AES算法的轻量化实现研究

赵跃华,马林林
ZHAO Yuehua, MA Linlin

江苏大学 计算机科学与通信工程学院,江苏 镇江 212013
Institute of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, Jiangsu 212013, China

ZHAO Yuehua, MA Linlin. Research of AES's lightweight complementation technique. Computer Engineering and Applications, 2015, 51(6):79-83.

Abstract: As a new generation of advanced encryption standard selected by NIST, AES's fast software implementations still occupy large storage space, which is bad for its applications in resource-constrained environments. Against the problem, a new lightweight implementation method of AES is presented. According to the characteristics of the round transformation equation, the method changes and merges its steps to optimize the algorithm. Compared with other optimization methods implemented with VC++6.0, the new algorithm is testified that it can not only reduce the storage space, but also improve the execution efficiency.

Key words: Advanced Encryption Standard(AES); Sbox; optimizations; round function; implementation technique

摘 要:作为被NIST选定的新一代高级加密标准,AES算法的快速软件实现仍占用较大的存储空间,这不利于其在资源受限环境中的应用。针对该问题,提出了一种AES轻量化的实现方法。该方法根据轮函数的特点,对其进行调序后合并与优化,以减少算法占用的存储空间,并提高算法的执行效率。在VC++6.0平台上与其他优化实现方案进行实验比较。结果表明,改进后的算法对存储空间要求较低,且执行效率较高。

关键词:高级加密标准(AES); S盒; 优化; 轮函数; 实现技术
文献标志码:A **中图分类号:**TP309.7 doi:10.3778/j.issn.1002-8331.1305-0440

1 引言

作为新一代高级加密标准,AES算法的实现研究备受密码学界的关注[1-3]。其软件实现[4-7]主要是用高级语言快速实现算法的加解密,算法执行效率和算法占用存储空间的大小是衡量AES软件实现方式优劣的主要因素。

文献[8]提出用两个256 Byte的表格分别存储S盒和逆S盒,该实现方法占用内存小,但算法的执行效率却很低;为提高算法执行效率,文献[8]又提出在加解密流程中分别使用256个4 Byte条目的表,该方法需要8 KB的存储空间,这并不适用于一些资源受限的环境;文献[9]对文献[8]进行改进,利用表格复用技术,在保证算法执行效率的同时,减少内存占用,但该实现方法仍然占用较大的内存。本文将对这些软件实现方法进行分析,在文献[9]的基础上,综合考虑算法执行效率和实现算法所需存储空间,对AES-128进行改进,设计一个轻量且高效的软件实现方法。

2 AES算法描述

AES是一个迭代型分组密码,算法包括加密、解密和密钥扩展三个模块[10]。轮函数是AES算法的核心模块,由四个不同的变换组成,分别为:字节替换SUB、行位移SR、列混淆MC和轮密钥加ADK。最后一轮中没有列混淆操作。文献[11]给出了AES算法设计原理。

2.1 AES算法的加密过程

设第 i 轮输入数据为 $S_i(i=1,2,\cdots,Nr)$,密钥 $K_i(i=1,2,\cdots,Nr)$,初始信息为 S_0,初始密钥为 K_0。加密操作如下。

作者简介:赵跃华(1958—),男,博士,教授,研究领域为信息安全、嵌入式操作系统;马林林(1987—),女,硕士研究生。
E-mail:zhaoyh@ujs.edu.cn

收稿日期:2013-05-31 **修回日期:**2013-08-26 **文章编号:**1002-8331(2015)06-0079-05
CNKI**网络优先出版:**2013-09-24,http://www.cnki.net/kcms/detail/11.2127.TP.20130924.0941.007.html

(1)RoundKey:初始信息 S_0 与初始密钥 K_0 异或,得 $S_0 \oplus K_0$。

(2)SUB:对 S_i 上的每个字节进行替换。它是AES算法中唯一的非线性运算。用函数 $S(x)$ 表示SUB操作,可由两步计算而得:

