第 36 卷 第 6 期 计 算 机 工 程 2010 年 3 月
Vol.36 No.6 Computer Engineering March 2010
·安全技术· 文章编号：1000—3428(2010)06—0128—03 文献标识码：A 中图分类号：TN918.1

针对低轮 PRESENT 的代数攻击

卜 凡，金晨辉
(解放军信息工程大学电子技术学院，郑州 450004)

摘 要：基于 MiniSAT 2.0 软件，研究对低轮 PRESENT 的代数攻击问题。提出将 S 盒表示为单项式个数较少的无冗余等效方程组的方法，将 PRESENT 的 S 盒表示为由 14 个单项式个数均≤6 的 8 元布尔方程构成的等效方程组，并基于不同的已知明文量，利用 MiniSAT 软件对 PRESENT 进行代数攻击实验，获得了较好的攻击效果。实验表明，在已知明文条件下可以在 121 h 内求出 80 bit 密钥的 5 轮 PRESENT 的全部密钥比特，在选择明文条件下可以在 203 h 内求出 6 轮 PRESENT 的全部密钥比特。

关键词：代数攻击；MiniSAT 软件；等效方程组；无冗余方程组；PRESENT 算法

Algebraic Attack on Low-round PRESENT

BU Fan, JIN Chen-hui
(Electronic Technology Institute, PLA Information Engineering University, Zhengzhou 450004)

【Abstract】This paper studies the methods of algebraic attack on PRESENT with the help of MiniSAT 2.0. It also analyzes the S-box of PRESENT and finds fourteen equivalent implicit functions of the S-box, which includes no redundancy functions and the number of normal expressions of each function is no more than six. Using different number of plaintexts, it does algebraic attack on low-round PRESENT, and the result is the best one at present. Result shows that if knowing plaintexts, it can recover the keys of five-round PRESENT using 80 bit keys in less than 121 h and if knowing the selective plaintexts, it also recovers the keys of six-round PRESENT in less than 203 h.

【Key words】algebraic attack; MiniSAT software; equivalent system of equations; non-redundant equation; PRESENT algorithm

1 概述

针对密码算法实现时会受到现有资源的限制(如 RFID 电子标签、传感器网络等)，文献[1]提出了一个具有 31 轮迭代的超轻量级分组密码算法 PRESENT。由于仅使用 4 进 4 出的 S 盒、比特移位和模 2 加运算，因此 PRESENT 具有很好的硬件实现性能。目前，对于 PRESENT 算法，除设计者所做的安全性分析之外，文献[2]还提出了差分攻击。利用文献[2]的方法，使用 2^{64} 个选择明文时，成功攻击 16 圈 PRESENT 算法的计算复杂度为 2^{64}。

抗代数攻击能力是评价密码算法安全性的一个重要方法。代数攻击是将密码算法表示为由明文和密文为已知参数，以密钥和中间变量为未知变量的一个代数方程组，并通过求解该方程组的方法，达到求解密钥的目的。目前，求解代数方程组的方法主要有线性化方法(包括直接线性化方法 XL[3]、扩展线性化方法 XSL[4]等)、求 Gröbner 基的方法(F4[5]算法、F5[6-7]算法等)和转化为可满足性问题(SAT 问题)并利用 SAT Solver 软件求解该 SAT 问题从而求解方程组的方法[8]。

在 PRESENT 算法的代数攻击方面，文献[1]研究了基于解超定代数方程组的 XL 方法和求 Gröbner 基方法，对 PRESENT 进行代数攻击的能力。基于 PRESENT 的 S 盒可以表示为 21 个 8 元二次方程组的事实，PRESENT 的加密算法和密钥编排算法共使用了 $n=32\times16=512$ 个 S 盒，设计者[1]证明了 PRESENT 算法可用由 4 216 个未知变量和 11 067 个二次方程构成的代数方程组表示，从而说明了利用现有的线性化方法不能对 PRESENT 进行有效的代数攻击。文献[1]还利用 Magma 软件中的 F4 算法对 2 轮 PRESENT 算法进行实验性攻击，但未能在合理的时间内实现对密钥的还原。

本文基于 SAT Solver 软件求解代数方程组方法，研究了对低轮 PRESENT 的代数攻击问题。

2 PRESENT 中 S 盒的少项等效方程组的构造

在将 S 盒表示为等效的非线性方程组时，如果每个方程的单项式个数都较少，则基于 MiniSAT 软件求解密码算法的等效方程组的代数攻击将更加有效。因此，寻找 S 盒的单项式个数较少的等效方程组表示，对于提高代数攻击的效率具有实际意义。

