提出了一套系统的基于查找表的构造实际的分组密码白盒实现的通用方法。他们将提出的方法应用在 DES 和 AES 上, 设计了最初的白盒 DES 实现 [3] 和白盒 AES 实现 [8]。但是随后的研究发表了许多针对这两个白盒实现的攻击 [9–14], 这些攻击表明 Chow 等人设计的两种白盒实现方案无法保护内嵌的密钥不被提取, 因而在白盒模型中是不安全的。其中, 由 Billet、Gilbert 和 Ech-Chatbi 所提出的一种针对 Chow 等人的白盒 AES 实现的有效攻击 [13]——即 BGE 攻击, 以它的三个发明者的首字母命名——成功的以 2^{30} 的复杂度将密钥信息恢复出来。2013 年, De Mulder 等人重新改良了 BGE 攻击, 在加入了 Tolhuizen 所提出的改进 [15] 之后, 进一步的将复杂度降低到 2^{22} 。BGE 攻击引发了学界对设计具有更高白盒安全性的新型白盒 AES 实现的研究。

由于 BGE 攻击的启发, 三种全新的白盒 AES 实现随即被发表: Bringer、Chabanne 和 Dottax 于 2006 年提出的对 AES 的一个变种引入扰动的方案 [16]; Xiao 和 Lai 于 2009 年发表的基于更大线性编码的方案 [17]; Karroumi 于 2010 年发表的利用 AES 对偶密码对 Chow 等人的白盒 AES 实现进行改良的方案 [18]。这三个方案都被宣称在白盒模型中是安全的, 因为它们能够抵御 Billet 等人所提出的攻击。其中, Bringer 等人的方案被 De Mulder、Wyseur 和 Preneel 于 2010 年攻破 [19], Xiao 和 Lai 的白盒方案被 De Mulder、Poelse 和 Preneel 于 2013 年攻破 [20], Karroumi 的白盒方案也于 2013 年被 De Mulder、Poelse 和 Preneel 攻破 [21]。

1.2 本文的研究内容及目标

本文旨在学习最前沿的白盒密码理论、白盒实现方案以及白盒攻击方法, 研究前人工作中值得借鉴的地方, 同时也分析其中的缺点, 并且最终根据这些知识和经验设计新的白盒实现方案。

具体来说, 我们将提出一种新型的基于查找表的白盒 AES 实现方案, 该方案能够抵御 BGE 攻击和 De Mulder 等人针对 Xiao-Lai 白盒 AES 实现的攻击。在我们的方案中, AES 轮函数的四种操作与额外添加的随机映射一起, 经过精密的构造被转换成为三种不同类型的查找表。对于与密钥相关的查找表, 我们在生成的过程中, 利用了更大的、内置了双字节轮密钥的映射; 而对于与密钥无关的查找表, 我们在保持高水平的安全性的前提下尽可能的减少性能损耗。

此外，我们设计了非常精密的查找表级联方法，目的是为了避免在不同类型查找表的连接边界上出现安全隐患。为了使查找表获得更好的混淆性，我们还在全体查找表中加入了随机非线性编码。在设计完成之后，我们对方案的安全性进行了分析和评估。在评估过程中，我们着重分析了方案在通用的评测指标下的安全性以及面对特定攻击的抵御能力。

1.3 文章结构

本文一共包含七个章节：

1. 绪论。这一章阐述了白盒密码的研究背景及其意义，简述了现阶段该领域的最新进展，并且指出了本文的研究内容及目标。
2. 分组密码和白盒密码及其安全性。在这一章中，我们将阐述分组密码、白盒密码的相关概念和定义，并且介绍与二者相关的一些安全性方面的基本概念。
3. 基于查找表的白盒 AES 实现及其相应的攻击。这一章介绍两种基于查找表的白盒实现方案，它们分别是 Chow 等人提出的白盒 AES 实现和 Xiao-Lai 白盒 AES 实现。与此同时，我们还将介绍对这两种方案的攻击：Billet 等人对 Chow 等人的白盒 AES 实现的攻击，即 BGE 攻击；De Mulder 等人对 Xiao-Lai 白盒 AES 实现的攻击。
4. 一种新的基于 16 to 32-bit 查找表的非线性混淆方案。这一章提出一种新的白盒 AES 实现方案，该方案基于更大的查找表：16 to 32-bit 查找表，并且利用了非线性混淆的技术，来抵御已知攻击。
5. 新方案的性能及白盒安全性分析。这一章分析新方案的性能及其白盒安全性。分析的内容包括：性能方面，算法的存储消耗以及对内存的访问量；在白盒安全性方面，衡量方案的白盒安全性指标，以及它在特定攻击下对密钥的保护情况。
6. 全文总结。这一章对论文中的工作做了归纳总结，并在此基础上展望了未来可能进行的研究。

第二章 分组密码和白盒密码及其安全性

2.1 分组密码

在密码学中，分组密码指的是一类通过一套由某个对称密钥唯一确定的加密解密变换对长度固定的数据序列——即分组——进行密码操作的确定性密码算法。分组密码与流密码和消息验证码一样，都采用对称密钥，故都属于对称密钥算法。这类算法的特点是用户（即通信各方）在使用算法进行加密或解密的过程中共享相同的密钥材料。而所谓共享相同的密钥材料，既可以是通信各方使用相同的密钥，也可以是各方使用不同的密钥但密钥之间能够通过一系列简单变换相互推导。由于能够高效的运行，对称密码算法作为强有力的工具被广泛的应用于是消息加密/解密以及验证中。而分组密码，作为对称密码最重要的分支，出现在几乎所有实际的密码组件中。本文纯粹关注分组密码。

2.1.1 高级加密标准 AES

1997 年，美国国家标准技术研究所（National Institute of Standards and Technology, NIST）开始征集提案，目的是征集一种分组密码方案作为新的加密标准。而这个标准也就是众所周知的高级加密标准 AES，人们希望用它替代已经不能满足需求的数据加密标准 DES。在 2000 年的时候，由 Daemen 和 Rijmen 所设计的分组密码 Rijndael[22, 23] 被选定为作为 AES 的算法，并于 2001 年作为 FIPS 197[7] 被公之于众。AES 的寿命预计能够持续 20 到 30 年。

AES 是一种分组长度为 128-bit 的密钥迭代分组密码，其基本结构是“代换-置换网络”（substitution-permutation network, SPN）。它的三个版本是 AES-128、AES-192 和 AES-256，分别支持长度为 128-bit、192-bit 和 256-bit 的密钥。通常，AES 包含 R 轮轮函数和 $R+1$ 个由主密钥通过 AES 密钥编排算法生成的 128-bit 轮密钥，而 R 的值为 10, 12, 14，分别对应 AES-128、AES-192 和 AES-256。AES 在运行过程中会维护一个 $4 \times 4 = 16$ 字节的状态矩阵，状态矩阵的初始状态和最终状态分别对应算法的输入和输出，每一步密码操作都作用在状态矩阵上并且更新状态值。

AES 的轮函数包含下列四种密码操作：

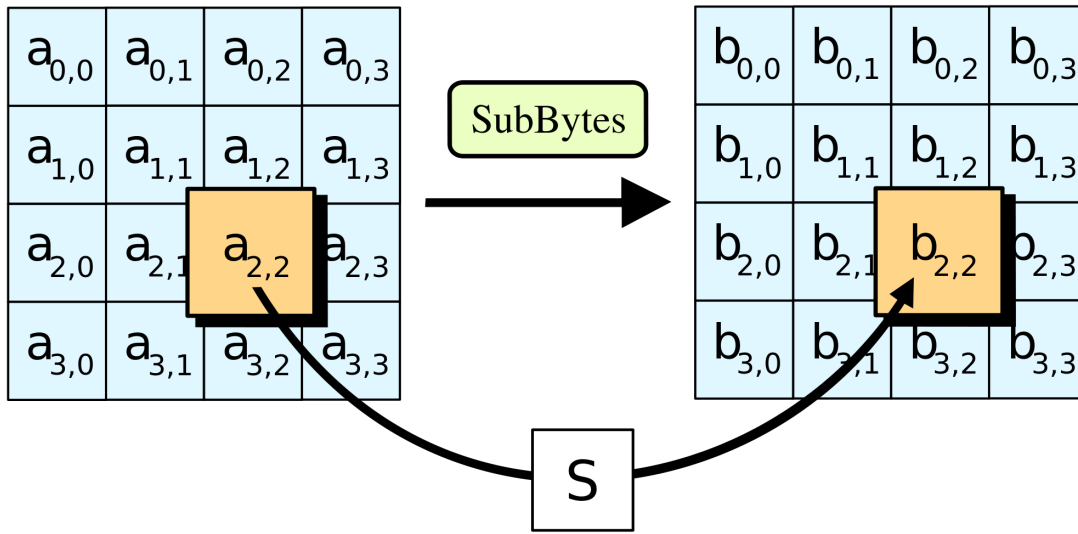
图 2-1 SubBytes 操作示意图¹.

Fig 2-1 Diagram of SubBytes.

1. 字节代换SubBytes。该操作利用 8-bit 的非线性双射 Rijndael S-box，记作 S ，对状态矩阵中的每一个字节进行代换，即将 x 代换为 $S(x)$ ，来为算法提供非线性性。

SubBytes 操作的示意图如图2-1所示。

2. 行位移ShiftRows。该操作对状态矩阵的每一行进行以字节为单位的循环位移。具体来说，对于 $0 \leq i \leq 3$ ，ShiftRows 操作将状态矩阵的第 i 行向左进行 i 个单位的循环位移。
3. 列混淆MixColumns。该操作是一个作用在状态矩阵每一列上的线性变换。利用一个在有限域 $\text{GF}(2^8)$ 上的 4×4 的可逆矩阵 MC ：

$$\text{MC} = \begin{bmatrix} \text{'02'} & \text{'03'} & \text{'01'} & \text{'01'} \\ \text{'01'} & \text{'02'} & \text{'03'} & \text{'01'} \\ \text{'01'} & \text{'01'} & \text{'02'} & \text{'03'} \\ \text{'03'} & \text{'01'} & \text{'01'} & \text{'02'} \end{bmatrix}$$

¹来自 Wikipedia

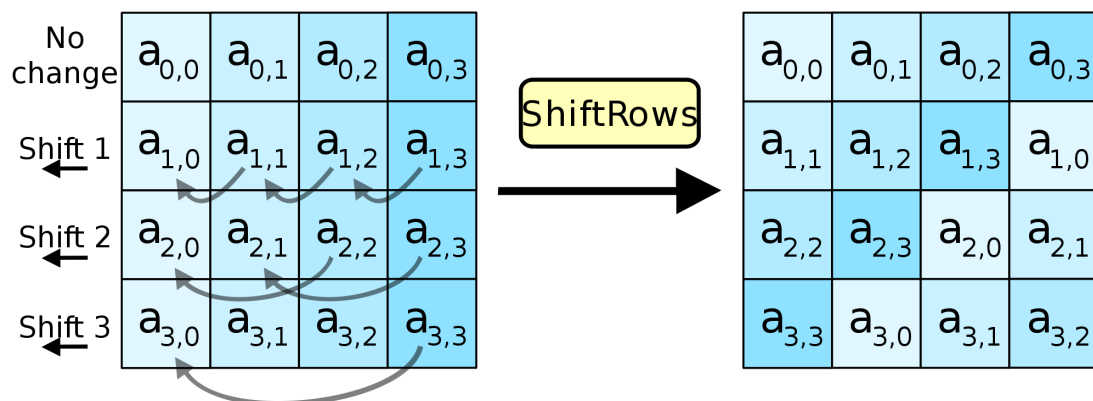
图 2-2 ShiftRows 操作示意图¹.

Fig 2-2 Diagram of ShiftRows.

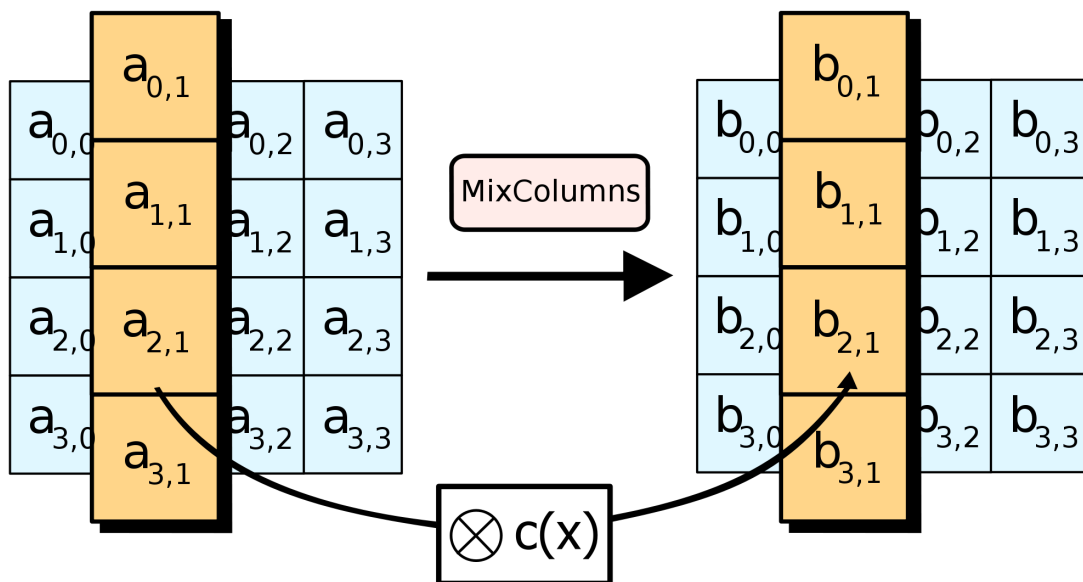
图 2-3 MixColumns 操作示意图¹.

Fig 2-3 Diagram of MixColumns.

