对二元一次不定方程背包方案的格攻击①

秦 颖[1], 潘 瑜[2]

[1](青海师范大学 计算机学院, 西宁 810000)
[2](江苏理工学院 计算机工程学院, 常州 213001)

摘 要: 研究分析背包密度大于 0.9408 的背包密码方案的安全性非常重要. 针对基于二元一次不定方程的难解函数的新型背包公钥密码算法, 由公钥和密文构造一个格来攻击该方案, 通过采用 NTL 库验证上述格攻击算法的效率, 从而证明了该攻击方法的有效性. 进而说明此新型背包公钥密码体制是不安全的.

关键词: 背包公钥密码; LLL 算法; 背包密度; NTL 库

Lattice Attack Based on Linear Indeterminate Equation Knapsack Cryptosystems

QIN Ying[1], PAN Yu[2]

[1](School of Computer Science, Qinghai Normal University, Xining 810000, China)
[2](School of Computer Engineering, Jangsu University of Technology, Changzhou 213001, China)

Abstract: It is very important to research and analyze one of Knapsack Cryptosystems in which the density of knapsack is greater than 0.9408. For the new knapsack public-key algorithm which is based on linear indeterminate equation in two variables, public key and ciphertext are used to construct a lattice to attack the scheme and the NTL library to verify the efficiency of the lattice attack algorithm, which finally illustrates the effectiveness of the attack method. It then proves the new knapsack public-key cryptosystem is insecure.

Key words: KPC; LLL Algorithm; knapsacks density; NTL library

1 引言

背包公钥密码体制和RSA 公钥体制一起, 被公认为是两个著名的公钥体制. 1978 年 Merkle 和 Hellman 首先提出了一个现在称为 MH 背包体制的密码体制[1], 虽然 MH 体制和 MH 的几个变形在 20 世纪 80 年代初被 Shamir[2]等人破译了, 但是, MH 公钥密码的思想和相关理论首先解释了公钥密码算法的本质, 同时背包密码体制具有快速加密和解密的优势以及背包问题的 NP 完全性, 所以背包方案尤其适用于一些内存空间、运行时间等资源受限制的场合, 在今天仍然具有研究价值.

在文献[3]中, 费向东等人设计了一种新型背包公钥密码算法——基于二元一次不定方程的难解函数的背包公钥密码算法, 并且证明了这个新型背包密码算法在受到明文恢复攻击时, 它利用二元一次整系数不定方程生成公钥序列, 相当于一次模乘运算, 提高了背包的密度; 同时在受到密钥恢复攻击的情况下, 算法的安全性规约为破解基于二元一次不定方程的难解函数F_2, 见公式(1):

$$\begin{cases} \mathrm{d}_i = 2^j u_i + v_i \quad i=1,\cdots,n \\ F_2(\mathrm{d}_i) = eu_i + fv_i \end{cases} \tag{1}$$

文中利用基于以上两种攻击的困难性, 证明了此算法的安全性.

然而借鉴文献[4]的攻击思想, 文中通过NTL 库[5], 运用不同的参数来进行实验, 并且调用 LLL 格规约基算法[6], 攻击者就可以在归约基中找到与明文背包向量相近的特殊向量, 从而恢复了明文, 因此破解了文献[3]中的密码方案.

① 收稿时间:2012-12-20;收到修改稿时间:2013-01-10

2 预备知识

2.1 背包密度

已知公钥序列 $A=(a_1,\cdots,a_n)$, n 位长二进制明文为 $(m)_2=(m_1,\cdots,m_i,\cdots,m_n)$, 被加密的密文为 $c=a_1m_1+\ldots+a_nm_n$, 其中 m_i 的取值为 0 或 1, 则背包密度为:

$$Density=n/lb(\max(a_i)),i=1,\cdots,n \quad (2)$$

在文献[7]中, Coster 等人证明了当背包密度小于 0.9408 时, 背包体制易遭低密度子集和攻击, 而当背包密度大于 1 时, 又造成了密码体制在实现过程中解密的不唯一性. 因此可以得出结论: 安全的背包公钥密码密度应该在[0.9408,1]区间内. 但在接下来的攻击过程中我们发现即使文献[3]中方案的背包密度大于 0.9408 也不一定是安全的.