先介绍布尔函数的代数正规型表示的定义。

定义 1 设 f 是一个 n 元布尔函数，则称 $f(x)=\bigoplus_{b\in\{0,1\}^n}c_b x^{(b)}$ 为 f 的代数正规型表示，并称 $\sum_{b\in\{0,1\}^n}c_b$ 为 f 的项数。其中，$\forall x=(x_0,x_1,\cdots,x_{n-1}),b=(b_0,b_1,\cdots,b_{n-1})\in\{0,1\}^n$，有 $x^{(b)}=\prod_{i=0}^{n-1}x_i^{b_i}$，且 $x_i^{b_i}$ 是 x_i 的 b_i 次方。

定义 2 设 $S:\{0,1\}^n\to\{0,1\}^m$，$f_1,f_2,\cdots,f_N:\{0,1\}^{n+m}\to\{0,1\}$。如果 $\forall z=(x,y)\in\{0,1\}^n\times\{0,1\}^m$，当 $S(x)=y$ 时，对 $1\le i\le N$ 都有 $f_i(z)=0$，则称方程组 $f_i(z)=0,\ 1\le i\le N$ 为 S 的一个必要方程组。如果 $\forall z=(x,y)\in\{0,1\}^n\times\{0,1\}^m$，$S(x)=y$ 等价于对 $1\le i\le N$ 都有 $f_i(z)=0$，则称该方程组为 S 的等效方程组。

必要方程组未必是等效方程组，利用 S 盒的必要方程组只能构造出密码算法的必要方程组。只有该必要方程组的解

基金项目：河南省杰出青年科学基金资助项目(0312001800)
作者简介：卜 凡(1982－)，男，硕士研究生，主研方向：密码学；金晨辉，教授
收稿日期：2009-09-03 **E-mail：**bufan1982@yahoo.cn

唯一时，才是密码算法的密钥。一般而言，如果没有细致的理论分析，不能保证由必要方程组能够唯一求出密码算法的原始密钥。但是，如果利用 S 盒的等效方程组表示，则可以简化这种分析。

下面研究 S 盒的单项式个数都较少的等效方程组的构造方法。

设 $S:\{0,1\}^n\to\{0,1\}^m$，则 $S_i(x)=y_i,\ \ 1\le i\le m$ 就是 S 的一个等效方程组，故等效方程组是存在的。显然有

定理 设 $S:\{0,1\}^n\to\{0,1\}^m$，则 S 的必要方程组 $f_i(z)=0,\ \ 1\le i\le N$ 是 S 的一个等效方程组的充要条件是该方程组的解数为 2^n。

定理解决了如何判断一个必要方程组是否为等效方程组的问题。因此，一个等效方程组可通过对必要方程组添加一些必要方程的方法获得。$S:\{0,1\}^n\to\{0,1\}^m$ 的必要方程可利用 2 种方法得到。

(1)通过求解以系数 $c_a,a\in\{0,1\}^{n+m}$ 为未知变量的线性方程组 $f_c(x,S(x))=0,\ \ x\in\{0,1\}^n$ 而获得。此时，线性方程组 $f_c(x,S(x))=0,\ \ x\in\{0,1\}^n$ 的每个解 $c_a,a\in\{0,1\}^{n+m}$ 都对应一个必要方程组。该方法的缺点是求解具有一些特殊要求的系数向量比较困难。

(2)通过检测满足特殊要求的布尔方程是否是必要方程的方法，构造满足特定要求的必要方程组。

本文将采取方法(2)构造 PRESENT 算法 S 盒的单项式个数均≤6 的必要方程，并通过逐步向已构造出的必要方程组中添加新的必要方程的方法，构造出 PRESENT 算法 S 盒的单项式个数均≤6 的等效方程组，最后采取去除等效方程组中的冗余方程的方法，构造出 PRESENT 算法 S 盒的单项式个数均≤6 的无冗余的等效方程组。采取先检测单项式个数较小的 $m+n$ 元布尔方程，再检测单项式个数较大的 $m+n$ 元布尔方程的方法搜索必要方程组。由于线性方程组 $f_c(x,S(x))=0,\ \ x\in\{0,1\}^n$ 由 2^n 个方程组成，且每个方程共有 2^{m+n} 个系数，因此存储该方程需要 2^{m+2n} 个存储单元，检测一个重量为 t 的向量是否是其解的最大计算复杂性为 2^nt。