首先,在有限域 GF(2^8) 内,对 S_i 中每字节求乘法的逆运算,即 $g(x)=x^{-1}$。

再对 $g(x)$ 进行仿射变换,即 $f(g(x))$。将每个字节记作 $(b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0)$,仿射变换即对字节中每位作变换:$b'_i = b_i \oplus b_{(i+4)\bmod 8} \oplus b_{(i+5)\bmod 8} \oplus b_{(i+6)\bmod 8} \oplus b_{(i+7)\bmod 8} \oplus c_i$,其中,$c_i$ 为(01100011)中的第 i 位。则有:

$$y = S(x) = f(g(x)) \tag{1}$$

可用查表法来实现S盒的字节替换功能,表所用空间为256 Byte。

(3)SR:对 S_i 的每行按不同的位移量进行行移位。

(4)MC:作用在 S_i 列上的替换操作。将 S_i 的列看作是 $GF(2^8)$ 上的多项式,并在模 x^4+1 下与给定多项式 $c(x)$ 相乘,$c(x)$ 为:

$$c(x) = 03 \cdot x^3 + 01 \cdot x^2 + 01 \cdot x + 02 \tag{2}$$

MC变换可以表示成:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{3}$$

(5)ADK:将 S_i 中的每字节与轮密钥 K_i 异或。

2.2　AES算法的解密过程

AES解密流程是加密流程的逆序,设第 i 轮输入数据为 $S'_i(i=1,2,\cdots,Nr)$,密钥 $IK_i(i=1,2,\cdots,Nr)$,初始信息为 S'_0,初始密钥为 IK_0。其操作如下:

(1)IRoundKey:S'_0 与会话逆密钥 IK_0 异或。

(2)ISUB:作用在 S'_i 上的逆字节替换。由式(1)得:

$$x = S^{-1}(y) = g^{-1}(f^{-1}(y)) = g(f^{-1}(y)) \tag{4}$$

此操作亦可用查表法实现,表所用空间为256 Byte。

(3)ISR:对 S'_i 的每一行按不同的位移量进行逆行移位。

(4)IMC:将 S'_i 的每一列在模 x^4+1 下与固定多项式 $d(x)$ 相乘变换,$d(x)$ 为:

$$d(x) = 0\mathrm{B} \cdot x^3 + 0\mathrm{D} \cdot x^2 + 09 \cdot x + 0\mathrm{E} \tag{5}$$

IMC可以表示成:

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0\mathrm{E} & 0\mathrm{B} & 0\mathrm{D} & 09 \\ 09 & 0\mathrm{E} & 0\mathrm{B} & 0\mathrm{D} \\ 0\mathrm{D} & 09 & 0\mathrm{E} & 0\mathrm{B} \\ 0\mathrm{B} & 0\mathrm{D} & 09 & 0\mathrm{E} \end{bmatrix} \times \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} \tag{6}$$

(5)IADK:将 S'_i 中的每字节与会话逆密钥 IK_i 异或。

3　AES算法的轻量优化

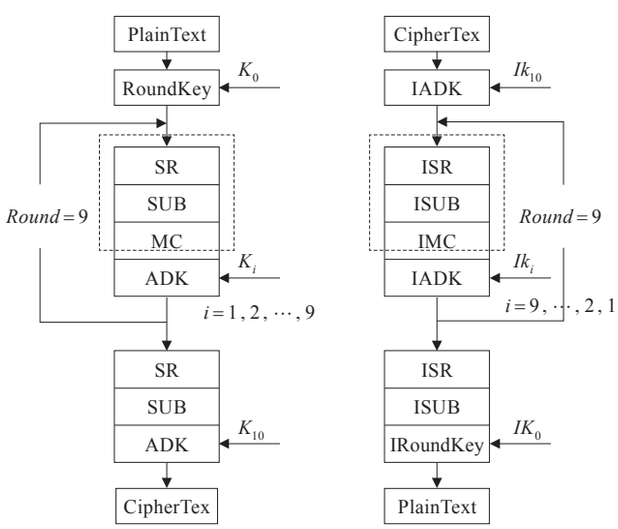
3.1　AES加解密过程的合并优化

AES加解密轮函数具有以下两条性质[12]:

(1)SUB/ISUB对输入信息的每个字节进行非线性替换,此操作与字节的位置无关;SR/ISR只对字节进行循环移位操作,而不改变字节的内容。

(2)MC/IMC和ADK/IADK均是线性操作,都有 $M(X+K)=M(X)+M(K)$ 成立。

为提高算法实现效率,本文利用上述两条性质,对AES加解密过程进行调序。在加密过程中将SUB与SR调序,在解密过程中将IMC与IADK调序。调序后的AES加解密流程如图1所示。