MixColumns 操作将状态矩阵的第 j 列 col_j ($0 \leq j \leq 3$) 与矩阵 MC 相乘获得第 j 列的更新值 col_j^{new} :

$$col_j^{new} = MC \cdot col_j$$

MixColumns 操作的示意图如图2-3所示。

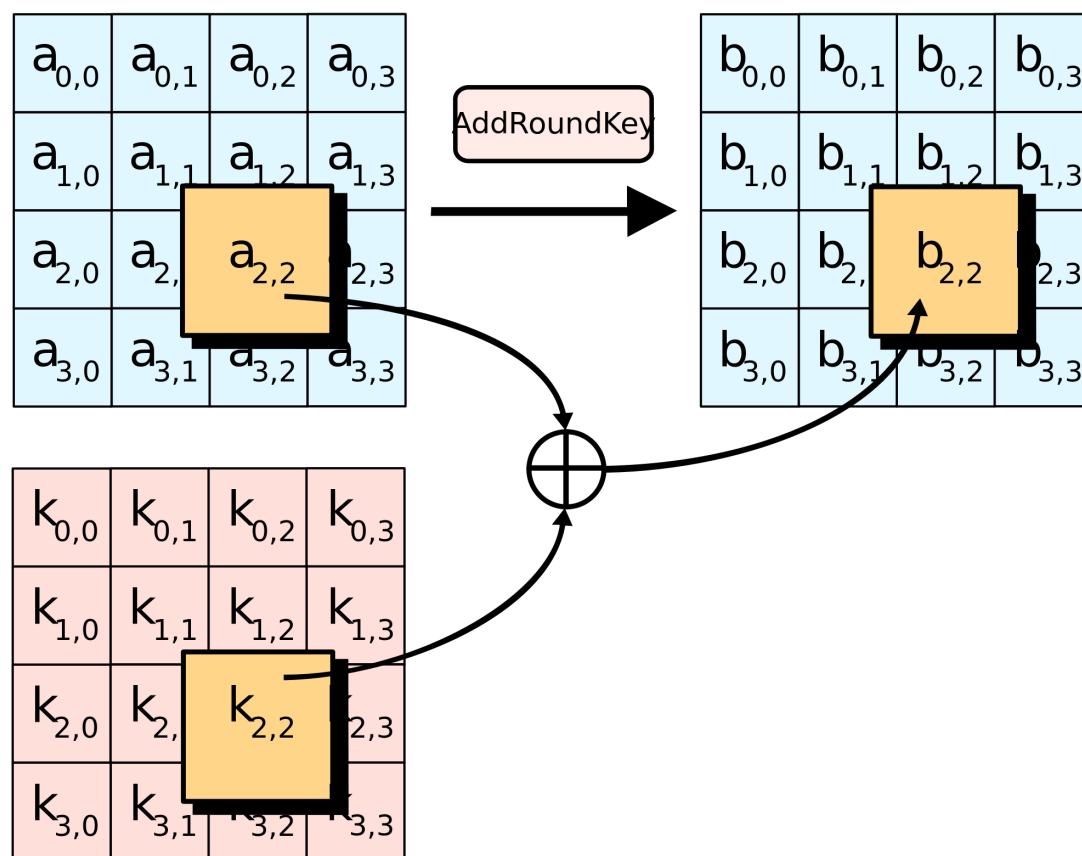
图 2-4 AddRoundKey 操作示意图¹.

Fig 2-4 Diagram of AddRoundKey.

4. 轮密钥加AddRoundKey。该操作将 128-bit 即 16 个字节的轮密钥与状态矩阵的 16 个字节对应进行模 -2 加法，即对每 bit 进行异或。AddRoundKey 操作的示意图如图2-4所示。

根据 SPN 的结构，“代换”结构，即混淆层，实际上就是 SubBytes 操作，而“置换”结构，即扩散层，则是由 ShiftRows 和 MixColumns 操作组成。AES 混淆层的巧妙设计使得数据在两轮混淆之后就能达到“全混淆”状态。

图2-5和图2-6分别描述了 AES-128 的标准加密和解密过程。

```

Require:  $plaintext, k^r$  ( $1 \leq r \leq 11$ )
Ensure:  $ciphertext$ 
 $state \leftarrow plaintext$ 
 $state \leftarrow \text{AddRoundKey}(state, k^1)$ 
for  $r \leftarrow 1$  to 9 do
     $state \leftarrow \text{SubBytes}(state)$ 
     $state \leftarrow \text{ShiftRows}(state)$ 
     $state \leftarrow \text{MixColumns}(state)$ 
     $state \leftarrow \text{AddRoundKey}(state, k^{r+1})$ 
end for
 $state \leftarrow \text{SubBytes}(state)$ 
 $state \leftarrow \text{ShiftRows}(state)$ 
 $state \leftarrow \text{AddRoundKey}(state, k^{11})$ 
 $ciphertext \leftarrow state$ 
return  $ciphertext$ 

```

图 2-5 标准的 AES 加密过程.

Fig 2-5 Conventional AES encryption process.

2.2 密码算法的安全性

在设计和分析密码算法时,我们需要考虑期望达到的安全性水平。一般情况下,我们将密码算法的安全性分三类:无条件安全性、计算安全性和可证明安全性。下面我们详细介绍前两类。

2.2.1 无条件安全性

作为最高的安全性水平,无条件安全性由 Shannon 于 1949 年提出 [24],其定义如下:

定义 2.1 (无条件安全性). 一套密码算法是无条件安全的当密文不泄露任何与明文相关的信息,或等价的,当明文与密文是统计独立的。

无条件安全性也称为完全安全型,在此安全性下,即使攻击者具有无限的计算能力也无法获取有用信息。Shannon 已经证明:在密钥随机选取并且不可

```

Require:  $ciphertext, k^r$  ( $1 \leq r \leq 11$ )
Ensure:  $plaintext$ 
 $state \leftarrow ciphertext$ 
 $state \leftarrow \text{AddRoundKey}(state, k^{11})$ 
for  $r \leftarrow 9$  to  $1$  do
     $state \leftarrow \text{InvShiftRows}(state)$ 
     $state \leftarrow \text{InvSubBytes}(state)$ 
     $state \leftarrow \text{AddRoundKey}(state, k^{r+1})$ 
     $state \leftarrow \text{InvMixColumns}(state)$ 
end for
 $state \leftarrow \text{InvShiftRows}(state)$ 
 $state \leftarrow \text{InvSubBytes}(state)$ 
 $state \leftarrow \text{AddRoundKey}(state, k^1)$ 
 $plaintext \leftarrow state$ 
return  $plaintext$ 

```

图 2-6 标准的 AES 解密过程.

Fig 2-6 Conventional AES decryption process.

重复的情况下, Vernam 密码, 即“一次一密”(One-time pad), 能够提供无条件安全性 [25]。此外, 他还证明了如果想要达到无条件安全性, 密钥的熵必须不低于明文的熵。这意味着密钥的长度必须不短于明文的长度, 并且密钥不得重复使用。

在分组密码中, 想要达到无条件安全性则必须使用理想分组密码。但是理想分组密码仅在实现上就是不现实的, 这是因为它需要能够储存 2^{n_k} 个在有限域 $\mathbb{F}_2^{m_b}$ 上的随机置换的空间。而一般实际的分组密码通过复杂的构造来近似理想分组密码。这种做法导致熵降低。因此, 实际的分组密码至多能够达到计算安全性。

2.2.2 计算安全性

相比在无条件安全性下假设攻击者具有无限的计算能力, 我们可以更符合实际情况的假设攻击者具有有限的能力, 也就是说他所能利用的计算资源是有

限的。这个假设引出了计算安全性的概念。

定义 2.2 (计算安全性). 一套使用 n_k -bit 密钥的分组密码是计算安全的当不存在任何对针该算法的复杂度低于穷举密钥搜索（即 2^{n_k} ）的攻击。

在上述定义中，攻击的复杂度指的是时间复杂度（即工作因数）、存储复杂度（即存储需求和内存访问量）以及数据复杂度（即数据种类和数据量）三者在预计算阶段和正式工作阶段的综合。

我们注意到，定义2.2指出为了确定一套分组密码是计算安全的，我们需要确保所有针对该分组密码的攻击的复杂度都高于某个特定的值。这实际上是难以实现的，因为我们不能保证检验所有的攻击方法。因此，在实践中，我们一般考虑在全体现有攻击中选择一个良好定义的、具有代表性的子集，并检验子集中的方法在攻击特定分组密码时的复杂度，并最终检验的结果来判定该分组密码是否具有计算安全性。

2.3 Kerckhoffs 假设

1883 年，Kerckhoffs 提出了当时在军用密码设计中应当遵守的六项原则。其中的第二条就是广为人知的 Kerckhoffs 假设，也称为 Kerckhoffs 原则。它在今天依然具有举足轻重的意义。

Kerckhoffs 提出，即使算法的每个细节都被公开，只要密钥没有泄露，密码算法就仍然应当保持其安全性。也就是说，密码算法的安全性应当完全依赖于密钥的保密而不是算法细节的保密。对于密码算法的设计者而言，最好的办法是先假定算法的细节已经为攻击者所知，然后在此假设下设计密码算法，这样一来，所设计的算法的安全性不依赖于算法细节的保密。在设计完成之后，公布细节与否便由设计者自由决定。

2.4 攻击模型

攻击模型是对攻击者攻击能力的归纳和总结，目的在于系统的阐述部署及运行密码算法的不安全的环境，并可以用于安全性水平评估。这对设计新型密码算法或者通过硬件/软件方法实现已有的密码算法而言都是不可或缺的。我们将介绍三种主要的攻击模型：黑盒模型、灰盒模型和白盒模型。

2.4.1 黑盒模型

在黑盒模型中，位于信道端点的用户（即发送者和接受者）都是经过认证的，是可信的；而信道是不安全的。

黑盒模型是最为保守的攻击模型，它假设攻击者仅能获取密码算法的输入及输出信息，其中输入信息相当于明文或者密钥而输出信息则相当于密文。根据攻击者能够获得的信息种类（输入、输出或二者皆有），以及他能够在信息上实施的操作（读、写或二者皆有），我们能够对黑盒模型中可能出现的攻击方式做一个分类：

- **唯密文攻击。**攻击者仅有对密文的只读权限。该攻击是黑盒模型中最弱的攻击类型，如果一套密码算法无法抵御该攻击，那么它就注定是失败的设计。
- **已知明文攻击。**攻击者具有对明文-密文对的只读权限。线性攻击属于该类型 [26]。
- **选择明文攻击。**攻击者具有对明文的写权限以及对相应密文的读权限。差分攻击属于该类型，因为选定特定形式的明文的能力对于差分攻击而言是必要条件 [27]。
- **选择密文攻击。**该攻击与选择明文攻击相似，不同之处在于攻击者具有对密文的写权限以及对相应明文的读权限。该攻击需要攻击者具有对解密过程的访问权限。
- **自适应选择明文/密文攻击。**与选择明文/密文攻击相似，不同之处在于攻击者能够根据攻击过程中先前获得的结果调整往后的选择。

上述五种攻击类型的攻击强度时依次增加的，在这五种类型中，攻击者的目标是提取密钥。

当然，黑盒模型中的攻击类型不仅局限于上述五种。由于密钥也被算作输入信息的一种，在设计基于分组密码的散列函数时，已知密钥攻击和选择密钥攻击 [28] 经常作为合适的攻击类型被采用。在这两种攻击类型中，密钥对攻击者而言是已知的或在其控制之下的。攻击者的目的是分析散列函数的输入或者输出。

2.4.2 灰盒模型

在真实情况下，密码算法总是以硬件或软件方式实现在物理设备上。在黑盒模型中，我们假设这些实现表现得像理想黑盒，能够防止任何对其内部数据或运行情况的观察和篡改。因此，攻击者的能力被限制在对黑盒的输入和输出的观察中。这常常使得攻击变得复杂，并且必须付出高昂的计算成本。

然而理想黑盒的假设其实是不现实的。实际上，假设信道端点的用户与信道一样是不可信任的，相比黑盒模型假设他们可信，更贴近事实。而这种假设就被称为“灰盒模型”。在灰盒模型中，攻击者的能力拓展到了对密码算法实现的有限接触。这使得攻击者能够获得的信息在黑盒模型的基础上有了极大的扩展。这些多获得的信息，一般来说是与密钥关系较弱的“算法实现特有信息”，不过即便如此，攻击者成功攻击的可能性也提高了很多。灰盒模型中的攻击一般针对密码算法的实现而不是算法本身，故被称为“算法实现攻击”。而被动式非入侵性的算法实现攻击，即“边信道分析攻击” (side-channel analysis attacks)，由 Kocher 于 1996 年提出 [2]，因其低成本和不可检测的特性成为灰盒模型下威胁性最大的攻击之一。而被边信道分析攻击所利用的算法实现特有信息又被称为边信道信息，这些信息往往指真实世界中算法实现在运行中通过边信道在不经意间泄露的信息，这些边信道可能是执行时间、功耗、电磁辐射等等。

2.4.3 白盒模型

当攻击者能够对物理设备进行干预并从中获取与密钥相关的边信道信息的时候，在上一小节中介绍的灰盒模型便是威胁最严重的攻击模型。但当攻击的对象从密码算法的硬件实现转换到软件实现的时候，灰盒模型的威力就减弱了很多，因为攻击者仅仅能够从类似于执行时间之类的边信道中获得信息。在实践中，由于软件实现所运行的平台的不安全性，攻击者对软件实现的掌控程度经常远远超出灰盒模型的定义：攻击者对于密码算法的软件实现及其运行环境都具有完全的访问及控制权限，能够随意观察或修改软件实现在运行过程中的中间值或者在内存中搜索密钥。而用这些手段获得的信心的危险性远非那些无意间泄露自边信道的信息可比。因此，一种新的“真实的”攻击模型由 Chow、Eisen、Johnson 和 van Oorschot 于 2002 年提出，他们将其命名为“白盒攻击环境”，简称为“白盒模型”。