下面给出线性布尔方程组的一种压缩存储方法，通过将线性布尔方程组表示为线性 32 位字方程组，该方法可将检测算法的存储复杂性和计算复杂性降低 32 倍。设 $\bigoplus_{j=1}^{t}a_{i,j}x_j=0,\ \ 1\le i\le 2^n$ 是一个具有 t 个未知变量的布尔方程组。对 $0\le i\le 2^{n-5}$，令 $A_{i,j}=\sum_{t=32i}^{32i+31}a_{t,j}2^t$，则 $(x_1,x_2,\cdots,x_t)$ 是方程组 $\bigoplus_{j=1}^{t}a_{i,j}x_j=0,\ \ 1\le i\le 2^n$ 的解的充要条件是对 $0\le i\le 2^{n-5}$，均有 $\bigoplus_{j=1}^{t}A_{i,j}x_j=0$。由于存储矩阵 $(a_{i,j})_{2^n\times t}$ 需要 2^nt 个存储单元，但存储矩阵 $(a_{i,j})_{2^{n-5}\times t}$ 只需要 $2^{n-5}t$ 个存储单元，因此存储量降低了 32 倍；此外，由于方程的数量减少了 32 倍，因此检测 $(x_1,x_2,\cdots,x_t)$ 的解的最大计算复杂性也降低了 32 倍。

上述存储压缩方法还可用于检测一个方程组是否是 S 盒的等效方程组。当 2^{m+n} 较大时，可以采取高斯消元法，求出 2^{m+n} 变元线性方程组的解数；当 2^{m+n} 较小时，可以采取穷举 $\{0,1\}^{m+n}$ 中所有元的方法，检测该 2^{m+n} 变元线性方程组的解数是否为 2^n，从而完成一个方程组是否是 S 盒的等效方程组的检测。

在检测 S 盒的一个由 n 个方程构成的等效方程组是否有冗余方程时，可以采取逐一测试每个方程是否是冗余方程的方法，检测该方程组是否有冗余方程，并在该方程是冗余方程时，去掉该冗余方程。此时，可采取检测去掉该方程后得到的方程组是否仍是等效方程组的方法，检测该方程是否是冗余方程。

下面利用上述原理和方法，构造 PRESENT 算法 S 盒的单项式个数均≤6 的无冗余的等效方程组。

PRESENT 采用 SP 网络设计，其分组长度为 64 bit、密钥长度为 80 bit 或 128 bit、迭代 31 圈，在 31 圈迭代执行完毕后，输出再与最后一个圈密钥逐位模 2 加。圈函数由圈密钥加、代替变换和逐比特移位变换(详见文献[1])组成。其中，圈密钥加是将圈函数的输入与圈密钥逐比特模 2 加，代替变换由 16 个 4 进 4 出 S 盒并置构成，S 盒均为{12,5,6,11,9,0,10,13, 3,14,15, 8,4,7, 1,2}。圈密钥通过初始密钥通过以下方式得到。对于 80 bit 密钥的情形，设 $K=k_{79}k_{78}\cdots k_0$ 为初始密钥，对于 j 从 1~32，则选取 $K_j=k_{79}k_{78}\cdots k_{16}$ 作为第 j 圈的圈密钥，然后利用下述步骤对 K 进行刷新：

(1)执行 $[k_{79}k_{78}\cdots k_0]=[k_{18}k_{17}\cdots k_{20}k_{19}]$；

(2)执行 $[k_{79}k_{78}k_{77}k_{76}]=S[k_{79}k_{78}k_{77}k_{76}]$；

(3)执行 $[k_{19}k_{18}k_{17}k_{16}k_{15}]=[k_{19}k_{18}k_{17}k_{16}k_{15}]\oplus$圈序号。