(a)调序后的加密流程　(b)调序后的解密流程

图1　调序后的加解密流程

调序后的AES加密轮函数的运算顺序依次为SR、SUB、MC和ADK,由于SUB与MC均作用于信息字节上,即可采用查表法同步实现。与原实现方案相比,此步优化减少了加密算法的复杂度,且提高算法的执行效率。此外,由于SR不改变字节的内容,只改变字节位置,可以用简单的数组下标选择实现,从而可同步实现SR、SUB和MC,进一步提高算法执行效率。

调序后的AES解密轮函数的运算顺序依次为ISR、ISUB、IMC和IADK。同理,调序后的AES解密轮函数亦可同步实现ISR、ISUB和IMC操作。与原实现方案的解密过程相比,利用查表法同步实现复杂且耗时的ISUB和IMC操作,虽然增加了一些实现算法所占用的存储空间,但却大大提高了解密算法的执行效率。

3.2　AES轮函数的优化

AES算法的实现涉及到异或运算(轮密钥加)、变量与常量的乘法(列混淆)、数组下标的选择(循环移位)

等,且每一轮的轮变换中只有SUB/ISUB是非线性操作,它是影响算法计算效率的主要因素。此外,相对异或等运算,变量与常量的乘法运算(MC/IMC)也并不容易实现[13-14]。为提高AES的软件实现计算速度,在存储空间允许的情况下,通常采用查表机制[15]。现有软件实现AES算法的方法有:

方法一,即公开的未经优化的AES例程,加解密过程中分别使用表 $S(x)$ 和逆 $S(x)$ 表,表占用空间为512 Byte。此方法在2.1节与2.2节中已作介绍。

方法二是对方法一的改进[8]。设经过MC和ADK后的输出为 y ,则有:

$$\begin{bmatrix} y_{4i} \\ y_{4i+1} \\ y_{4i+2} \\ y_{4i+3} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \otimes \begin{bmatrix} S_{4i} \\ S_{4i+1} \\ S_{4i+2} \\ S_{4i+3} \end{bmatrix} \oplus \begin{bmatrix} K_{4i} \\ K_{4i+1} \\ K_{4i+2} \\ K_{4i+3} \end{bmatrix},$$

$$i=0,1,2,3 = \begin{bmatrix} 02 \\ 01 \\ 01 \\ 03 \end{bmatrix} \otimes S_{4i} \oplus \begin{bmatrix} 03 \\ 02 \\ 01 \\ 01 \end{bmatrix} \otimes S_{4i+1} \oplus \begin{bmatrix} 01 \\ 03 \\ 02 \\ 01 \end{bmatrix} \otimes S_{4i+2} \oplus \begin{bmatrix} 01 \\ 01 \\ 03 \\ 02 \end{bmatrix} \otimes S_{4i+3} \tag{7}$$

根据式(7),定义4张表 R :

$$\boldsymbol{R}_0[a] = \begin{bmatrix} 02 \cdot S(a) \\ 01 \cdot S(a) \\ 01 \cdot S(a) \\ 03 \cdot S(a) \end{bmatrix}, \boldsymbol{R}_1[a] = \begin{bmatrix} 03 \cdot S(a) \\ 02 \cdot S(a) \\ 01 \cdot S(a) \\ 01 \cdot S(a) \end{bmatrix}$$

$$\boldsymbol{R}_2[a] = \begin{bmatrix} 01 \cdot S(a) \\ 03 \cdot S(a) \\ 02 \cdot S(a) \\ 01 \cdot S(a) \end{bmatrix}, \boldsymbol{R}_3[a] = \begin{bmatrix} 01 \cdot S(a) \\ 01 \cdot S(a) \\ 03 \cdot S(a) \\ 02 \cdot S(a) \end{bmatrix}$$

那么利用这些表,轮函数可写成:

$$\begin{bmatrix} y_{4i} \\ y_{4i+1} \\ y_{4i+2} \\ y_{4i+3} \end{bmatrix} = \boldsymbol{R}_0[S_{4i}] \oplus \boldsymbol{R}_1[S_{4i+1}] \oplus \boldsymbol{R}_2[S_{4i+2}] \oplus \boldsymbol{R}_3[S_{4i+3}] \oplus \begin{bmatrix} K_{4i} \\ K_{4i+1} \\ K_{4i+2} \\ K_{4i+3} \end{bmatrix} \tag{8}$$