2.2 格

给定 n 个线性独立的向量: $b_1,b_2,\ldots,b_n\in R^m$, 格的产生是通过这些向量 $b_1,b_2,\ldots,b_n\in R^m$ 在公式(3)中的定义:

$$L(b_1,b_2,\ldots,b_n)=\left\{\sum x_ib_i\mid x_i\in Z\right\} \quad (3)$$

我们把 $b_1,b_2,\ldots,b_n$ 称作格的一组基. 同样地, 如果我们定义 B 是 $m\times n$ 的矩阵, 其中列为 $b_1,b_2,\ldots,b_n$, 那么由 B 产生的格为:

$$L(B)=L(b_1,b_2,\ldots,b_n)=\left\{Bx\mid x\in Z^n\right\} \quad (4)$$

自从 Ajtai 里程碑式的论文[8]发表以来, 利用格上的难题来构思新的密码编码或密码分析的想法, 在密码学中得到了很多的应用. 本文就是利用格的思想来进行密码分析的.

2.3 LLL 算法

LLL 算法是 A.K.Lenstra, H.W.Lenstra 和 L.Lovasz 三个人于 1982 年在文献[6]中提出的一个最短向量问题(SVP)的近似算法. 对于 n 维的格, 它给定的近似率为 $(2/\sqrt{3})^n$. 在许多应用中, 这个算法适用于为常数的情况; 如此, 我们就可获得一个常数近似因子.

LLL 算法的应用很广, 如: 整数或有理数的多项式因式分解、发现整数关系、整数编程以及密码分析中对背包密码系统的攻击和一些特殊情况下的 RSA 的低公共指数攻击.

如果基 $B=\{b_1,\cdots,b_n\}\in R^n$, 满足公式(5)的条件, 我们就称它是一个格的 LLL 规约基:

$$\begin{cases}\forall l\le i\le n,j<i.\mid u_{i,j}\mid\le 1/2\\ \forall l\le i<n.\delta\parallel\tilde{b}_i\parallel^2\le\parallel u_{i+1},\tilde{b}_i+\tilde{b}_{i+1}\parallel^2\end{cases} \quad (5)$$

格规约基是格理论中的一个重要研究内容, 许多格上面的问题都可以通过格规约来近似求解, 对很多密码算法的分析最后都可以等价成一个格规约问题, 因此它非常具有理论与实用价值. 本文就是通过构造格, 利用 LLL 规约基来实现对文献[3]中所提出算法的攻击.

3 基于二元一次不定方程的改进背包密码算法

3.1 密钥生成算法

Step1: 选取超递增初始序列 $B=(b_1,\ldots,b_2,\ldots,b_n)$, $b_1=\max(b_i),i=1,\cdots,n$;

Step2: 选互素的正整数 w 和 m, 满足 $m>\sum_{i=1}^{n}b_i$;

Step3: 运用乘数 w, 对 B 序列项 b_i 进行模乘运算, 将 B 转换 D 序列:

$d_i=wb_i(\mathrm{mod}\,m)$, $i=1,\cdots,n$;. $D=(d_1,\ldots,d_n)$

设|m| = h, 则 $\max(|d_i|)=h,i=1,\ldots,n$. |m|表示 m 的比特位长度;

Step4: 对 D 序列项 d_1 进行对称分组 d_1 的长度以 h= |m|位计(如高位不足, 以 0 计), 左起第 $j=h/2$ 位分为左、右长度相等两部分, 形成 u_i 和 v_i, $|u_i|=|v_i|$, 其中 $d_i=u_i*2^j+v_i$;

Step5: 选互素的正整数 e 和 f, 满足 $e>\sum_{i=1}^{j}v_i$, $f>\sum_{i=1}^{j}u_i$, $a_i=eu_i+fv_i,i=1,\ldots,n$.

输出公钥: $pk=\{A=(a_1,\cdots,a_n)\}$;

私钥: $sk=\{e,f,w^{-1},m\}$ 超递增初始序列 B 作为固定值嵌入算法中.

3.2 加密算法

给定公钥 pk 和二进制明文:

$(x)_2=(x_1\cdots x_i\cdots x_n)$, $x_i=0$ 或 1, 可以计算输出密文: $c=(a_1x_1+\cdots+a_nx_n)$.