在白盒模型中，终端用户像在灰盒模型中一样被认为是不可信任的，此外还具有两种能力：

1. 对密码软件实现的完全访问与控制权限
2. 对运行软件实现的平台的完全掌控

攻击者能够实施的攻击方式包括：

- 静态分析。通过使用反汇编器或反编译器，攻击者能够对软件实现进行逆向工程并且有针对性的篡改软件实现的内容以增加攻击效率。
- 动态分析。通过使用具有断点插入功能的调试器，攻击者能够观察任意中间值并且随意修改，即注入精心选取的错误，还能够在任意时间任意地点启动或停止软件的执行。
- 在内存中搜索密钥

上述的三种攻击方式是白盒模型中攻击者最常用的，但我们也应该认识到，攻击者所能使用的方式不仅仅局限于上述三条。

2.5 白盒密码

由 Chow、Eisen、Johnson 和 van Oorschot 于 2002 年提出的白盒密码 [3] 可谓是一个崭新的研究领域。它的出现源自于持续增长的对设计和实现能够部署于不可信任的运行环境——即白盒环境——中的强密码算法的需求。在白盒环境中，攻击者不仅拥有密码算法的硬件/软件实现，还具有这些算法实现——特别是软件实现——及其运行环境的完全访问及控制的权限。而白盒密码则关注于保护在白盒环境中运行的密码算法的软件实现。

白盒密码是一种等价转换技术，它将传统密码算法转换成一种在功能上与之等价但具备白盒安全性的算法。经过转换的密码算法运行起来像是一个黑盒，即使它运行在白盒模型下。也就是说，即使攻击者具有对密码算法实现及其运行环境的完全掌控，攻击者仍然无法获得相比于黑盒环境下更多的攻击优势。

2.6 白盒安全性

白盒安全性指的是白盒实现在面对白盒环境中的攻击时的防护能力，更确切地说，是保护白盒实现中内置密钥不被提取的能力。

正如在设计和分析密码算法时，我们需要通过攻击模型来衡量算法的安全性。在白盒安全性的研究中，我们同样需要一些安全性衡量指标来帮助我们量化安全性水平。Chow 等人 [3] 提出了两种通用的指标：白盒多样性和白盒含混度，用以衡量基于查找表的白盒实现所能达到的白盒安全性水平。

白盒多样性，记为 $WB-div$ ，表示了某种特定的未经编码的查找表通过随机选取且相互独立的置换所能生成的所有经过编码的查找表的个数。对于密钥相关查找表而言，由于内置的密钥材料的改变也必须考虑在内。白盒含混度，记为 $WB-amb$ ，刻画了生成某种特定的经过编码的查找表的可能的构造方式的个数。

白盒多样性和白盒含混度其原本的目的是衡量单个查找表的白盒安全性，但他们实际上还能够衡量某个密钥给定的分组密码的基于查找表的白盒实现的安全性，只不过对于后者，其定义会有点变化：白盒多样性表示了所有在功能上等价的白盒实现的个数；而白盒含混度则统计了与某个确定的白盒实现相关的所有可能的内置密钥。有一点需要注意的是：由于查找表网络中各个节点的关系都需要纳入考量，因此计算整个白盒实现的白盒多样性和白盒含混度是非常复杂的工作。

2.7 本章总结

在这一章中，我们将阐述分组密码、白盒密码的相关概念和定义，并且介绍与二者相关的一些安全性方面的基本概念。

分组密码是一类采用对称密钥的确定性密码算法，它利用一套由密钥确定的加密解密变换对长度固定的分组进行密码操作。做为对称密钥算法的一种，它的用户（即通信各方）在使用算法进行加密或解密的过程中共享相同的密钥材料。

高级加密标准 AES 作为分组密码的代表，是美国国家标准技术研究所于 2001 年敲定的新的分组加密标准。其目的是替代已经不能满足需求的数据加密标准 DES。AES 源自于 Rijndael，而后者是由 Daemen 和 Rijmen 于 2000 年所

设计的“代换-置换网络”分组密码。

在密码算法的设计中，**Kerckhoffs** 假设指出，即使算法的每个细节都被公开，只要密钥没有泄露，密码算法就仍然应当保持其安全性，即：信息的安

全性于密钥。

在安全性方面，密码算法的安全性一般分三类：无条件安全性、计算安全性和可证明安全性。无条件安全性水平最高，但想要达到则必须使用理想分组密码。由于理想分组密码在其实现上是不现实的，因而实际的分组密码至多能够达到计算安全性。而计算安全性按照其定义在估计的时候是难以操作的，在实践中，我们一般考虑在全体现有攻击中选择一个良好定义的、具有代表性的子集，并检验子集中的方法在攻击特定分组密码时的复杂度，并最终检验的结果来判定该分组密码是否具有计算安全性。

而攻击模型是对攻击者攻击能力的归纳和总结，目的在于系统的阐述部署及运行密码算法的不安全的环境，并可以用于安全性水平评估。对于分组密码设计和分析而言常见的模型有黑盒模型、灰盒模型和白盒模型，其强度依次增加。

其中，我们最为关注的是白盒模型，在白盒模型中，攻击者对于密码算法的软件实现及其运行环境都具有完全的访问及控制权限，能够随意观察或修改软件实现在运行过程中的中间值或者在内存中搜索密钥。而白盒密码则关注于保护在白盒环境中运行的密码算法的软件实现。

白盒密码是一种等价转换技术，它将传统密码算法转换成一种在功能上与之等价但具备白盒安全性的算法。对于经过转换的密码算法，即使其运行在白盒环境中，攻击者仍然无法获得相比于黑盒环境下更多的攻击优势。

白盒安全性指的是白盒实现在面对白盒环境中的攻击时的防护能力，更确切地说，是保护白盒实现中内置密钥不被提取的能力。在衡量基于查找表的白盒实现所能达到的白盒安全性水平时，通常要用到两种通用指标：白盒多样性和白盒含混度。前者表示了某种特定的未经编码的查找表通过随机选取且相互独立的置换所能生成的所有经过编码的查找表的个数，而后者表示了生成某种特定的经过编码的查找表的可能的构造方式的个数。

第三章 基于查找表的白盒 AES 实现及其相应的攻击

根据白盒模型中的假设，密码算法中的每一步操作都有可能造成关键信息的泄露。因此，我们在设计白盒实现方案的时候会很自然地想到为每步操作加入不可被攻击者分离的随机性来引入含混性，从而使得攻击者不能从每步操作的输入输出中获取关键信息。Chow 等人^[3]在其白盒 AES 实现^[3]及白盒 DES 实现^[8]中为我们展示了一种实际的白盒实现的构造策略：先将密码算法轮函数中的各密码操作进行整合，构造出一系列实际的查找表，再向其中以一种攻击者无法剥离的方式注入随机双射作为操作的输入输出编码。通过将密码算法中的所有操作转换成一系列首尾相接的查找表，我们就能够构造基于查找表的白盒实现方案。本章中，我们将介绍两种基于查找表的白盒实现方案，它们分别是 Chow 等人提出的白盒 AES 实现和 Xiao-Lai 白盒 AES 实现。与此同时，我们还将介绍对这两种方案的攻击：Billet 等人对 Chow 等人的白盒 AES 实现的攻击，即 BGE 攻击；De Mulder 等人对 Xiao-Lai 白盒 AES 实现的攻击。

3.1 Chow 等人的白盒 AES 实现

Chow 等人的白盒 AES 实现方案分为两个主要的步骤：

1. 对标准 AES 算法的轮函数进行拆分和重组，然后转化为一系列未经编码的查找表
2. 向未经编码的查找表中加入随机可逆映射对其输入输出进行编码

为了能够使加密操作和随机置换更好的整合，Chow 等人使用了与标准 AES 算法等价但又不完全相同的算法。他们重新定义了 AES 算法中轮函数的边界，并且更改了加密操作的顺序，将前后相接的两轮交界处的相邻的两步操作，即 AddRoundKey 和 SubBytes，调整到同一轮中。Chow 等人所使用的等价算法的加密过程如图 3-1 所示，其中 \hat{k}^r ($1 \leq r \leq 10$) 表示的是经过行位移操作

的第 r 轮的轮密钥, 可以表示为:

$$\hat{k}^r = \text{ShiftRows}(k^r)$$

接下来, 每步 **AddRoundKey** 操作与后面相接的 **SubBytes** 结合, 生成一个 8-bit 双射的查找表 *T-box*。在每一轮中, 128-bit 的轮密钥将会最终产生 16 个 **T-box**。如果我们用 k_i^r 来表示第 r 轮 ($1 \leq r \leq 10$) 的轮密钥 k^r 的第 $i+1$ 个字节 ($0 \leq i \leq 15$), 那么 **T-box** 可以通过下面的公式来表示:

$$T_i^r(x) = \begin{cases} S(x \oplus \hat{k}_i^r) & 1 \leq r \leq 9 \\ S(x \oplus \hat{k}_i^r) \oplus k_i^{r+1} & r = 10 \end{cases}$$

其中 $0 \leq i \leq 15$ 。

紧接着, 表示 **MixColumns** 操作的 4×4 矩阵 **MC** (以字节为元素) 被分割成四个列向量, 分别以 $y_0 \ y_1 \ y_2 \ y_3$ 表示。而每个向量都被用于构造查找表 Ty_i , 其构造方式表示如下:

$$\begin{aligned} Ty_0(x_0) &= y_0 x_0 = ['02' \ '01' \ '01' \ '03']^\top x_0, \\ Ty_1(x_1) &= y_1 x_1 = ['03' \ '02' \ '01' \ '01']^\top x_1, \\ Ty_2(x_2) &= y_2 x_2 = ['01' \ '03' \ '02' \ '01']^\top x_2, \\ Ty_3(x_3) &= y_3 x_3 = ['01' \ '01' \ '03' \ '02']^\top x_3. \end{aligned}$$

其中 $x_0 \ x_1 \ x_2 \ x_3$ 表示 **MixColumns** 操作的四个输入字节。而利用查找表 Ty_i , **MixColumns** 操作可以进一步的被表示如下:

$$\begin{aligned} \text{MC} \cdot [x_0|x_1|x_2|x_3]^\top &= Ty_0(x_0) \oplus Ty_1(x_1) \\ &\oplus Ty_2(x_2) \oplus Ty_3(x_3) \end{aligned}$$

然后, 每一个 **T-box**: T_i^r , 会与相应的查找表 $Ty_{(i \bmod 4)}$ 结合, 生成一个 8 to 32-bit 的映射:

$$Ty_i \circ T_i^r(x) = Ty_i(T_i^r(x))$$

其中 $0 \leq i \leq 3$, $0 \leq r \leq 10$ 。将查找表 $Ty_i \circ T_i^r(x)$ 与两个线性可逆映射——作为输入混淆双射的 8×8 -bit 映射 mb_i^r 和作为输出双射的 32×32 -bit 映射 $\text{MB}_{(i \bmod 4)}^r$

Require: $plaintext, k^r$ ($1 \leq r \leq 11$)
Ensure: $ciphertext$

```

state  $\leftarrow$  plaintext
SHIFTROUNDKEY( $k^1, \dots, k^{10}$ )
for  $r \leftarrow 1, 9$  do
    state  $\leftarrow$  ShiftRows(state)
    state  $\leftarrow$  AddRoundKey(state,  $\hat{k}^r$ )
    state  $\leftarrow$  SubBytes(state)
    state  $\leftarrow$  MixColumns(state)
end for
state  $\leftarrow$  ShiftRows(state)
state  $\leftarrow$  AddRoundKey(state,  $\hat{k}^{10}$ )
state  $\leftarrow$  SubBytes(state)
state  $\leftarrow$  AddRoundKey(state,  $k^{11}$ )
ciphertext  $\leftarrow$  state
return ciphertext

```

```

procedure SHIFTROUNDKEY( $k^1, \dots, k^{10}$ )
     $r \leftarrow 1$ 
    while  $r \leq 10$  do
         $\hat{k}^r \leftarrow$  ShiftRows( $k^r$ )
         $r \leftarrow r + 1$ 
    end while
end procedure

```

图 3-1 一个等价的 AES 加密过程.

Fig 3-1 An equivalent AES encryption process.

——进行结合，查找表 **TypeII** 就构造完成了，其构造方式表示如下：

$$\text{TypeII}_i^r = \text{mathhtt{MB}}_{(i \bmod 4)}^r \circ T_{y(i \bmod 4)} \circ T_i^r \circ \text{mb}_i^r$$

为了抵消嵌入在查找表 **TypeII** 中的混淆双射 **mb** 和 **MB** 对数据流的影响，**Chow** 等人设计了查找表 **TypeIII**，该查找表是从 8-bit 到 32-bit 的映射，其构造过程使用了与 **TypeII** 类似的方法：

$$\text{TypeIII}_i^r = \text{diag}(\text{mb}_0^{-1r+1}, \dots, \text{mb}_3^{-1r+1}) \circ \text{MB}_{(i \bmod 4)}^{-1r}$$

此外，8 to 128-bit 的查找表 **TypeI** 被构造用来控制外部编码。

在完成整个白盒实现的构造之前，最后一个步骤是为每种查找表添加非线性的可逆映射以作为这些查找表的内部输入输出编码。由于不同的查找表的输入输出长度不同，**Chow** 等人通过将若干 4-bit 非线性双射并联的方式构造符合不同长度要求的输入输出编码。由于非线性编码的加入，每个查找表的输出都经过了非线性变换，异或操作不能直接进行，**Chow** 等人因此构造了查找表 **TypeIV** 用来完成对原数据的异或操作：**TypeIV** 以两个经过编码的半字节未输入，先将其解码，然后将两个半字节异或为一个，最后对结果重新编码。为了方便起见，我们定义：

$$\begin{aligned} P_{i,j}^r &= \text{mb}_i^r \circ (\text{enc} \parallel \text{enc}) \\ Q_{i,j}^r &= (\text{dec} \parallel \text{dec}) \circ \text{mb}_i^{-1r+1} \end{aligned}$$

其中 **enc** 表示 4-bit 非线性双射编码，而 **dec** 表示 4-bit 非线性双射解码。

3.2 BGE 攻击

2004 年，在 **Chow** 等人的白盒 AES 实现发表两年之后，**Billet**、**Gilbert** 和 **Ech-Chatbi** 提出了一种针对该白盒实现的有效代数攻击 [13]。该攻击随后被以其三位作者的首字母命名为 **BGE** 攻击。