在 PRESENT 算法的圈函数中，圈密钥加和逐比特移位变换都是线性变换，因此很容易得到这些变换的以输入和输出比特位为未知量的线性方程。设其 S 盒的输入和输出分别为 $\sum_{i=1}^{4}x_i2^{i-1}$ 和 $\sum_{i=1}^{4}y_i2^{i-1}$，则利用本文给出的方法，找出了该 S 盒的每个布尔方程最多含 6 个单项式的等效方程组，再去除其中的冗余方程后，得到了由 14 个布尔方程构成的等效方程组，该方程组具体为

$$\begin{cases}
x_2x_3\oplus y_2y_4\oplus x_3\oplus x_4\oplus y_3\oplus 1=0\\
x_3x_4\oplus y_1y_4\oplus y_3y_4\oplus x_1\oplus y_2\oplus 1=0\\
x_3y_2y_3\oplus x_1y_2\oplus x_4y_2\oplus y_1y_2=0\\
x_1x_2x_3y_2\oplus x_1x_2y_4\oplus x_1x_2=0\\
x_1x_2\oplus x_2x_4\oplus x_2y_1=0\\
x_1y_1y_3y_4\oplus x_2y_2y_3\oplus x_2y_3=0\\
x_1y_1y_2y_3\oplus x_2x_3y_2\oplus x_3y_2=0\\
x_1x_2x_4\oplus x_1y_2y_3\oplus x_1x_4=0\\
x_1y_2y_3y_4\oplus x_2y_2y_4\oplus y_2y_4=0\\
x_2y_2y_4\oplus x_3y_4\oplus y_1y_4=0\\
x_2x_4y_1\oplus x_3x_4y_1\oplus y_1y_2y_3=0\\
x_1y_2y_3\oplus x_2y_2y_3\oplus y_2y_3=0\\
y_2y_3y_4\oplus x_1y_3\oplus y_1y_3=0\\
y_1y_2y_4\oplus x_1y_1\oplus y_1y_3=0
\end{cases}$$

3 对 PRESENT 算法基于 MiniSAT 的代数攻击

利用 S 盒的等效方程组表示，对低轮 PRESENT 进行了代数攻击。主要方法如下：

(1)建立以 PRESENT 算法中 S 变换的输入和输出比特位，以及圈密钥加的输出比特为未知量的代数方程组，其中，S 盒利用上述等效方程组表示，圈密钥加作为线性变换，可直接形成以输入输出的各比特位为未知量的线性方程。由于逐比特移位变换的输入和输出分别是本轮 S 变换的输出和下轮圈密钥加的输入，因此可不将圈密钥加的输入作为中间变量，从而减少密码算法的等效方程组表示中未知变量的个数。

(2)将方程组的求解问题转化为 SAT 问题，进而利用 MiniSAT 软件进行求解。转化过程中利用文献[8]中的方法进

行：首先将每个单项式替换为一个新未知量，然后对该替换对应的新方程进行 SAT 语句转换，最后，对用新变量替换原单项式后得到的线性方程组进行 SAT 语句转换。

说明：在利用已知 t 个明密对进行代数攻击时，只须将涉及到明密文的那些代数方程转化为 SAT 问题，然后再与原有的语句集共同进行 MiniSAT 软件求解，这样可以大大减少 SAT 语句转换的工作量。

上述代数攻击方法利用 MiniSAT 2.0 软件，对 80 bit 密钥的低轮 PRESENT 算法进行了大量的攻击试验。试验设备采用主频为 2.2 GHz 的 AMD Athlon 64×2 Dual CPU、内存为 1 GB，每种情况均做了 4 例实验。实验情况如下：

(1)对于 3 轮 PRESENT，利用 1 个已知明密对求出全部密钥比特的平均时间为 3 593.6 s，不到 1 h。

(2)对于 4 轮 PRESENT，利用 2 个已知明密对求出全部密钥比特的平均时间为 10 800 s，约 3 h；利用 3 个已知明密对求出全部密钥比特的计算时间为 1 440.01 s，约 23 min；利用 4 个已知明密对求出全部密钥比特的计算时间约为 6 min。

(3)对于 5 轮 PRESENT，利用 4 个已知明密对求出全部密钥比特的平均计算时间约为 120 h 14 min；利用 5 个已知明密对求出全部密钥比特的时间约为 183 h。

(4)对于 6 轮 PRESENT，利用 4 个(任意 2 个仅有 1 bit 不同)已知明密对，求出全部密钥比特的平均计算时间约为 202 h 23 min。

说明：从试验结果可以看出，在密钥求解时，由于增加已知明密文的个数，将会增加方程个数与密钥变量个数的差值，因此有利于减低 MiniSAT 的运行时间。但是，增加明密文的个数相当于变相增加了中间变量的个数，因而增加了 SAT 语句集中的语句数，从而使 MiniSAT 的求解时间变长。因此为减少 MiniSAT 的求解时间，不能无限制增加已知明密文的对数。