在加解密过程中均使用了4张256个4 Byte条目的表,表占用空间为8 192 Byte。

方法三是对方法二的改进[9]。利用表复用技术,使用3张256 Byte的表来实现方法二中的4张256个4 Byte条目的表,即 $S(x)$, $2S(x)$ 和 $3S(x)$;同理,解密时使用了5张大小为256 Byte的表,即 $S^{-1}(x)$, $9S^{-1}(x)$, $11S^{-1}(x)$, $13S^{-1}(x)$ 和 $14S^{-1}(x)$ 。则表占用空间为2 048 Byte。

3.2.1 AES加密轮函数的轻量实现

本文在方法三[9]的基础上进行改进。方法三中提出加密轮变换由四个如下形式部分组成:

$$\begin{bmatrix} y_{4i} \\ y_{4i+1} \\ y_{4i+2} \\ y_{4i+3} \end{bmatrix} = \begin{bmatrix} c_0 & c_3 & c_2 & c_1 \\ c_1 & c_0 & c_3 & c_2 \\ c_2 & c_1 & c_0 & c_3 \\ c_3 & c_2 & c_1 & c_0 \end{bmatrix} \otimes \begin{bmatrix} S_{4i} \\ S_{4i+1} \\ S_{4i+2} \\ S_{4i+3} \end{bmatrix} \oplus \begin{bmatrix} K_{4i} \\ K_{4i+1} \\ K_{4i+2} \\ K_{4i+3} \end{bmatrix} \tag{9}$$

其中, $i=0,1,2,3$ 。本文对式(9)进行数学变换,则有:

$$\begin{bmatrix} c_0 & c_3 & c_2 & c_1 \\ c_1 & c_0 & c_3 & c_2 \\ c_2 & c_1 & c_0 & c_3 \\ c_3 & c_2 & c_1 & c_0 \end{bmatrix} \otimes \begin{bmatrix} S_{4i} \\ S_{4i+1} \\ S_{4i+2} \\ S_{4i+3} \end{bmatrix} = \left(c_0 \otimes \begin{bmatrix} S_{4i} \\ S_{4i+1} \\ S_{4i+2} \\ S_{4i+3} \end{bmatrix} \right) \oplus \left(c_1 \otimes \begin{bmatrix} S_{4i+3} \\ S_{4i} \\ S_{4i+1} \\ S_{4i+2} \end{bmatrix} \right) \oplus \left(c_2 \otimes \begin{bmatrix} S_{4i+2} \\ S_{4i+3} \\ S_{4i} \\ S_{4i+1} \end{bmatrix} \right) \oplus \left(c_3 \otimes \begin{bmatrix} S_{4i+1} \\ S_{4i+2} \\ S_{4i+3} \\ S_{4i} \end{bmatrix} \right) \tag{10}$$

定义4个表 $T_0 = c_0[S(x)]$, $T_1 = c_1[S(x)]$, $T_2 = c_2[S(x)]$ 和 $T_3 = c_3[S(x)]$,加密轮函数可写成:

$$\begin{bmatrix} y_{4i} \\ y_{4i+1} \\ y_{4i+2} \\ y_{4i+3} \end{bmatrix} = T_0 \begin{bmatrix} S_{4i} \\ S_{4i+1} \\ S_{4i+2} \\ S_{4i+3} \end{bmatrix} \oplus T_1 \begin{bmatrix} S_{4i+3} \\ S_{4i} \\ S_{4i+1} \\ S_{4i+2} \end{bmatrix} \oplus T_2 \begin{bmatrix} S_{4i+2} \\ S_{4i+3} \\ S_{4i} \\ S_{4i+1} \end{bmatrix} \oplus T_3 \begin{bmatrix} S_{4i+1} \\ S_{4i+2} \\ S_{4i+3} \\ S_{4i} \end{bmatrix} \tag{11}$$

其中, $c_0 = 02$, $c_1 = c_2 = 01$, $c_3 = 03$,则有 $T_1 = c_1[S(x)] = T_2 = c_2[S(x)]$,即表 T_2 可由表 T_1 复用实现。又有 $c_3 = c_0 \oplus c_1$,则有 $T_3 = T_0 \oplus T_1$,即表 T_3 可由表 T_0 和表 T_1 异或实现。于是,在加密过程中,只需使用2张256 Byte的表(即表 T_0 和表 T_1),通过查表法和数组下标的选择,可同步实现SR、SUB和MC操作,减少实现算法占用存储空间的同时提高算法的执行效率。