由于查找表在设计时加入了许多混淆技术以获得较高的白盒安全性水平，如果攻击者仅仅局限于对单个查找表进行密钥提取，那么查找表所提供的白盒多样性及白盒含混度会使他的攻击变得十分困难。**Billet** 等人设计了一种更有效的方法来获取查找表中所蕴含的密钥信息。它利用了相邻查找表中相互对应

的随机编码能够相互抵消的特点, 将同一轮所涉及的若干查找表结合在一起进行分析。

首先, BGE 攻击筛选出那些输入位于状态矩阵同一列的查找表 TypeII_i^r 。然后将它们与对应的查找表 TypeIII 结合。这样一来, 由于两类查找表所包含的正变换与逆变换相互抵消, 混淆双射 MB 的实际上被移除, 仅留下非线性置换 $P_{i,j}^r$ 和 $Q_{i,j}^r$ ($2 \leq r \leq 9$, $0 \leq i, j \leq 3$) 来分别编码 $T_{i,j}^r$ 的输入以及操作 MixColumn 的输出。

接下来, 非线性的输出置换 $Q_{i,j}^r$ 将被转化为仿射映射。由于相邻查找表相对应的输入输出编码——前表的输出编码和所对应的后表的输入编码——是互逆变换, 因此能够相互抵消。因此一旦将输出置换 $Q_{i,j}^r$ 转化为仿射映射其对应的输入置换 $P_{i,j}^{r+1}$ 也同时被转化为仿射映射。

在将所有的非线性输入输出置换转化为仿射映射之后, 攻击者将确定输出置换 $Q_{i,j}^r$ 和密钥相关的仿射映射

$$\tilde{P}_{i,j}^r(x) = P_{i,j}^r(x) \oplus k_{i,j}^r, \quad 0 \leq i, j \leq 3$$

在攻击者已经得到了 $P_{i,j}^r$ 的情况下, 他能够轻易地提取轮密钥字节

$$k_{i,j}^r = \tilde{P}_{i,j}^r(0) \oplus P_{i,j}^r(0), \quad 0 \leq i, j \leq 3$$

当攻击者对连续三轮的查找表进行攻击, 并提取得到轮密钥字节 $k_{i,j}^r$ 和 $k_{i,j}^{r+1}$ 之后, 最终, 攻击者能够通过观察白盒实现运行时的数据流确定轮密钥 k , 并且还能够利用公开的 AES 密钥编排算法来反向推导主密钥。这样一来, Chow 等人所提出的白盒 AES 实现就被攻破了。

3.3 Xiao-Lai 白盒 AES 实现

2009 年, Xiao 和 Lai 提出了一种新型白盒 AES 实现方案 [17], 并声称他们的方案能够抵御 BGE 攻击 [13]。Xiao 和 Lai 在设计他们的白盒 AES 实现方案时依然沿用了 Chow 等人提出的基于查找表的方法, 并且使用了与 Chow 等人相同的等价 AES 加密解密过程。但与 Chow 等人不同的是, Xiao 和 Lai 对查找表的构造方式做出了如下的改进:

1. 全体随机映射和随机编码都是 \mathbb{F}_2 -线性的。因此, 所有的异或操作 (XOR) 都可以直接作用于已经过编码的数据, 也就意味着, Chow 等人的方案中的由一系列查找表 XOR 构成的树形结构被省略了。

2. 随机映射和随机编码同时作用于至少两个字节的数据而不是仅仅作用于四个 bit (Chow 等人的方案中的非线性编码) 或一个字节 (Chow 方案中的线性双向映射)
3. ShiftRows 操作以矩阵-向量乘法的形式显式的实现出来而不是隐式的将其融进数据流中。

由于 Xiao-Lai 白盒 AES 实现沿用了在 Chow 等人的实现中所使用的等价 AES 加密解密过程, 因此对轮函数的拆分和重组是相同的。为了方便起见, 我们将 T-box 的构造方式重写如下:

$$T_i^r(x) = \begin{cases} S(x \oplus \hat{k}_i^r) & 1 \leq r \leq 9 \\ S(x \oplus \hat{k}_i^r) \oplus k_i^{r+1} & r = 10 \end{cases}$$

其中 $0 \leq i \leq 15$ 。

与 Chow 等人的方案不同的是, Xiao 和 Lai 将 4×4 矩阵 MC 分割成为两个 4×2 子矩阵 MC₀ 和 MC₁, 而不是四个列向量, 其中 MC₀ 表示矩阵 MC 的四列中的前两列而 MC₁ 则表示剩余两列。利用子矩阵 MC₀ 和 MC₁, 矩阵 MixColumns 能够被表示为:

$$\begin{bmatrix} state_{4i} \\ state_{4i+1} \\ state_{4i+2} \\ state_{4i+3} \end{bmatrix} = MC_0 \cdot \begin{bmatrix} state_{4i} \\ state_{4i+1} \end{bmatrix} \oplus MC_1 \cdot \begin{bmatrix} state_{4i+2} \\ state_{4i+3} \end{bmatrix}$$

其中 $0 \leq i \leq 3$ 。

接下来, 第 r 轮 ($1 \leq r \leq 9$) 的两个 T-box: T_{2i}^r 和 T_{2i+1}^r ($0 \leq i \leq 7$) 将与矩阵 MC_(i mod 2) 结合, 生成一个 16 to 32-bit 的映射。然后每个这样的映射会与两个随机生成的线性双射编码—— L_i^r 和 $R_{[i/2]}^r$ ——结合, 以增加混淆性, 其中 L_i^r 是查找表的输入编码而 $R_{[i/2]}^r$ 则是输出编码。由于第十轮不包含 MixColumns 操作, 输入编码 L_i^{10} 和输出编码 $R_{[i/2]}^{10}$ 将直接与 T-box—— T_{2i}^{10} 及 T_{2i+1}^{10} ——相连。每一轮会包含八个这样的映射, 它们都以查找表的形式被实现出来。如果将这些查找表记作 TMC_i^r, 则它们可以被表示为:

$$TMC_i^r = R_{[i/2]}^r \circ MC_{(i \bmod 2)} \circ (S \parallel S) \circ \oplus_{(\hat{k}_{2i}^r \parallel \hat{k}_{2i+1}^r)} \circ L_i^r$$

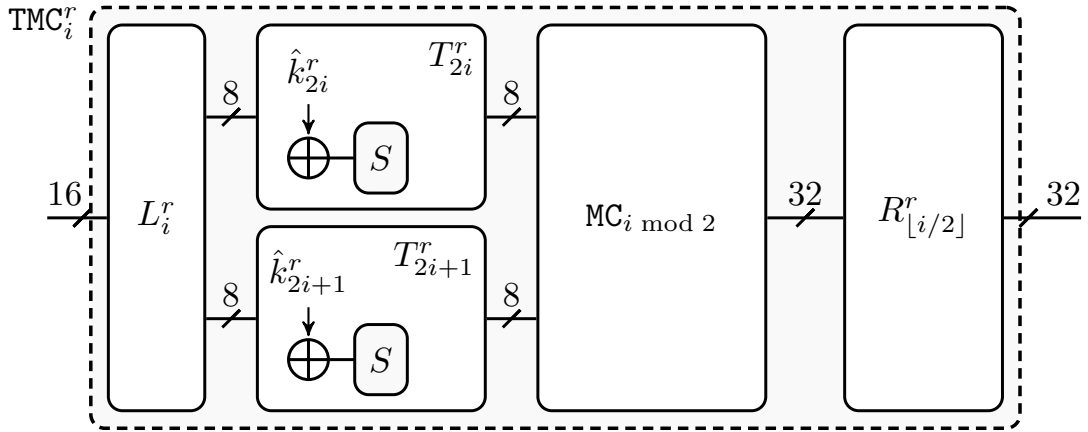


图 3-2 查找表 TMC 的结构.

Fig 3-2 The structure of TMC.

其中 $0 \leq i \leq 7$, $(S||S)$ 表示通过并联两个 AES 的 S-box 得到的 16-bit 双射的 S-box, 而 \oplus_c 表示函数

$$\oplus_c(x) = x \oplus c$$

。查找表 TMC_i^r 的结构如图 3-2 所示。

接下来, Xiao 和 Lai 定义了两个 128-bit 线性可逆随机映射 IN 和 OUT, 并将它们分别作用于明文和密文, 作为外部编码。因其线性性, 这两个映射能够被表示为在有限域 $\mathbb{GF}(2)$ 上的 128-bit 非奇异方阵。

最后, 对于第 r 轮 ($1 \leq r \leq 10$), 一个在有限域 $\mathbb{GF}(2)$ 上的 128-bit 非奇异方阵 M^r 被生成出来用于代替 ShiftRows 操作, 并同时控制外部/内部编码。我们观察到 ShiftRows 操作同样能够用一个在有限域 $\mathbb{GF}(2)$ 上的 128-bit 非奇异方阵 SR 来表示。因此, M^r 操作可以被表示如下:

$$M^r = \begin{cases} (\text{LL}^1)^{-1} \circ \text{SR} \circ \text{IN}^{-1} & r = 1 \\ (\text{LL}^r)^{-1} \circ \text{SR} \circ (\text{RR}^{r-1})^{-1} & 2 \leq r \leq 10 \\ \text{OUT} \circ (\text{RR}^{10})^{-1} & r = 11 \end{cases} \quad (3-1)$$

其中

$$\text{LL}^r = \text{diag}(L_0^r, \dots, L_7^r)$$

$$\text{RR}^r = \text{diag}(R_0^r, \dots, R_3^r)$$

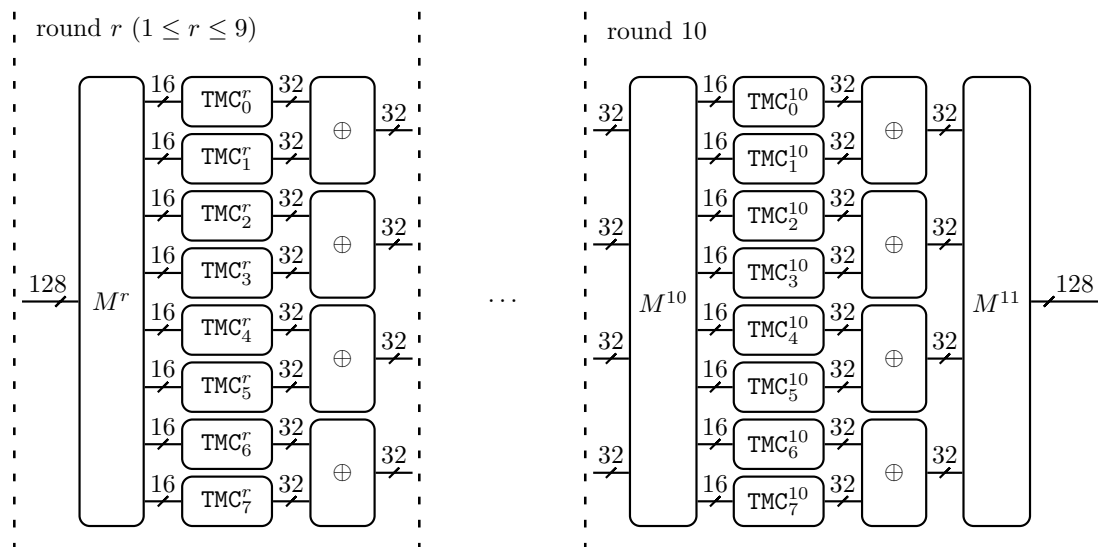


图 3-3 Xiao-Lai 白盒 AES 实现的结构.

Fig 3-3 The structure of Xiao-Lai white-box implementation.

M^r 操作都被实现为在有限域 $\mathbb{GF}(2)$ 上的矩阵乘法的形式，并且在运行时动态计算。

按照上述的步骤，Xiao 和 Lai 的白盒 AES 实现方案就已经构造完毕了。在运行过程中，与 M^r 进行矩阵-向量乘法会和搜索查找表 TMC_i^r 交替进行，其流程结构如图3-3所示。

3.4 De Mulder 等人的攻击

2012 年，De Mulder、Roelse 和 Preneel 针对 Xiao-Lai 白盒 AES 实现提出了一种有效攻击。该攻击的核心在于对线性等价算法 (linear equivalence algorithm) 的应用，而该算法是由 Biryukov 等人于 2003 年提出的。

定义 3.1. 两个 n -bit 置换 (或 S-box) —— S_1 和 S_2 —— 被称为线性等价当存在一对 n -bit 的线性映射 (A, B) 使下式成立：

$$S_2 = B \circ S_1 \circ A$$

而使得上式成立的线性映射对 (A, B) 被称为 S_1 和 S_2 之间的一种线性等价关系。

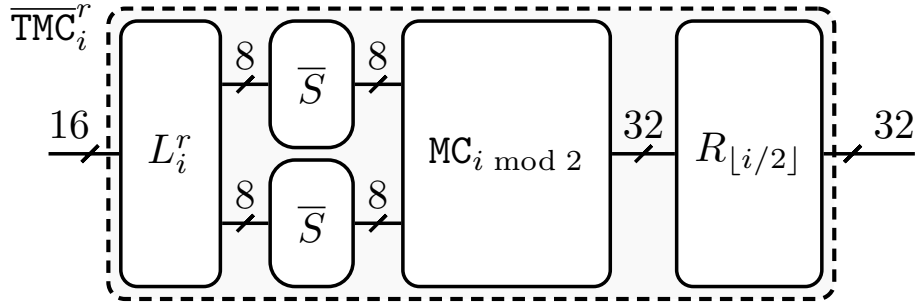


图 3-4 查找表 TMC 的结构.

Fig 3-4 The structure of TMC.