4 结束语

本文基于 SAT Solver 软件求解代数方程组方法，研究了对低轮 PRESENT 的代数攻击问题。获得了目前最好的攻击效果。研究结果表明，在已知明文条件下可以求出 80 bit 密钥的 5 轮 PRESENT 的全部密钥比特，在选择明文条件下可以求出 80 bit 密钥的 6 轮 PRESENT 的全部密钥比特。对其他具有类似结构的算法，也可采取本文方法进行类似攻击。

参考文献

[1] Bogdanov A, Knudsen L R, Leander G, et al. PRESENT: An Ultra-lightweight Block Cipher[EB/OL]. (2007-04-03). http://www.ist-ubisecsens.org/publications/present_ches2007.pdf.

[2] Wang Meiqin. Differential Cryptanalysis of PRESENT[EB/OL]. (2007-04-08). http://eprint.iacr.org/2007/408.

[3] Courtois N T, Klimov A, Patarin J. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations[EB/OL]. (2000-08-07). http://www.iacr.org/archive/eurocrypt2000/1807/18070398-new.pdf.

[4] Kipnis A, Shamir A. Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization[C]//Proc. of Advances in Cryptology-Crypto'99. [S. l.]: Springer-Verlag, 1999: 19-30.

[5] Faugere J C. A New Efficient Algorithm for Computing Gröbner Basis(F4)[EB/OL]. (1999-05-12). http://www-spaces.lip6.fr/@papers/F99a.pdf.

[6] Faugere J C. A New Efficient Algorithm for Computing Gröbner Basis Without Reduction to Zero(F5)[EB/OL]. (2002-04-05). http://www-spaces.lip6.fr/@papers/F02a.pdf.

[7] Seger A J M. Algebraic Attacks from a Gröbner Basis Perspectives[EB/OL]. (2004-11-04). http://www.win.tue.nl/~henkvt/images/ReportSegers.

[8] Bard G V, Courtois N T, Gregory C J. Efficient Methods for Conversion and Solution of Sparse Systems of Low-degree Multivariate Polynomials over GF(2) via SAT-Solvers[EB/OL]. (2007-02-04). http://eprint.iacr.org/2007/024.

编辑　任吉慧

(上接第 127 页)

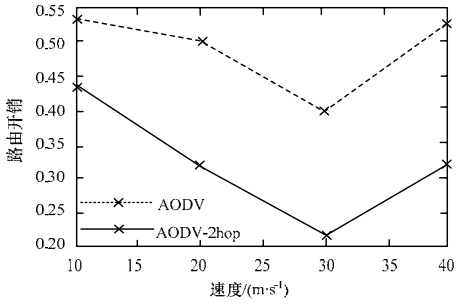

图 6　路由开销

5 结束语

AODV 是 Ad Hoc 中最理想的算法，更能适应网络拓扑动态变化的环境，是目前的研究热点，但是研究主要集中在路由发现阶段，关于路由维护的研究比较少。本文在文献[3]的基础上提出了一种改进的 AODV 局部修复算法，将修复限制在 2 跳的范围内，修复对象不仅包括下一跳节点、下 2 跳节点，还包括拥有“足够新”路由的节点。仿真结果显示，与文献[3]提出的算法相比，在提高包投递率的同时，改进算法路由开销明显减少，接近原路由算法的一半。下一步工作是减少端到端的延迟，使路由的跳数最短，并根据终端带宽、能量等因素自适应网络的动态变化，使修复更加灵活。

参考文献

[1] 洪锡军. 无线自组网路由协议研究[J]. 计算机工程, 2005, 31(8): 105-107.

[2] Broch J, Maltz D, Johnson D, et al. A Performance Comparison of Multi-hop Wireless Ad Hoc Network Routing Protocols[C]// Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking. New York, USA: IEEE Press, 1998: 85-97.

[3] Perkins C, Royer E B, Das S. Ad Hoc on Demand Distance Vector (AODV) Routing[S]. RFC 3561, 2003.

[4] 肖百龙, 郭　伟, 刘　军, 等. 移动自组织路由局部修复算法的研究[J]. 计算机研究与发展, 2007, 44(8): 1383-1389.

[5] Liu Genping, Wong K J, Lee B S, et al. PATCH: A Novel Local Recovery Mechanism for Mobile Ad-hoc Networks[C]//Proc. of IEEE VTC'03. Ohio, USA: IEEE Press, 2003: 2995-2999.

编辑　张正兴