3.2.2 AES解密轮函数的轻量优化

观察图1,调序后的解密过程具有与加密过程相同的实现结构,因此在解密过程中也可同步实现ISR、ISUB和IMC操作。

为减少方法三解密过程中使用的查找表数量,对IMC的系数矩阵进行数学变换得:

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \oplus \begin{bmatrix} 08 & 08 & 08 & 08 \\ 08 & 08 & 08 & 08 \\ 08 & 08 & 08 & 08 \\ 08 & 08 & 08 & 08 \end{bmatrix} \oplus \begin{bmatrix} 04 & 00 & 04 & 00 \\ 00 & 04 & 00 & 04 \\ 04 & 00 & 04 & 00 \\ 00 & 04 & 00 & 04 \end{bmatrix} \tag{12}$$

则解密轮函数输出表示为:

表1　各种轻量化实现方法的实验结果

实验方法	使用表数量		表总占用	算法执行效率/(Mb·s^{-1})	
	加密过程	解密过程	内存/Byte	加密效率	解密效率
方法一	1	1	512	78.86	62.53
方法二	4	4	8 192	875.06	872.83
方法三	3	5	2 048	873.12	872.14
本文方法	2	4	1 536	872.69	872.06

$$\begin{bmatrix} y'_{4i} \\ y'_{4i+1} \\ y'_{4i+2} \\ y'_{4i+3} \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \otimes \begin{bmatrix} S'_{4i} \\ S'_{4i+1} \\ S'_{4i+2} \\ S'_{4i+3} \end{bmatrix} \oplus \begin{bmatrix} IK_{4i} \\ IK_{4i+1} \\ IK_{4i+2} \\ IK_{4i+3} \end{bmatrix} =$$

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \otimes \begin{bmatrix} S'_{4i} \\ S'_{4i+1} \\ S'_{4i+2} \\ S'_{4i+3} \end{bmatrix} \oplus$$

$$\begin{bmatrix} 08 & 08 & 08 & 08 \\ 08 & 08 & 08 & 08 \\ 08 & 08 & 08 & 08 \\ 08 & 08 & 08 & 08 \end{bmatrix} \otimes \begin{bmatrix} S'_{4i} \\ S'_{4i+1} \\ S'_{4i+2} \\ S'_{4i+3} \end{bmatrix} \oplus$$

$$\begin{bmatrix} 04 & 00 & 04 & 00 \\ 00 & 04 & 00 & 04 \\ 04 & 00 & 04 & 00 \\ 00 & 04 & 00 & 04 \end{bmatrix} \otimes \begin{bmatrix} S'_{4i} \\ S'_{4i+1} \\ S'_{4i+2} \\ S'_{4i+3} \end{bmatrix} \oplus \begin{bmatrix} IK_{4i} \\ IK_{4i+1} \\ IK_{4i+2} \\ IK_{4i+3} \end{bmatrix} \quad (13)$$

由式(13)可定义4张表：

$T_4 = 02[S^{-1}(x)]$；$T_5 = S^{-1}(x)$

$T_6 = 08[S^{-1}(x)]$；$T_7 = 04[S^{-1}(x)]$

利用这些表，解密轮函数可写成：

$$\begin{bmatrix} y'_{4i} \\ y'_{4i+1} \\ y'_{4i+2} \\ y'_{4i+3} \end{bmatrix} = T_2 \begin{bmatrix} S'_{4i} \\ S'_{4i+1} \\ S'_{4i+2} \\ S'_{4i+3} \end{bmatrix} \oplus T_2 \begin{bmatrix} S'_{4i+1} \\ S'_{4i+2} \\ S'_{4i+3} \\ S'_{4i} \end{bmatrix} \oplus T_3 \begin{bmatrix} S'_{4i+2} \\ S'_{4i+3} \\ S'_{4i} \\ S'_{4i+1} \end{bmatrix} \oplus T_3 \begin{bmatrix} S'_{4i+3} \\ S'_{4i} \\ S'_{4i+1} \\ S'_{4i+2} \end{bmatrix} \oplus T_4 \begin{bmatrix} S'_{4i} \\ S'_{4i+1} \\ S'_{4i+2} \\ S'_{4i+3} \end{bmatrix} \oplus$$