线性等价算法提出的目的是解决两个映射间线性等价关系的判定问题，即给定两个 n -bit 双射置换 S_1 和 S_2 ，确定二者是否线性等价。如果二者之间确实存在线性等价关系，那么算法输出一个由二者之间全体线性等价关系构成的集合，否则，输出信息声明线性等价关系不存在。

给定一个对映射 A 的初始猜测，我们能够推算出一部分关于映射 B 的信息。而利用映射 B 的泄露信息，我们能够反过来揭示一部分映射 A 中的信息。这个过程就被称为“猜测的指数放大效应”。根据线性特性，给定 k 个线性无关的向量，攻击者能够确定 2^k 个由已知向量生成的向量。由此，攻击者能够有效地计算两个映射之间的线性相关关系。

De Mulder 等人在其攻击中采用了线性等价算法的一个变体，利用 Xiao-Lai 白盒 AES 实现的结构，将攻击复杂度降至 2^{32} 。他们先把密钥相关的查找表 TMC_i^1 （如图 3-2 所示）转化成为密钥无关的查找表 $\overline{\text{TMC}}_i^1$ （如图 3-4 所示），再构造一个由第一轮中某个特定的查找表 $\overline{\text{TMC}}_i^1$ ($t \in \{0, \dots, 7\}$) 及其后续的矩阵 M^2 所组成的特殊轮。将矩阵 M^2 的相应的输出字节置零，攻击者就能够确定由 L_t^1 所编码的一对字节之间的双射，从而获取线性等价关系的解集。

3.5 本章总结

本章介绍了两种基于查找表的白盒实现方案：Chow 等人提出的白盒 AES 实现和 Xiao-Lai 白盒 AES 实现。与此同时，我们还介绍了针对上述两方案的攻击：Billet 等人对 Chow 等人的白盒 AES 实现的攻击，即 BGE 攻击以及 De Mulder 等人对 Xiao-Lai 白盒 AES 实现的攻击。

由于白盒环境中的攻击者具有对算法实现以其运行环境都能够完全掌控, 白盒实现的设计需要不同于以往的构造策略。**Chow** 等人的白盒实现首次展示了一种基于查找表的构造策略。它先将密码算法轮函数中的各密码操作进行调整然后整合, 构造出一系列实际的查找表, 再向其中以一种攻击者无法剥离的方式加入随机可逆映射对其输入输出进行编码。具体来说, 他们先将与标准 AES 算法等价但是更容易用于白盒实现的等价算法, 利用加密操作中的线性特性, 构造一系列实际的查找表, 并以随机可逆映射对其进行输入输出编码, 而事前选定的密钥就内嵌在这些查找表中。通过用查表操作替代传统的密码操作, 一部分中间过程就被隐藏了, 而通过随机编码注入的混淆性让攻击者难以通过观察单个查找表的输入输出推导关键信息。

Chow 等人对查找表的混淆方式使得暴力破解对其是难以奏效的。但是, **Billet** 等人提出了一种高效的代数攻击, 它利用了相邻查找表中相互对应的随机编码能够相互抵消的特点, 将同一轮所涉及的若干查找表结合在一起进行分析, 可以轻易地从 **Chow** 等人的白盒实现中提取密钥信息。**BGE** 攻击的核心是将大于 8-bit 的随机映射剥离掉, 而对于 8-bit 的映射, 利用算法将其非线性部分移除, 从而转换为仿射映射, 并最终从包含密钥的仿射映射中提取密钥字节。

为了抵御 **BGE** 攻击, **Xiao** 和 **Lai** 提出了新的白盒 AES 实现方案。该方案在设计时依然沿用了 **Chow** 等人提出的基于查找表的方法。但与 **Chow** 等人不同的是, **Xiao** 和 **Lai** 对查找表的构造方式做出了三项改进: 首先, 将非线性的随机编码改为 \mathbb{F}_2 -线性的, 这样做简化了结构, 省略了用于异或操作的查找表; 其次, 令随机映射同时作用于至少两个字节的数据而不是仅仅作用于四个 bit (**Chow** 等人的方案中的非线性编码) 或一个字节 (**Chow** 方案中的线性双向映射), 结果是获得了更大的查找表, 提升了白盒安全性; 最后, 将 **ShiftRows** 操作以矩阵-向量乘法的形式显式的实现出来而不是隐式的将其融进数据流中。

下一章, 我们将提出一种全新的基于更大查找表的白盒 AES 实现, 该实现需要在保持可用性的基础上尽可能提高白盒安全性, 特别是面对 **BGE** 攻击和 **De Mulder** 等人的攻击的防护能力。

第四章 一种新的基于 16 to 32-bit 查找表的非线性混淆方案

在本章中，我们将描述一种全新的白盒 AES 实现。我们的设计采用了与 Chow 等人的白盒实现类似的基本策略：将 AES 轮函数中不同的操作整合在一起，并转换成为一系列查找表。这些生成的查找表将会进一步与输入输出双射进行混合，再通过非线性的输入输出编码，最终生成攻击者无法进行密钥提取的查找表。

而新方案新颖之处在于采用了更大的密钥相关查找表：16 to 32-bit 的查找表 $nTMC$ ，以提供更高的白盒安全性。在分析了 Xiao-Lai 白盒实现被攻破的原因之后，我们采用了非线性输入输出编码，并且设计了立体的查找表关联方式，以避免新方案被 De Mulder 等人所构造的对密钥相关查找表的单独分析。

根据 AES 算法的等价算法 (如图3-1所示)，我们能够将我们的设计方案实现为一个具有十轮迭代的过程，其中，每轮迭代均包含三个连续的步骤：

1. 步骤 1 对应 ShiftRows 操作；
2. 步骤 2 实际上是一个提供辅助功能的步骤，它完成在有限域 $GF(2)$ 上的加法，即异或运算；
3. 步骤 3 则涵盖了传统 AES 算法的轮函数中除 ShiftRows 外的其他三项操作：AddRoundKey, SubBytes 和 MixColumns。

上述的每一个步骤都被实现为一个由同一类型查找表组成的集合。因此，加密过程的流程结构可以表示如图4-1所示。

下面我们分别阐述三种类型的查找表，他们分别对应上述的三个步骤。

4.1 步骤 3 中使用的查找表 $nTMC$

为了更清晰而准确地理解整个加密过程的实现，我们不妨以步骤 3 使用的查找表 $nTMC_i^r$ (其中, $1 \leq r \leq 11$; $0 \leq i \leq 7$) 作为切入点。要构造 $nTMC_i^r$ ，我

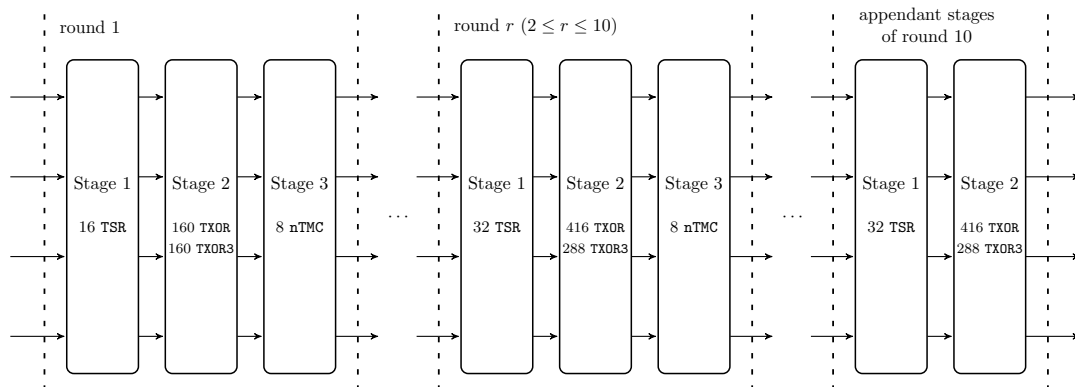


图 4-1 加密过程的流程结构.

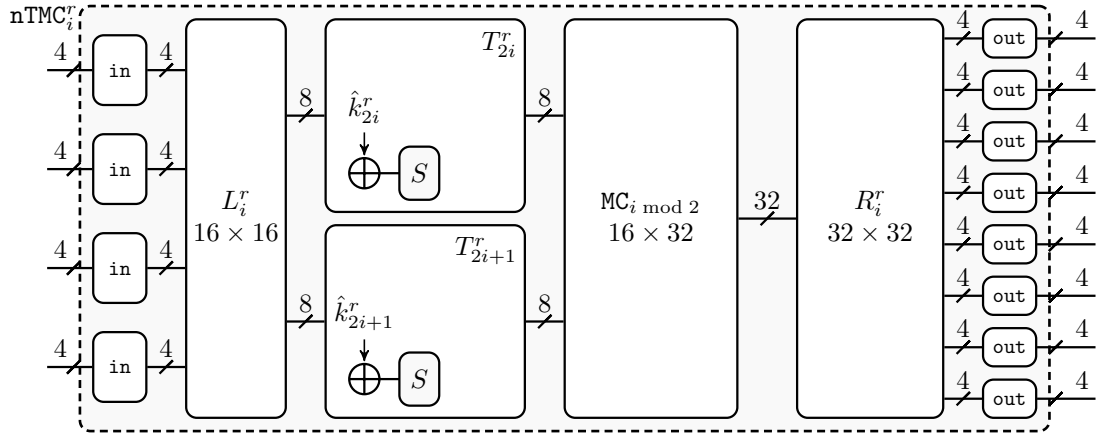
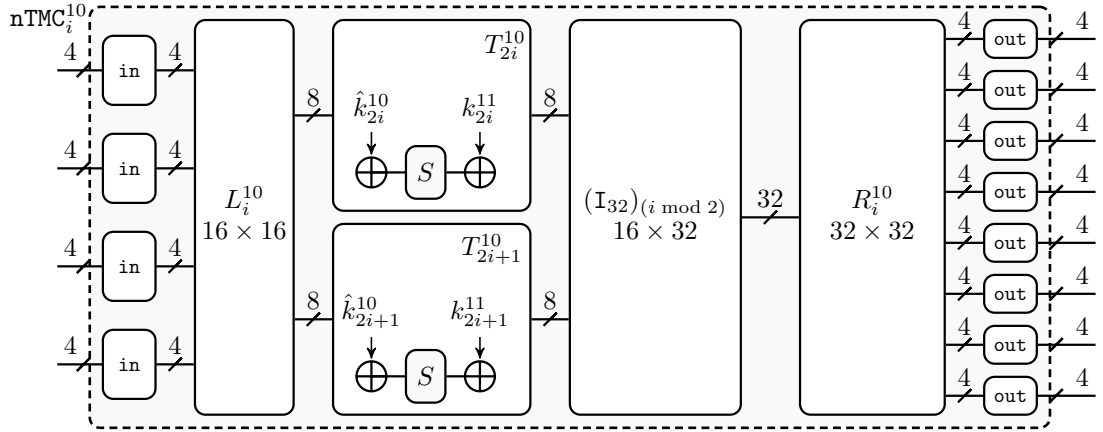
Fig 4-1 The structure of encryption process.

们必须首先将 **AddRoundKey** 与相应的 **SubBytes** 结合以生成 T-box: T_{2i}^r 和 T_{2i+1}^r 。之后, 我们把生成的两个 T-box 同对应的矩阵 $MC_{i \bmod 2}$ ——**MixColumns** 矩阵 **MC** 的一个 4×2 的子矩阵——进行关联。而互不相同的线性映射 L_i^r 和 R_i^r 被作为输入输出双射引入以混淆输入输出。最后, 4-bit 的非线性输入输出编码被加入进来, 这些小的非线性的置换为我们的查找表提供了非线性的混淆特性从而使已有的针对线性混淆的攻击失去作用, 因而非常重要, 表名中的“n”实际上正是非线性 (non-linear) 的含义。在每一轮中, 一共有八个 $nTMC_i^r$ 按照上述的方法生成, 其结构如图4-2所示。

我们注意到, 每个 $nTMC_i^r$ 的 32-bit 输出都与 **MixColumns** 矩阵 **MC** 的一半——某一个子矩阵 MC_0 或者 MC_1 ——相关。而两个对应的 $nTMC_{2i}^r$ 和 $nTMC_{2i+1}^r$ 的输出分别与两个互补的子矩阵相关。因此, 将两个对应表的输出结合起来, 通过合适的手段就能够还原出矩阵 **MC** 的作用。但由于我们为了提升安全性而引入的非线性置换, 不同的 $nTMC_i^r$ 的输出是无法简单相加或者异或的。为了结合步骤 3 中不同查找表的输出, 并进一步完成 **ShiftRows** 操作, 我们设计了步骤 1 以及相应的 $TSR_{i,j}^{r+1}$ 来对输出做适当的变换。

4.2 步骤 1 中使用的查找表 TSR

我们知道, 步骤 1 的关键在于完成 **ShiftRows** 操作。除此之外, 相应的输入输出编码也必须被整合进步骤 1 的查找表中, 用以抵消双向混淆映射, 使之与相邻步骤契合。因此, 我们将 **ShiftRows** 操作与相应的线性输入输出编码

(a) 第 r 轮 ($1 \leq r \leq 9$)

(b) 第 10 轮

图 4-2 表 nTMC 的结构.

Fig 4-2 The structure of nTMC.