3.3 解密算法

Step1: 利用私钥 sk 解不定方程 $eu+fv=c$, 方法为, 运用扩展欧几里德算法, 计算 X 和 Y, 使得 $eu+fv=1$; 易见 XC 和 YC 是方程的一组解, 以此计算出该不定方程的正整数解 u 和 v;

Step2: 计算 $d=u*2^j+v$;

Step3: 计算 $z=w^{-1}*d(\mathrm{mod}\,m)$;

Step4: 运用超递增初始序列 B 计算得明文 x.

4 攻击方法

4.1 基本思想

上述基于二元一次不定方程的改进背包密码算法, 称作是可证明的启发式方法, 文中说明了无论攻击者从公钥序列出发, 还是从初始序列出发, 此密码体制都具有可证明的安全性.

同时文献[3]中给出的一个例子中的背包密度是0.9771, 符合安全的条件: 背包公钥密码的密度应尽量接近 1 才能保证安全性.

然而, 通过我们设计的格攻击算法进行不同参数的计算实验, 由计算结果可以得出背包密度即使是接近 1 的情况下(0.9990243902), 这个改进的背包密码算法也能被攻破, 从而说明了此方案是不安全的.

4.2 格攻击算法设计

1) 给定公钥 $A=(a_1,\cdots,a_n)$ 和密文 $c=(a_1x_1+\cdots+a_nx_n)$, 构造的格 L 如公式(6)所示:

$$L=\begin{pmatrix} -c^2 & -1 & -1 & \cdots & -1 & 1 \\ a_1*c & 2 & 0 & \cdots & 0 & 0 \\ a_2*c & 0 & 2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_n*c & 0 & 0 & \cdots & 2 & 0 \end{pmatrix} \tag{6}$$

2) 调用 LLL 算法, 得到 L 的规约基 B.

3) 接着在 B 中寻找形如 $B_i=(0,-1/1,-1/1,\cdots,-1/1,1)$ 的向量(即以 0 开头, 1 结尾, 中间可以是 1 或 -1 的向量), 其中 $B_i=(B_{i,0},B_{i,1},\cdots,B_{i,n},B_{i,n+1})$, 若存在这样的向量, 即可知道明文 $m=(m_1,\cdots,m_n)$, 明文满足公式(7)所描述的关系:

$$\begin{cases} m_j=0, \text{若} B_{i,j}=-1 \\ m_j=1, \text{若} B_{i,j}=1 \end{cases} (j=1,\cdots,n) \tag{7}$$

4) 时间复杂度分析: 调用 LLL 算法所需的时间为 $n^6\log^3(n\|A\|_\infty)\approx n^6\log^3 n$ (但实际的时间远远小于这个时间界), 实验过程中维数为 1024 的时候, 花费的时间粗略统计为: 6 小时, 接着是查找特殊向量 B_i 的时间为 $O(n^2)$ (这里 B 是 $n\times n$ 的向量), 所以该启发式格攻击的时间复杂度是多项式时间.

4.3 计算实验

文中采用了 NTL 库来验证上述格攻击算法的效率. 为了节省空间, 下面给出 n=32 时的具体的相关参数值:

B=(2 3 7 15 28 57 113 227 453 907 1813 3627 7253 14510 290183 58037 116075 232151 464303 928607 1857215 3714431 7428863 14857727 29715455 59430911 118861823 237723647 475447295 950894591 1901789183 3803578367)

m= 7607417881 w= 3226643473

D=(6453286946 2072512538 7371668549 2755144809 6664420553 1340648817 7062072042 2135951795 1045260117 5317163707 7407683941 2827175593 2427707713 2547153556 3697549960 508683805 4244011083 4107247758 3833721108 3286667808 2192561208 4348008 3235339489 2089904570 7406452613 2824712937 1268651466 5763946405 7147118402 2306044515 231314622 3689272717)

U=(98469 31624 112482 42040 101690 20456 107758 32592 15949 81133 113032 43139 37043 38866 56420 7761 64758 62671 58497 50150 33455 66 49367 31889 113013 43101 19358 87950 109056 35187 3529 56293)

V=(22562 2074 48197 11369 64713 44401 43754 2483 26453 31419 18789 18089 5766531380 8840 58909 30795 41102 61716 37408 54328 22632 23777 27066 32645 45801 5578 55205 24386 29283 38078 54669)

e=1075567 f=1758795

根据以上参数, 利用 $a_i=eu_i+fv_i$, 计算出公钥 $A=(a_1,\cdots,a_n)$, 接着用加密算法计算出密文 $c=(a_1x_1+\cdots+a_nx_n)$, 最后用上述启发式格攻击算法恢复明文 $(x)_2$.