$$T_4 \begin{bmatrix} S'_{4i+1} \\ S'_{4i+2} \\ S'_{4i+3} \\ S'_{4i} \end{bmatrix} \oplus T_4 \begin{bmatrix} S'_{4i+2} \\ S'_{4i+3} \\ S'_{4i} \\ S'_{4i+1} \end{bmatrix} \oplus T_4 \begin{bmatrix} S'_{4i+3} \\ S'_{4i} \\ S'_{4i+1} \\ S'_{4i+2} \end{bmatrix} \oplus$$

$$T_5 \begin{bmatrix} S'_{4i} \\ S'_{4i+1} \\ S'_{4i+2} \\ S'_{4i+3} \end{bmatrix} \oplus T_5 \begin{bmatrix} S'_{4i+2} \\ S'_{4i+3} \\ S'_{4i} \\ S'_{4i+1} \end{bmatrix} \oplus \begin{bmatrix} IK_{4i} \\ IK_{4i+1} \\ IK_{4i+2} \\ IK_{4i+3} \end{bmatrix} \quad (14)$$

则在解密过程中可使用该4张256 Byte的表，将复杂且耗时的ISUB和IMC操作直接通过查表来实现，从而减小内存，并极大地提高解密效率。

综上，本文的轻量实现方法总共使用了6张256 Byte的表。其中，在加密过程中使用了$S[x]$和$02S[x]$两张表，在解密过程中使用了$S^{-1}[x]$、$02S^{-1}[x]$、$04S^{-1}[x]$和$08S^{-1}[x]$四张表，比方法三所需的空间减少了25%。

4　实验结果分析

在Intel P4 2.93 GHz，1.49 GB RAM PC机上，以Windows XP系统为平台，使用Visual C++6.0实现几种AES加解密优化算法，进行效率比较测试。

实验时，由于加密一个分组的时间相当短，本文采用一次加密多个分组进行加密时间的测试。并且将轮密钥的生成视为预运算，不加入实验测试。测试时，每次均采用随机加密10 MB数据，每种方法各测试5次，结果取其平均值。实验结果如表1所示。

对表中的实验数据进行分析，方法一采用的查找表只实现了S盒的功能，并不涉及列混淆操作的优化，在有限域GF(2^8)中，乘法运算比查表操作、循环移位、异或运算耗时多，其平均加解密效率最低，仅为本文方法效率的9.04%和7.17%。

方法二中，由于加解密的每一轮轮函数只需经过4×4次查表与12次异或操作即可实现，所以其加解密效率最高；但其每一张表均是256个4 Byte条目的表，因而其表总占用内存最大。

与其他方法相比，由于本文方法的轮函数中少了一些中间转换步骤，其加解密效率比原实现方案有了大幅提高；又因为改进后的算法将行移位、字节替换和列混淆整合同步实现，也减少了一些中间的存储状态而节省空间的使用。与方法二相比，算法所用的存储空间节省了81.25%，但其加解密效率仅减少了2.71‰和0.88‰；与方法三相比，算法的执行效率相当，但是所用的存储空间节省了25%。综合考虑算法占用存储空间和算法执行效率，本文的AES软件实现方法具有一定的优势。

5　总结

本文主要研究AES的软件实现方法。在分析AES算法设计原理以及现有的优化实现方案的基础上，对加解密轮函数进行合并优化，设计了一种较轻量且高效的实现方法。经比较实验，其结果表明本文算法对存储空间要求低且执行效率高，这在某些资源受限的环境中，具有较好的实用意义。

参考文献:

[1] Kaminsky A, Kurdziel M, Radziszowski S.An overview of cryptanalysis research for the advanced encryption standard[C]//Proceedings-IEEE Military Communications Conference MILCOM,2010:1310-1316.

[2] Ali L, Aris I, Hossain F S, et al.Design of an ultra high speed AES processor for next generation IT security[J]. Computers and Electrical Engineering, 2011, 37(6): 1160-1170.

[3] Rizvi S A M, Hussian S Z, Wadhwa N.Performance analysis of AES and towfish encryption schemes[C]//Proceedings-2011 International Conference on Communication Systems and Network Technologies,2011:76-79.