$(L_i^r)^{-1}$ 及 $(R_i^{r-1})^{-1}$ 结合在一起, 定义 M^r 操作。易知, M^r 操作本质上是一个从 128-bit 输入到 128-bit 输出的双向线性映射变换, 因而可以表示成一个 128-bit 非奇异方阵 M^r (定义为式3-1)。我们当然会首先想到将这个映射关系转化为查找表, 但是一个输入为 128-bit 的查找表是无论如何难以实现的, 所以我们利用了一系列较小的并且能够实际转化为查找表的映射来替代一个巨大的映射。为了达到这个目的, 我们首先需要将矩阵 M^r 切分成为一系列细长的像长条一般的子矩阵。

对于任意列向量 $\mathbf{x} \in \mathbb{GF}(2)^{128}$, \mathbf{x}^\top (\mathbf{x} 的转置向量) 能够被表示成 $(\mathbf{x}_0^\top, \dots, \mathbf{x}_{31}^\top)$, 其中 \mathbf{x}_i 表示 \mathbf{x} 在 $\mathbb{GF}(2)^4$ 上的一个子向量。令 \mathbf{x} 作为变换 M^r 的输入, 则输出可以表示如下:

$$M^r(\mathbf{x}) = M^r \cdot \mathbf{x} = \sum_{j=0}^{31} (M^r)_j \cdot \mathbf{x}_j$$

其中 $(M^r)_j$ ($0 \leq j \leq 31$) 表示了由矩阵 M^r 的第 $4j+1$ 列至第 $4j+4$ 列 (共四列) 构成的 128×4 -bit 子矩阵。

如果我们更进一步, 令 $s = \lfloor j/8 \rfloor$ 及 $t = j \bmod 8$ 并定义

$$(\mathbf{LL}^r)^{-1} = \text{diag}((L_0^r)^{-1}, \dots, (L_7^r)^{-1})$$

其中 $1 \leq r \leq 10$, 那么通过式3-1我们能将 $(M^r)_j$ 推导如下:

$$M_j^r = \begin{cases} (\mathbf{LL}^1)^{-1} \circ \mathbf{SR} \circ \mathbf{IN}_j^{-1} & r = 1 \\ (\mathbf{LL}^r)^{-1} \circ \mathbf{SR}_s \circ (R_s^{r-1})_t^{-1} & 2 \leq r \leq 10 \\ \mathbf{OUT}_s \circ (R_s^{10})_t^{-1} & r = 11 \end{cases} \quad (4-1)$$

其中 \mathbf{SR}_j ($0 \leq j \leq 3$) 表示了由矩阵 \mathbf{SR} 的第 $32j+1$ 列至第 $32j+32$ 列 (共 32 列) 构成的 128×32 -bit 子矩阵, 并且矩阵 \mathbf{SR} 能够表示为

$$\mathbf{SR} = [\mathbf{SR}_0 | \mathbf{SR}_1 | \mathbf{SR}_2 | \mathbf{SR}_3]$$

类似的, $(R_i^r)_j^{-1}$ ($0 \leq i \leq 3; 0 \leq j \leq 7$) 表示了由矩阵 $(R_i^r)^{-1}$ 的第 $4j+1$ 列至第 $4j+4$ 列 (共四列) 构成的 32×4 -bit 子矩阵, 并有

$$(R_i^r)^{-1} = [(R_i^r)_0^{-1} | \dots | (R_i^r)_7^{-1}]$$

此外, \mathbf{IN}_j^{-1} ($0 \leq j \leq 15$) 表示了矩阵 \mathbf{IN}^{-1} 的 128×8 -bit 子矩阵“条带”, 而类似的, 矩阵 \mathbf{OUT}_j ($0 \leq j \leq 3$) 表示了矩阵 \mathbf{OUT} 的 128×32 -bit 子矩阵。

我们接下来利用式4-1中 M_j^r 构造 $\text{TSR}_{i,j}^r$ (当 $r = 1$ 时, 有 $0 \leq i \leq 3$ 和 $0 \leq j \leq 3$; 当 $2 \leq r \leq 11$ 时, 有 $0 \leq i \leq 3$ 和 $0 \leq j \leq 7$)。根据上文所述, 对于 nTMC_i^{r-1} 的输出的加和操作会嵌入在 $\text{TSR}_{i,j}^r$ 中, 这意味着每一个 8 to 128-bit 的 $\text{TSR}_{i,j}^r$ 将会包含两个 4 to 128-bit 的映射。这两个映射将以适当的方式结合起来就像我们对它们做加法运算一样。在第 2 轮至第 11 轮的每一轮中, 都将会有 32 个 $\text{TSR}_{i,j}^r$ 被生成出来; 在第 1 轮中, 则只有 16 个查找表。每一个 $\text{TSR}_{i,j}^r$ 的结构如图4-3所示。

我们注意到, 随机矩阵 $(R_{2i}^r)_j^{-1}$ 和 $(R_{2i+1}^r)_j^{-1}$ 中的列向量在选取的过程中需要确保相互独立。在这个前提下, 这两个矩阵及其对应的异或操作的组合实际上等价于一个列满秩的 $32 \times 8\text{-bit}$ 矩阵, 而它的输出是在有限域 $\text{GF}(2)^{32}$ 的某个八维子空间中的 32-bit 向量。

当 $1 \leq r \leq 10$ 以及 $0 \leq i \leq 3$ 时, 每两个 nTMC_{2i}^r 和 nTMC_{2i+1}^r 将会与八个相应的 $\text{TSR}_{i,j}^{r+1}$ ($0 \leq j \leq 7$) 相关联。每个 $\text{TSR}_{i,j}^{r+1}$ 将从分别从 nTMC_{2i}^r 和 nTMC_{2i+1}^r 的输出中各取 4-bit (从第 $4j$ 位到第 $4j + 3$ 位) 作为其 8-bit 输入。 nTMC_i^r 与 $\text{TSR}_{i,j}^r$ 的连接方式如图4-4所示。

4.3 步骤 2 中使用的查找表 TXOR 和 TXOR3

在每一轮中, 32 个步骤 1 的 128-bit 输出将会在步骤 2 中被合并成为一个 128-bit 的值。为此, 我们定义了两种查找表来执行异或操作: 其中的一种表 TXOR 将 8-bit 输入映射成为 4-bit 输出而另一种表 TXOR3 将 12-bit 输入映射成为 4-bit 输出。这两种映射分别以二或三个半字节作为输入, 先对其进行解码将真正需要异或的数据恢复出来, 再进行异或运算生成一个半字节, 最后进行非线性编码。图4-5阐述了 TXOR 和 TXOR3 的结构。

尽管 TXOR3 相比于 TXOR 体积更大, 但是它能够提供更高的安全性: TXOR3 的白盒多样性为 2^{177} , 相比 TXOR 的 $2^{132.8}$, 提高了大约 2^{44} 。具体的分析我们会在下一节展开。更重要的是, 引入 TXOR3 能够降低由一系列异或操作 (即查找表 TXOR) 所构成的树形结构的层级, 这实际上意味着对查找表的总体访问时间会有所下降。具体来说, 如果仅仅使用 TXOR, 那么为了将 32 个半字节合并起来, 我们必须使用 $16 + 8 + 4 + 2 + 1 = 31$ 个 TXOR, 这将产生一个五层的树形结构; 而如果我们引入 TXOR3, 经过适当的设计, 树形结构能够被压缩至四层, 并且所用到的查找表的数量会更少: 在顶层, 我们可以使用八个 TXOR3 以及四

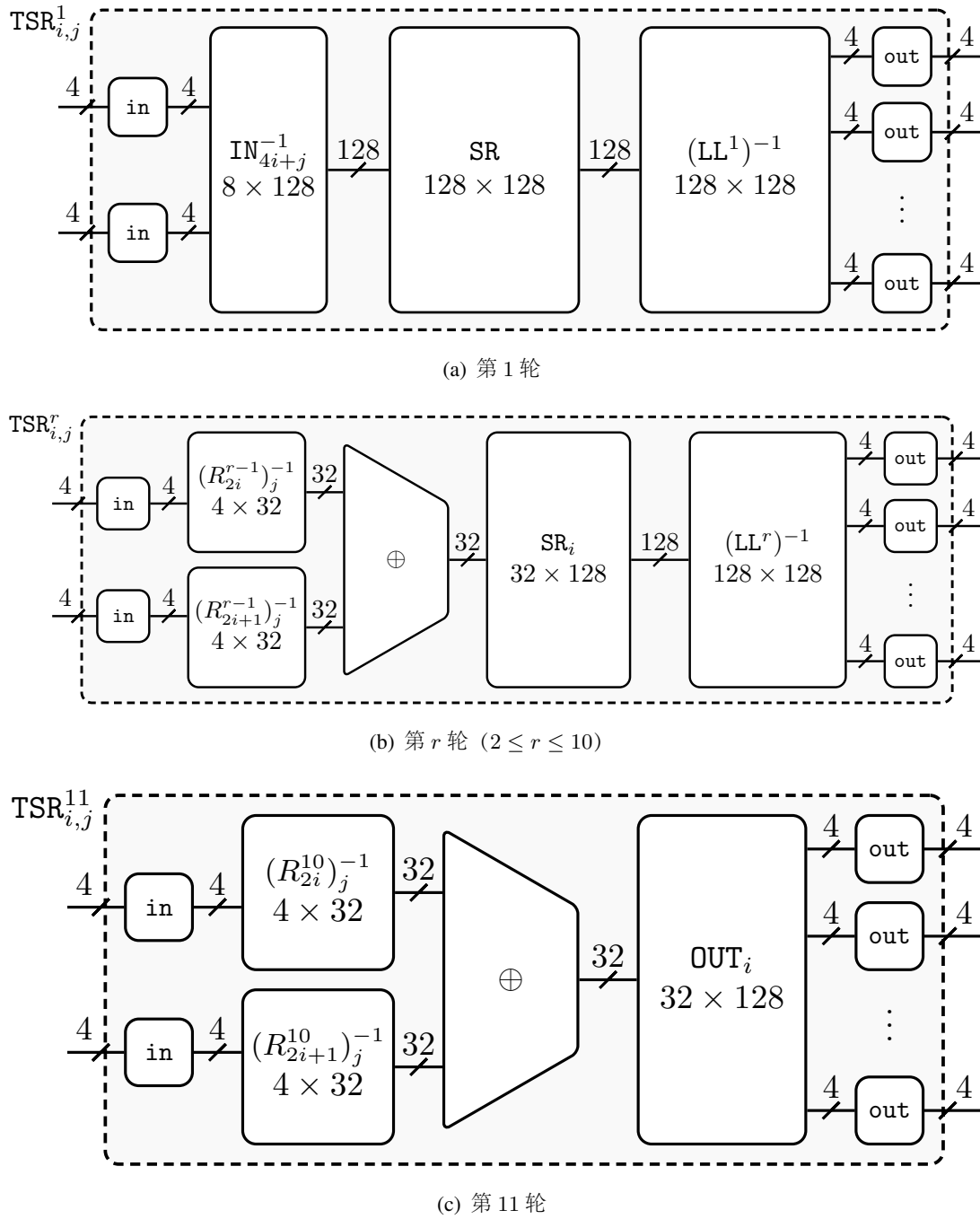


图 4-3 表 TSR 的结构.

Fig 4-3 The structure of nTMC.

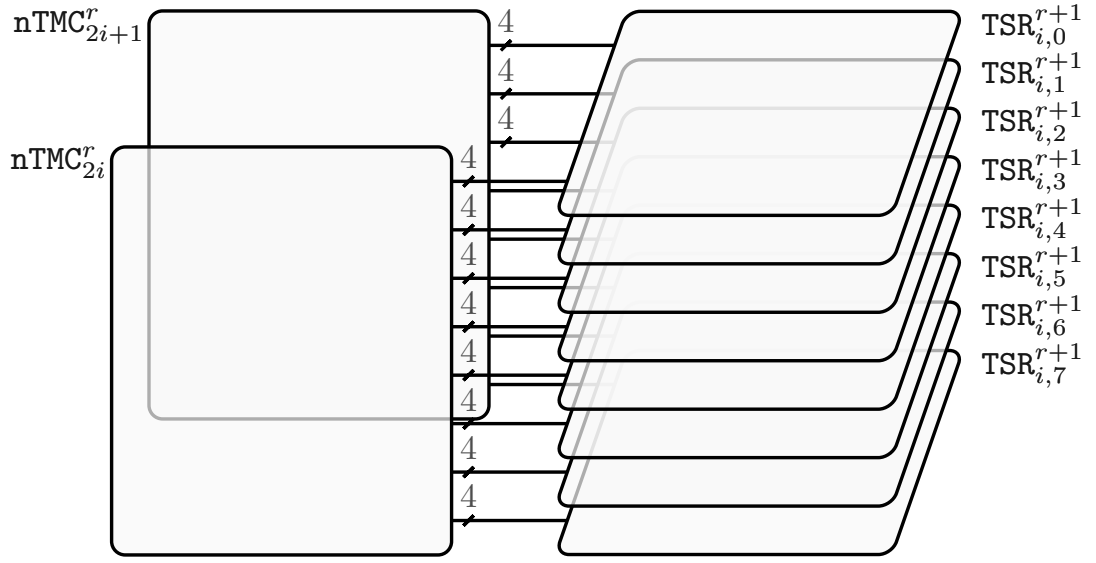


图 4-4 nTMC 和 TSR 的连接方式.

Fig 4-4 The binding form of nTMC and TSR.

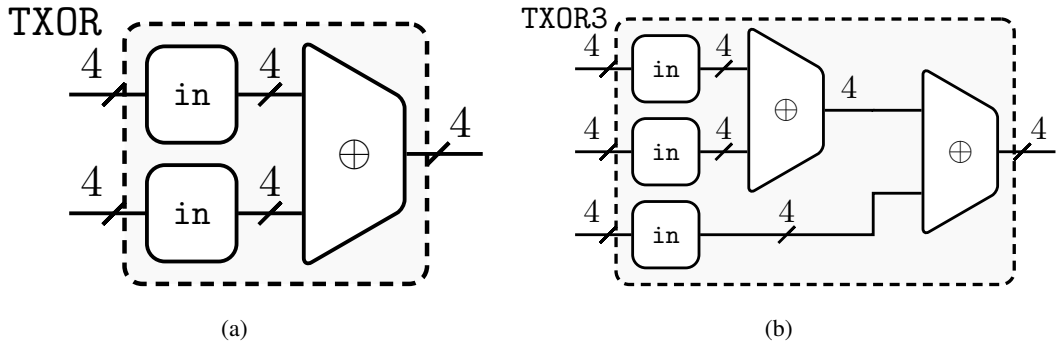


图 4-5 TXOR(a) 和 TXOR3(b) 的结构.

Fig 4-5 The structures of TXOR (a) and TXOR3 (b).