利用基于二元一次不定方程的改进背包密码算法, 根据不同的参数进行了计算实验, 实验结果如表 1 所示, 说明了启发式攻击的有效性, 从而证明了此改进背包方案的不安全性. 其中 n 表示格的维数, |m|、|w|、|e|、|f|分别表示参数的比特位长度.

表 1 不同参数格攻击实验结果

n	|m|	|w|	|e|	|f|	背包密度 n/|m|	实验次数	成功率
32	33	32	21	21	0.9696969697	10	100%
64	65	65	37	38	0.9846153846	10	100%
128	129	128	70	71	0.992248062	10	50%
256	257	257	135	136	0.9961089494	10	30%
512	513	511	265	265	0.9980596823	10	10%
1024	1025	1024	521	522	0.9990243902	10	10%

5 结语

本文利用LLL算法对一种基于二元一次不定方程的改进背包密码算法进行了格攻击. 以计算实验的方式证明即使背包密度在安全区间内也并不能保证背包方案的安全性, 虽然实验并不是百分之百的成功, 但也说明了格攻击算法的有效性以及文献[3]中改进算法的不安全性. 至于理论方面的证明, 将在下一步的研究中进行.

参考文献

1 Merkle RC, Hellman MH. Hiding information and signatures in trapdoor knapsacks.IEEE Trans.on Info.Theory,1978,IT-24 (5):525−530.

2 Shamir A. A polynomial-time algorithm for breaking the basic Merkle-Hellman cryptosystem. IEEE Trans. on Information Theory, 1984,30(5):699−704.

3 费向东,潘郁.安全背包公钥密码的要点和设计.信息网络安全,2012,(9):81−84.

4 古春生,于志敏,景征俊.基于随机背包公钥密码的格攻击.计算机应用研究,2012,(9):3486−3488.

5 Shoup V. NTL.a library for doing number theory. [2009-08-14].http://shoup.net/ntl/

6 Lenstra AK, Lenstra HW, Lovaz L. Factoring Polynomials with Rational Coefficients. Math Ann, 1982(261): 515−534.

7 Coster MJ, Joux A, LaM acchia BA, et al. Improved low-density subset sum algorithms. Computational Complexity, 1992,2(2):111−128.

8 Ajtai M. Generating hard instances of lattice problems. Proc. of the 28th Annual ACM Symposium on Theory of Computing. 1996: 99−108.

(上接第 186 页)

prefetching and caching. Proc. of the 1997 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems. Seattle, Washington, 1997: 100−114.

5 Patterson RH, Gibson GA, Ginting E. Informed prefetching and caching. 15th ACM Symposium on Operating System Principles. 1995.

6 Anderson TE, Dahlin MD, Neefe JM, Patterson DA, Roselli DS, Wang RY. Server-less network file systems. ACM Trans. on Computer Systems, ACM, February1996,14(1):41−79.

7 Hendricks J, Sambasivan RR, Sinnamohideen S, Ganger GR. Improving small file performance in object-based storage. Technical Report 06-104, Parallel Data Laboratory, Carnegie Mellon University, 2006.

8 Patil S, Gibson G. Scale and Concurrency of GIGA+: File System Directories with Millions of Files. Proc. of the 9th USENIX Conference on File and Storage Technologies (FAST'11). San Jose CA, February 2011.

9 https://btrfs.wiki.kernel.org/.April 24,2012.

10 Radkov P, Yin L, Goyal P, Sarkar P, Shenoy P. A Performance Comparison of NFS and iSCSI for IP-Networked Storage. Proc. of the 3rd USENIX Conference on File and Storage Technologies. March 31, 2004, San Francisco, CA.