[4] Babu T R, Murthy K V V S, Sunil G.Implementation of AES algorithm on ARM[C]//International Conference and Workshop on Emerging Trends in Technology 2011, 2011: 1211-1213.

[5] Osvik D A, Bos J W, Stefan D, et al.Fast software AES encryption[J].Fast Software Encryption-17th International Workshop,2010:75-93.

[6] Ahmed W, Mahmood H, Siddique U.The efficient implementation of S8 AES algorithm[C]//Proceedings of the World Congress on Engineering,2011:1215-1219.

[7] Barnes A, Fernando R, Mettananda K, et al.Improving the throughput of the AES algorithm with multicore processors[C]//IEEE 7th International Conference on Industrial and Information System,ICIIS 2012,2012.

[8] Dacemen,Rijmen V.高级加密标准(AES)算法——Rijndael的设计[M].北京:清华大学出版社,2003:31-64,223-230.

[9] 崔国华,唐国富,洪帆.AES算法的实现研究[J].计算机应用研究,2004(8):99-101.

[10] 吴文玲,冯登国,张文涛.分组密码的设计与分析[M].2版.北京:清华大学出版社,2009:14-22.

[11] Dacemen J, Rijmen V.The block cipher rijndael smart card research and applications[C]//Lecture Notes in Computer Science.Berlin:Springer-Verlag,1997,1267:149-165.

[12] Lin Bing, Xia Kewei, Liang Wenli.Reconfigurable implementation of AES algorithm IP core based on pipeline structure[J].Journal of Southeast University(English Edition),2010,26(1):21-25.

[13] Lee R B, Chen Yuyuan.Processor accelerator for AES[C]// Proceedings of the 2010 IEEE 8th Symposium on Application Specific Processors,SASP'10,2010:16-21.

[14] Li Nan.Research of database encryption based on fast AES algorithm implementation[C]//Advanced in Computer Science, Environment, and Education Communications in Computer and Information Science,2011:31-35.

[15] Paar C.The Advanced Encryption Standard(AES)[C]// Cryptographic Hardware and Embedded System-CHES 2011 Computer Science,2011:95-107.

(上接71页)

参考文献:

[1] Mambo M, Usuda K, Okamoo E.Proxy signatures:delegation of the power to sign messages[J].IEICE Transactions on Fundamentals,1996,E79-A:1338-1354.

[2] 李继国,曹珍富,李建中,等.代理签名的现状与发展[J].通信学报,2003(10):114-124.

[3] Lee B, Kim H, Kim K.Strong proxy signature and its applications[C]//Proceedings of SCIS2001, Oiso, Japan, 2001:603-608.

[4] Shamir A.Identity based cryptosystems and signature schemes[C]//Blakely G R, Chaum D.Proceedings of Crypto 1984,California,USA,1984,196:47-53.

[5] 李继国,曹珍富.一个改进的门限代理签名方案[J].计算机研究与发展,2002,39(11):1513-1518.

[6] Huang H F, Chang C C.A novel efficient(t,n) threshold proxy signature scheme[J].Information Sciences,2006,176: 1338-1349.

[7] 于静,李志慧.一种有代理的门限签名方案[J].计算机工程与应用,2009,45(13):127-129.

[8] Al-Riyami S, Paterson K.Certificateless public key cryptography[C]//Laih C S.Proceedings of the Asiacrypt 2003, Taipei, Taiwan, China, 2003, 2894:312-323.

[9] Choi K, Lee D.Certificateless proxy signature scheme[C]// Proceedings of the 3rd International Conference on Multimedia, Information Technology and Its Applications-MITA 2007, Manila, Philippines, 2007:437-440.

[10] Zhang Lei, Zhang Futai, Wu Qianhong.Delegation of signing rights using certificateless proxy signatures[J]. Information Sciences,2012,184(1):298-309.

[11] 于静,李志慧.无证书的代理门限签名方案[J].计算机工程与应用,2013,49(12):74-76.

[12] 伊丽江,白国强,肖国镇.代理多重签名:一类新的代理签名方案[J].电子学报,2001,29(4):569-570.

[13] 王晓明,陈火炎,付方伟.前向安全的代理签名方案[J].通信学报,2005,26(11):38-42.

[14] Zhang Mingwu, Yang Bo, Takagi T.Group-oriented setting's multisigncryption scheme with threshold designcryption[J]. Information Sciences,2011,181:4041-4050.