个 TXOR，把 32 个半字节合并为 12 个；在第二层和第三层，我们分别使用六个和三个 TXOR，将 12 个半字节先合并为六个，再到三个；在底层，我们使用一个 TXOR3，将三个半字节合并为一个半字节，而这就是最终的结果。用这种双表配合的方法，在整个异或运算的阶段，仅有 22 次查表操作而不是在仅使用 TXOR3 的情况下所需的 31 次，这减少了 29% 的访问时间。

Require: $ciphertext, k^r$ ($1 \leq r \leq 11$)

Ensure: $plaintext$

```

state  $\leftarrow ciphertext$ 
InvShiftRoundKey( $k^{11}$ )
state  $\leftarrow$  InvShiftRows( $state$ )
state  $\leftarrow$  AddRoundKey( $state, \tilde{k}^{11}$ )
state  $\leftarrow$  InvSubBytes( $state$ )
state  $\leftarrow$  AddRoundKey( $state, k^{10}$ )
state  $\leftarrow$  InvMixColumns( $state$ )
for  $r \leftarrow 9$  downto 2 do
    state  $\leftarrow$  InvShiftRows( $state$ )
    state  $\leftarrow$  InvSubBytes( $state$ )
    state  $\leftarrow$  AddRoundKey( $state, k^r$ )
    state  $\leftarrow$  InvMixColumns( $state$ )
end for

```

图 4-6 一个等价的 AES 解密过程.

Fig 4-6 An equivalent AES decryption process.

4.4 解密过程

在实现解密过程之前，对标准的 AES 解密过程，我们需要重新定义其轮函数的边界并且调整轮函数中各项操作的顺序。在调整之后，我们得到一个等价的解密算法，如图4-6所示。

我们注意到上述算法中有几个操作与加密算法中的对应操作有些许不同：在解密算法中，步骤 1 执行 ShiftRows 的逆操作 InvShiftRows；步骤 3 使用了 T_i^r 的逆映射 IT_i^r 以及 MC 的逆矩阵 IMC 的分块子矩阵 IMC_i 。具体的， IMC_j ($j = 0, 1$) 表示了 InvMixColumns 操作对应的矩阵 IMC 的一个 4×2 子矩阵，并有

$$IMC = [IMC_0 | IMC_1]$$

而逆 T-box: IT_i^r 能够表示为:

$$IT_i^r(x) = \begin{cases} IS(x \oplus \tilde{k}_i^{11}) \oplus k_i^{10} & r = 1 \\ IS(x) \oplus k_i^{11-r} & 2 \leq r \leq 10 \end{cases}$$

其中 IS 表示了标准 AES 算法中的逆 S-box, 而 \tilde{k}^r 则为子密钥 k^r 经过 `InvShiftRows` 操作之后的结构, 即

$$\tilde{k}^r = \text{InvShiftRows}(k^r)$$

尽管加密与解密过程之间有细微的不同, 我们还是可以认为解密过程的基本流程结构与加密过程的是一致的, 而它们所使用的查找表也是类似的。

4.5 本章总结

本章描述了一种新的白盒 AES 实现方案, 该方案的设计思路依然是利用查表操作替代传统的密码操作以隐藏中间过程。我们在实现的过程中采用了更大的、包含了两个密钥字节的查找表: 16 to 32-bit 查找表, 并且利用了非线性混淆的技术, 生成攻击者无法进行秘钥提取的查找表。

新方案包含十轮迭代, 而每轮迭代则包含三个连续的步骤: 步骤 1 包含了传统算法中的 `ShiftRows` 操作; 步骤 2 作为一个提供辅助功能的步骤, 负责异或运算; 步骤 3 包含了 `AddRoundKey`, `SubBytes` 和 `MixColumns`。而每个步骤都被实现为一个由同一类型查找表组成的集合。

在分析了 Xiao-Lai 白盒实现能够抵御 BGE 攻击却被 De Mulder 等人的攻击攻破的原因之后, 设计了立体的查找表关联方式, 以抵御这两种攻击。

下一章将具体分析新方案的性能以及白盒安全性, 并特别阐述了其对 BGE 攻击和 De Mulder 等人的攻击的防护。

第五章 新方案的性能及白盒安全性分析

在本章中，我们将对上一章中描述的白盒 AES 实现方案的性能及白盒安全性进行分析。分析的内容包括：性能方面，算法的存储消耗以及对内存的访问量；在白盒安全性方面，衡量方案的白盒安全性指标，以及它在特定攻击下对密钥的保护情况。

5.1 算法实现的性能

我们首先评估我们的新方案的运行性能，包括理论上程序的体积以及运行时对查找表的访问情况。

nTMC_i^r 是从 16-bit 输入到 32-bit 输出的查找表，因此每个表需要 $2^{16} \times 32\text{bit}$ ，即 256kB 空间； $\text{TSR}_{i,j}^r$ 是 8 to 128-bit 查找表，每个占用 $2^8 \times 128\text{bit} = 4\text{kB}$ 空间；8 to 4-bit 的查找表 TXOR，每张占用 $2^8 \times 4\text{bit} = 128\text{B}$ 空间，而 12 to 4-bit 的查找表 TXOR3，每张占用 $2^{12} \times 4\text{bit} = 2\text{kB}$ 空间。

在加密和解密过程中，每一轮都包含八个 nTMC_i^r 。第一轮包含了 16 个 $\text{TSR}_{i,j}^1$ 、 $32 \times (2+3) = 160$ 个 TXOR 以及 $32 \times (4+1) = 160$ 个 TXOR3；在第二至第九轮的每一轮中，32 个 $\text{TSR}_{i,j}^r$ 、 $32 \times 13 = 416$ 个 TXOR 以及 $32 \times 9 = 288$ 个 TXOR3 被构造出来。由于最后一轮包含了两个步骤 1 及步骤 2，共有 64 个 $\text{TSR}_{i,j}^r$ ($r = 10, 11$)、832 个 TXOR 以及 576 个 TXOR3 被包含其中。因此，对我们的白盒方案的每个过程（加密或解密）都将使用 80 个 nTMC_i^r 、336 个 $\text{TSR}_{i,j}^r$ 、4320 个 TXOR 以及 3040 个 TXOR3，这些查找表合计将占用 $80 \times 256\text{kB} + 336 \times 4\text{kB} + 4320 \times 128\text{B} + 3040 \times 2\text{kB} = 28444\text{kB}$ 空间。而由于每一个表在加密/解密过程中都会被访问到且仅会被访问一次，而且我们的方案除了访问查找表外没有其余的操作，因而在整个加密/解密过程中共有 $80 + 336 + 4320 + 3040 = 7776$ 次查表操作。

新方案的性能指标总结如表5-1

表 5-1 新方案所包含的查找表的数量和空间占用量.

Table 5-1 The number of tables and the corresponding memory consumption of the new scheme

Table Type	Size of Each	Amount	Size
nTMC	256kB	80	20480
TSR	4kB	336	1344
TXOR	128B	4320	540
TXOR	2kB	3040	6080
TOTAL	—	—	28444

5.2 算法实现的白盒安全性衡量指标

我们已经知道了衡量加密算法的白盒实现的白盒安全性需要用到的安全性指标：白盒多样性以及白盒含混度。在计算这两个指标之前，我们必须首先引入两个基本的计算公式来计算查找表中非奇异矩阵或列满秩矩阵的总数：

1. 由有限域 $\mathbb{GF}(q)$ 上全体 n 阶非奇异矩阵构成的集合的大小可以用如下公式计算：

$$size = \prod_{k=0}^{n-1} (q^n - q^k) \quad (5-1)$$

2. 而更一般的，由有限域 $\mathbb{GF}(q)$ 上全体 $m \times n$ -bit 列满秩矩阵构成的集合的大小可以利用公式：

$$size = 2^{n(m-n)} \times \prod_{k=0}^{n-1} (q^n - q^k) \quad (5-2)$$

举例而言，根据式5-1，有限域 $\mathbb{GF}(2)$ 上的 4×4 -bit 非奇异矩阵共有 20160 个。根据式5-1、5-2这两个公式，我们能够计算所提方案中每种查找表的白盒多样性：

- $\text{nTMC}_i^r: (2^4!)^{12} \times 2^{254} \times (2^8)^2 \times 2^{1023} \approx 2^{1824}$
- $\text{TSR}_{i,j}^1: (2^4!)^{34} \times 20160^{64} \times (2^{254})^8 \approx 2^{4452}$
- $\text{TSR}_{i,j}^r: (2^4!)^{34} \times (2^{126})^2 \times (2^{254})^8 \approx 2^{3789}$

- $\text{TSR}_{i,j}^{11}: (2^4!)^{34} \times (2^{126})^2 \times 20160^{256} \approx 2^{5417}$
- $\text{TXOR}: (2^4!)^2 \times 2^4! \approx 2^{132.8}$
- $\text{TXOR3}: (2^4!)^3 \times 2^4! \approx 2^{177}$

相比白盒多样性，白盒含混度的估计就显得更为复杂。步骤 2 中使用的 TXOR 和 TXOR3 是相对而言较为容易估计的：

- $\text{TXOR}: (2^4!) \times 2^4 \approx 2^{48.2}$
- $\text{TXOR3}: (2^4!) \times 2^8 \approx 2^{52.2}$

对于其余两种查找表，我们给出其白盒含混度的下界：

- $\text{TSR}_{i,j}^r: (2^4!)^2 \times 20160^{32} \approx 2^{546.1}$
- $\text{nTMC}_i^r: 2^{254.2} \times 2^{16} \approx 2^{270.2}$

5.2.1 算法实现在 BGE 攻击下的安全性

我们将会证明 BGE 攻击无法应用在我们提出的新方案上，其原因在于 BGE 攻击所依赖的最为核心的性质——输出字节对 (y_i, y_j) 之间唯一的线性相关性——并不一定能够成立。

由于内置密钥的查找表 nTMC 已经过了混淆，对其进行单独分析难度很大，故我们考虑将 nTMC 和 TSR 结合在一起进行分析。对于 $2 \leq r \leq 10$ 、 $0 \leq i \leq 15$ ，我们令 x_i^{r-1} 表示第 $r-1$ 轮中步骤 3 的第 i 个输入字节，令 y_i^r 表示第 r 轮中步骤 1 的第 i 个输出字节，然后定义 8-bit 值 u_i 和 v_i 使得它们满足

$$\begin{aligned} (u_{2i} \| u_{2i+1})^\top &= L_i^{r-1} \cdot (x_{2i}^{r-1} \| x_{2i+1}^{r-1})^\top \\ (y_{2i}^r \| y_{2i+1}^r)^\top &= (L_i^r)^{-1} \cdot (v_{2i} \| v_{2i+1})^\top \end{aligned}$$

即

$$(v_{2i} \| v_{2i+1})^\top = L_i^r \cdot (y_{2i}^r \| y_{2i+1}^r)^\top$$

此外，我们把现行混淆双射 $(L_i^r)^{-1}$ 表示为如下形式：

$$(L_i^r)^{-1} = \begin{bmatrix} (L_i^r)_0^{-1} & (L_i^r)_1^{-1} \\ (L_i^r)_2^{-1} & (L_i^r)_3^{-1} \end{bmatrix}$$

其中 $(L_i^r)_s^{-1}$ ($0 \leq s \leq 3$) 表示矩阵 $(L_i^r)^{-1}$ 的一个 8×8 -bit 的子矩阵, 而该子矩阵有可能不可逆。

利用上述的定义和记号, u_i 与 v_i 之间的关系能够被表示为

$$v_j = \bigoplus_{k=0}^3 mc_{l,k} \otimes T_{4i+k}^r(u_{4i+k})$$

其中 $l = isr(j) \bmod 4$ 、 $i = \lfloor isr(j) \rfloor$, 而 $isr(j)$ 表示下标 j 经过行位移逆变换 `InvShiftRows` 的结果。以 v_0 和 v_1 为例, 我们能够将其表示为

$$\begin{aligned} v_0 &= '02' \otimes T_0^r(u_0) \oplus '03' \otimes T_1^r(u_1) \oplus '01' \otimes T_2^r(u_2) \oplus '01' \otimes T_3^r(u_3) \\ v_1 &= '01' \otimes T_4^r(u_4) \oplus '02' \otimes T_5^r(u_5) \oplus '03' \otimes T_6^r(u_6) \oplus '01' \otimes T_7^r(u_7) \end{aligned}$$

进一步的, 输出字节 y_{2i}^r 和 y_{2i+1}^r 能够被表示如下

$$\begin{aligned} y_{2i}^r &= [(L_i^r)_0^{-1} (L_i^r)_1^{-1}] \cdot (v_{2i} \| v_{2i+1})^\top \\ y_{2i+1}^r &= [(L_i^r)_2^{-1} (L_i^r)_3^{-1}] \cdot (v_{2i} \| v_{2i+1})^\top \end{aligned}$$

然而, 由于可逆矩阵 $(L_i^r)^{-1}$ 的子矩阵 $(L_i^r)_s^{-1}$ 有可能是不可逆的, 我们能够断言 y_{2i}^r 和 y_{2i+1}^r 之间不一定必然存在仿射关系。除此之外, 如果定义输出双字节

$$\begin{aligned} Y_i^r &= (y_{2i}^r \| y_{2i+1}^r) \\ Y_j^r &= (y_{2j}^r \| y_{2j+1}^r) \end{aligned}$$

则 Y_i^r 和 Y_j^r 之间的关系也同样无法确定, 这是因为二者所牵扯的变量以及 `T-box` 都不完全相同。

5.2.2 算法实现在 De Mulder 等人提出的攻击下的安全性

接下来我们将阐述我们所提出的白盒 AES 实现方案能够抵御 De Mulder 等人的攻击。

首先, 在我们的方案中, 所有查找表都加入了非线性输入输出编码, 因此 De Mulder 等人的攻击所利用的“线性”等价算法是无法直接应用在我们的方案中的。

其次, 即使我们不考虑这些非线性编码, 攻击者仍然不得不面对更大的解集: 相比于对 Xiao-Lai 白盒 AES 实现进行攻击时获得的大小为 2^8 的解集, 攻

击我们的方案产生的解集为 2^{24} ，这是由于我们的方案采用了立体的结构，查找表之间的连接方式使得查找表 $\text{TSR}_{i,j}^r$ 每个输出字节都与状态矩阵中每一列的四个字节相关，而这四个字节正是查找表 nTMC_{2i}^{r-1} 和 nTMC_{2i+1}^{r-1} 的输入。更换言之，攻击者所能利用的用于提取输入编码 L_i^r 中蕴含关键信息的关系，由于查找表之间新的连接方式已经变得比原先更加复杂：

$$\begin{aligned} u_0 &= f_l^t(u_1, u_2, u_3) \\ &= (\bar{S})^{-1} [mc_{l,0}^{-1} \otimes (mc_{l,1} \\ &\quad \otimes \bar{S}(u_1) \oplus mc_{l,2} \otimes \bar{S}(u_2) \oplus mc_{l,3} \otimes \bar{S}(u_3))] \end{aligned}$$

其中 $mc_{l,i}$ ($0 \leq l, i \leq 3$) 表示矩阵 MC 第 l 列的系数而 \bar{S} 则表示一个新的由标准 AES 算法中的 S-box 导出的 8-bit 的双射 S-box，其定义为

$$\bar{S} = S \circ \oplus_{52}$$

上述的 u_i 之间的关系的结果就是攻击者会得到一个比原先大得多的解集 \mathcal{S}_l^t ($0 \leq t \leq 3$):

$$\begin{aligned} \mathcal{S}_l^t &= \left\{ x = (x_0 \| x_1) \in \mathbb{GF}(2^{32}) \mid L_{2t}^1(x_0) = u_0 \| u_1 \right. \\ &\quad \left. \wedge L_{2t+1}^1(x_1) = u_2 \| u_3 \wedge u_0 = f_l^t(u_1, u_2, u_3) \right\} \end{aligned}$$

其大小达到了 $|\mathcal{S}_l^t| = 2^{24}$ 。由于解集的规模，De Mulder 等人的攻击在我们的方案上的时间复杂度至少为 2^{64} ，这对于白盒密码的实际应用场景而言已经称得上是比较不错的安全性下界了。当然这还是没有考虑非线性的编码的情况，如果考虑先将非线性编码剥离，再实施 De Mulder 等人的攻击，那么时间复杂度将会更高。

5.3 本章总结

本章分析新方案的性能及其白盒安全性。分析的内容包括：性能方面，算法的存储消耗以及对内存的访问量；在白盒安全性方面，衡量方案的白盒安全性指标，以及它在特定攻击下对密钥的保护情况。

在性能方面，新方案的加密/解密过程只包含查表操作，各类查找表的数量为：80 个 nTMC、336 个 TSR、4320 个 TXOR 以及 3040 个 TXOR3，这合计将占用

28444kB 存储空间。而由于每一个表在加密/解密过程中都会被访问到，并且仅会被访问一次，因而在整个加密/解密过程中共有 $80 + 336 + 4320 + 3040 = 7776$ 次查表操作。

在安全性方面，首先，白盒多样性及白盒含混度两个指标都表现不错，特别是密钥相关的查找表 **nTMC**，其白盒多样性达到了 2^{1824} ，而其白盒含混度下界也高达 $2^{270.2}$ 。

其次，我们阐述了我们认为 **BGE** 攻击不适用于新方案的原因：**BGE** 攻击所依赖的最为核心的性质——输出字节对 (y_i, y_j) 之间唯一的线性相关性——在新方案上并不一定能够成立。

此外，我们也说明了新方案能够抵御 **De Mulder** 等人的攻击的两点原因：其一，新方案中的所有查找表都加入了非线性输入输出编码，因此 **De Mulder** 等人的攻击所利用的“线性”等价算法无法直接应用在我们的方案中；其二，由于新方案采用了立体的结构，攻击者在应用线性等价算法时得到的解集从 2^8 上升到了 2^{24} ，这使得攻击的时间复杂度至少为 2^{64} 。

全文总结

主要贡献

本文描述了一种全新的基于查找表的白盒 AES 实现，该实现对 BGE 攻击和 De Mulder 等人的攻击都显示出了很好的防护性。新方案采用了更大的密钥相关查找表 **nTMC**——一个 16 to 32-bit 的映射——来实现三步密码操作 **AddRoundKey**、**SubBytes** 和 **MixColumns** 的组合。每个查找表 **nTMC** 都内置了两个轮密钥字节，并且经过随机混淆双射的编码。

除每一轮的 16 个查找表 **nTMC** 之外，我们设计了 32 个 8 to 128-bit 的密钥无关查找表 **TSR** 来实现 **ShiftRows** 操作。为了使 **nTMC** 和 **TSR** 以一种安全的方式协同工作，我们设计了一种立体连接模式，其中每个 **TSR** 分别从两个 **nTMC** 的对应输出位置上各取一个半字节来作为自身的 8-bit 输入。32 个 **TSR** 的输出通过一个由查找表 **TXOR** 和 **TXOR3** 所构成的树形网络结构被合并成为一个 128-bit 的值。由于新方案中全体查找表都经过了非线性的随机编码，因此想要正确的合并对编码后的结果必须先对其解码再进行异或运算，而查找表 **TXOR** 和 **TXOR3** 正是因此而生。

在新方案设计完毕之后，我们对其进行了详细的分析，包括性能方面和白盒安全性方面。在性能方面，新方案中的加密和解密过程均只包含查表操作，对各类查找表的统计结果为：80 个 **nTMC**、336 个 **TSR**、4320 个 **TXOR** 以及 3040 个 **TXOR3**，共计消耗 28444kB 空间。由于每一个表在加密/解密过程中都被访问且仅有一次，因而在整个加密/解密过程中共有 $80 + 336 + 4320 + 3040 = 7776$ 次查表操作。而在安全性方面，我们首先计算了各类查找表的白盒多样性及白盒含混度，结果较好，特别是密钥相关的查找表 **nTMC**，其白盒多样性达到了 2^{1824} ，而其白盒含混度下界也高达 $2^{270.2}$ 。其次，我们阐述了我们认为 BGE 攻击不适用于新方案的原因，即 BGE 攻击所依赖的最为核心的性质——输出字节对 (y_i, y_j) 之间唯一的线性相关性——在新方案上并不一定能够成立。此外，我们也说明了新方案能够抵御 De Mulder 等人的攻击的两点原因：其一，新方案中的所有查找表都加入了非线性输入输出编码，因此 De Mulder 等人的攻击所利用的“线性”等价算法无法直接应用在我们的方案中；其二，由于新方案采

用了立体的结构，攻击者在应用线性等价算法时得到的解集从 2^8 上升到了 2^{24} ，这使得攻击的时间复杂度至少为 2^{64} 。而这在实际应用层面已经能够满足安全性要求。

将来的工作

将来的研究应当着眼于在更一般的情景下证明白盒实现的安全性，也就是说，证明其安全性不是仅仅针对在某些特定场景下的特定的攻击而言，而是在一般的情况下，对所有典型的攻击而言。一旦我们能证明其在一般情况下的安全性，那么方案的应用范围就将显著扩大。

此外，查找表的性能尚存很大改进空间，其查找表数量以及总体存储空间占用量均较多，影响了读写性能并且占用了较多的空间。如何减少查找表数量以减少内存访问以及如何降低查找表空间占用以降低资源占用都是值得继续研究的方向。

参考文献

- [1] RIVEST R L. Cryptography[C]//Handbook of theoretical computer science (vol. A). .[S.l.]: [s.n.] , 1991:617–755.
- [2] KOCHER P C. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems[C]//Advances in Cryptology—CRYPTO’ 96. .[S.l.]: [s.n.] , 1996:104–113.
- [3] CHOW S, EISEN P, JOHNSON H, et al. White-box cryptography and an AES implementation[C]//Selected Areas in Cryptography. .[S.l.]: [s.n.] , 2003:250–270.
- [4] SHAMIR A, VAN SOMEREN N. Playing ‘hide and seek’ with stored keys[C]//Financial cryptography. .[S.l.]: [s.n.] , 1999:118–124.
- [5] HALDERMAN J A, SCHOEN S D, HENINGER N, et al. Lest we remember: cold-boot attacks on encryption keys[J]. Communications of the ACM, 2009, 52(5):91–98.
- [6] PUB N F. 46-3. Data Encryption Standard[J]. Federal Information Processing Standards, National Bureau of Standards, US Department of Commerce, 1977.
- [7] AES N. Advanced encryption standard[J]. Federal Information Processing Standard, FIPS-197, 2001, 12.
- [8] CHOW S, EISEN P, JOHNSON H, et al. A white-box DES implementation for DRM applications[M]//Digital Rights Management.[S.l.]: Springer, 2003:1–15.
- [9] JACOB M, BONEH D, FELTEN E. Attacking an obfuscated cipher by injecting faults[M]//Digital Rights Management.[S.l.]: Springer, 2003:16–31.
- [10] LINK H E, NEUMANN W D. Clarifying obfuscation: improving the security of white-box DES[C]//Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on. .[S.l.]: [s.n.] , 2005 , 1:679–684.

- [11] WYSEUR B, MICHIELS W, GORISSEN P, et al. Cryptanalysis of white-box DES implementations with arbitrary external encodings[C]//Selected Areas in Cryptography. .[S.l.]: [s.n.] , 2007:264–277.
- [12] GOUBIN L, MASEREEL J M, QUISQUATER M. Cryptanalysis of white box DES implementations[C]//Selected Areas in Cryptography. .[S.l.]: [s.n.] , 2007:278–295.
- [13] BILLET O, GILBERT H, ECH-CHATBI C. Cryptanalysis of a white box AES implementation[C]//Selected Areas in Cryptography. .[S.l.]: [s.n.] , 2005:227–240.
- [14] MICHIELS W, GORISSEN P, HOLLMANN H D. Cryptanalysis of a generic class of white-box implementations[C]//Selected Areas in Cryptography. .[S.l.]: [s.n.] , 2009:414–428.
- [15] TOLHUIZEN L. Improved cryptanalysis of an AES implementation[C]//Proceedings of the 33rd WIC Symposium on Information Theory in the Benelux, Boekelo, The Netherlands, May 24–25, 2012. .[S.l.]: [s.n.] , 2012.
- [16] BRINGER J, CHABANNE H, DOTTAX E. White Box Cryptography: Another Attempt.[J]. IACR Cryptology ePrint Archive, 2006, 2006:468.
- [17] XIAO Y, LAI X. A secure implementation of white-box AES[C]//Computer Science and its Applications, 2009. CSA'09. 2nd International Conference on. .[S.l.]: [s.n.] , 2009:1–6.
- [18] KARROUMI M. Protecting white-box AES with dual ciphers[M]//Information Security and Cryptology-ICISC 2010.[S.l.]: Springer, 2011:278–291.
- [19] DE MULDER Y, WYSEUR B, PRENEEL B. Cryptanalysis of a perturbed white-box AES implementation[M]//Progress in Cryptology-INDOCRYPT 2010.[S.l.]: Springer, 2010:292–310.
- [20] DE MULDER Y, ROELSE P, PRENEEL B. Cryptanalysis of the Xiao–Lai White-Box AES Implementation[C]//Selected Areas in Cryptography. .[S.l.]: [s.n.] , 2013:34–49.

- [21] DE MULDER Y, ROELSE P, PRENEEL B. Revisiting the BGE Attack on a White-Box AES Implementation.[J]. IACR Cryptology ePrint Archive, 2013, 2013:450.
- [22] DAEMEN J, RIJMEN V. AES proposal: Rijndael[J]. 1998.
- [23] DAEMEN J, RIJMEN V. The design of Rijndael: AES-the advanced encryption standard[M].[S.l.]: Springer, 2002.
- [24] SHANNON C E. Communication Theory of Secrecy Systems*[J]. Bell system technical journal, 1949, 28(4):656–715.
- [25] VERNAM G S. Cipher printing telegraph systems: For secret wire and radio telegraphic communications[J]. AIEE, Journal of the, 1926, 45(2):109–115.
- [26] MATSUI M. Linear cryptanalysis method for DES cipher[C]//Advances in Cryptology—EUROCRYPT’ 93. .[S.l.]: [s.n.] , 1994:386–397.
- [27] BIHAM E, SHAMIR A. Differential cryptanalysis of DES-like cryptosystems[J]. Journal of CRYPTOLOGY, 1991, 4(1):3–72.
- [28] KNUDSEN L R, RIJMEN V. Known-key distinguishers for some block ciphers[M]//Advances in Cryptology—ASIACRYPT 2007.[S.l.]: Springer, 2007:315–324.

致 谢

在此谨向在我攻读硕士研究生期间给予我帮助的诸位老师、同学、亲友致以衷心的感谢。

首先感谢我的导师来学嘉教授这两年多来对我的帮助、指导和教诲。来教授治学严谨、学术造诣高深，是在国内外享有盛誉的著名密码学家。来教授在学术上对我的严格要求和悉心指导使我不断进步，对我的教诲令我受益终生。

感谢龙宇老师在密码算法与协议的相关问题上给予我指导；感谢林婷婷与我就方案设计进行的富有意义的讨论；感谢黄格仕对插图绘制方面的耐心指点；感谢方习文对文章叙述上所提出的宝贵意见。

谨以此文献给我挚爱的双亲和一直给予我陪伴的刘婧姝。

希望此文不是我学术思考的终点。

攻读学位期间发表的学术论文目录

- [1] R. LUO, X. LAI AND R. YOU. "A New Attempt of White-box AES Implementation"[C]. In Security, Pattern Analysis and Cybernetics, 2014. ICSPAC'2014. International Conference on. IEEE, 2014.
- [2] R. LUO AND W. ZHAO. "A Robust Iterative Algorithm for Parameter Estimation of the Generalized Gamma Distribution"[C]. In Computational Science and Engineering, 2014. IEEE CSE'2014. 17th International Conference on. IEEE, 2014.

攻读学位期间参与的项目

[1] 自然科学基金项目 61073149、61272440、61472251

[2] 中国博士后科学基金 2013M531174、2014T70417


上海交通大学

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权上海交通大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于 ☐ 保密，在____年解密后适用本授权书。
☒ 不保密。

(请在以上方框内打“√”)

学位论文作者签名：

指导教师签名：

日期：2015年 1 月 21 日

日期： 年 月 日

上海交通大学

学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：罗睿

日期：2015年 1 月 21日