计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2018,27(4):34−38 [doi: 10.15888/j.cnki.csa.006302]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

改进椭圆曲线密码体制在 SET 协议中的应用①

卢闻捷

(浙江理工大学 信息学院, 杭州 310018)
通讯作者: 卢闻捷, E-mail: lastyu@vip.qq.com

摘　要: 在电子商务中, 安全电子交易 (SET) 协议作为目前安全性较高的协议之一, 解决了一定的安全问题. 然而, 由于 SET 加解密方案安全强度不足, 其安全性逐渐受到人们的怀疑. 由此, 提出一种改进的椭圆曲线密码体制用于替代原有私钥加密算法, 提高协议的速度、性能及安全性. 针对 ECC 加解密过程中点乘法运算耗时较多而影响数据加解密速度的问题, 通过对几种改进的数乘算法进行比较, 提出一种改进的 NAF 算法. 比较可得出改进算法相对于现有算法拥有更好的时间复杂度并使用更少的计算资源. 同时融合使用 MD5 哈希生成算法进一步提高了现有椭圆曲线密码体制的安全性.

关键词: SET 协议; 椭圆曲线密码体制; 快速算法; MD5; 混合密码学

引用格式: 卢闻捷.改进椭圆曲线密码体制在 SET 协议中的应用.计算机系统应用,2018,27(4):34–38. http://www.c-s-a.org.cn/1003-3254/6302.html

Application of Improved Elliptic Curve Cryptosystem to SET Protocol

LU Wen-Jie

(School of Informatics and Electronics, Zhejiang Sci-Tech University, Hangzhou 310018, China)

Abstract: In e-commerce, as one of the most secure protocols, the SET protocol resolves some security issues. However, SET encryption and decryption program have gradually been doubted because of its lack of security guarantee. An improved elliptic curve cryptography is proposed to replace the original private key encryption algorithm which can improve the speed, performance, and security of the protocol. In this study, an improved NAF algorithm is proposed to develop the encryption and decryption speed of data affected by time-consuming ECC point multiplication. Compared with existing algorithms, the improved algorithm has better time complexity and it uses less computational resources. Besides, the MD5 hash generation algorithm is used to further improve the security of the existing elliptic curve cryptosystem.

Key words: SET protocol; elliptic curve cryptosystems; fast algorithm; MD5; hybrid cryptography

安全电子交易协议 (Secure Electronic Transaction, SET), 是一个安全的电子支付模型, 它主要为了解决用户、商家、银行的支付交易行为. SET 的出现在最大程度上帮助交易参与方提高了交易的信任度, 也保证了交易信息的安全性和完整性. 参与 SET 一次完整流程的成员有: 持卡者、商家、银行 (包括发卡行与收单行)、支付网关以及认证中心. 其中持卡人是支付卡持有者, 得到了发行者的授权; 发卡行是给持卡者发行信用卡的金融机构; 商家是提供货物或服务的经营者; 发卡行是给商家提供开设帐号的金融机构; 收单行也是一个金融机构, 它是接受付款的最终端, 为商家建立一个账户并处理支付卡授权和支付; 支付网关是金融网的安全屏障与关口, 实现对支付信息从 Internet 到银行内部网络的转换, 用来处理支付功能, 并对商家和持卡人进行认证; 认证中心是参与交易各方都信任的第三方中介组织, 为交易过程中的成员进行身份验证[1,2].

① 收稿时间: 2017-07-25; 修改时间: 2017-08-17; 采用时间: 2017-08-22; csa 在线出版时间: 2018-03-31

SET 核心技术包括数字签名、数字证书、加密算法体制等. 其中 SET 中的加密算法体制主要应用于支付过程中的数据交换, 在一般的 SET 支付环境中使用的公钥加密算法是 RSA 的公钥密码体制. 本文针对 SET 中的加密算法体制进行研究改进.

1 椭圆曲线加密算法

椭圆曲线加密算法 (Elliptic Curve Cryptography, ECC)[3–6]是基于椭圆曲线离散对数问题的密码体制, 相对于别的算法, 它的每 bit 安全性最高. 相比 RSA 密码体系, ECC 体系无论在安全性能上还是通信带宽上都具有一定的优势, 文献[3]中对相同安全级别下的 ECC 和 RSA 的密钥长度和密码长度进行了对比, 结果显示随着对于安全性能的要求加强, RSA 算法的密匙长度增加的越快, ECC 比 RSA 的硬件要求就更低.

椭圆曲线常用的定义的有限域包括两种: 素数域 $GF(p)$ 和二进制域 $GF(2^m)$. 在二进制域上, 元素在计算机中可实现并行的加、减、乘等操作, 因此硬件实现的椭圆曲线加密系统中通常使用二进制域的椭圆曲线加密方法. $GF(2^m)$ 上一个非奇异椭圆曲线 E 由 Weierstrass 公式定义的: $y^2+xy=x^3+ax^2+b$. 其中 a, b 是 $GF(2^m)$ 中的元素, 且 $b\neq 0$. 同时曲线 E 包括一个无穷远点作为曲线上点的加法单位元, 用 O 表示. 令 $P_1=(x_1, y_1)$, $P_2=(x_2, y_2)$, 是 E 上的两个点, 且 $P_1\neq -P_2$, 则 $P_3=P_1+P_2=(x_3, y_3)$ 可由下式计算:

1) $P_1\neq P_2$

$$\begin{cases} \lambda=(y_2+y_1)/(x_2+x_1) \\ x_3=\lambda^2+\lambda+x_1+x_2+a \\ y_3=\lambda(x_1+x_3)+x_3+y_1 \end{cases} \tag{1}$$

2) $P_1=P_2$

$$\begin{cases} \lambda=y_1/x_1+x_1 \\ x_3=\lambda+\lambda+a \\ y_3=\lambda(x_1+x_3)+x_3+y_1 \end{cases} \tag{2}$$

ECC 的加密、解密过程, 公钥协议都要求计算椭圆曲线数乘运算.

方案流程:

首先选择一个基域 F, 定义一个点 P 为该域上的椭圆曲线 E 和 F 上为素数阶, 点的坐标以 (x_p, y_p) 表示. 有限域 F, 域元素 a 和 b 为椭圆曲线的参数, 公开信息为点 P 和阶 n. 系统建立后每个使用者将会生成各自的密钥.

1) 选取一个随机证书 d, d 在区间[1, n–1]内

2) 计算点 $Q=dP$;

3) Q 为实体的公开密钥, 整数 d 则为实体的私钥.

生成密钥后的每个使用者将进行加密处理, 当实体 B 给实体 A 发送一条信息 M 时, 实体 B 执行步骤:

① 查找 A 的公开密钥 Q;

② 将信息 M 表示为域元素 $M\in Fq$;

③ 选择一个随机整数 k, k 属于区间[1, n–1];

④ 计算点 $(x_1, y_1)=kP$;

⑤ 计算点 $(x_2, y_2)=kQ$;

⑥ 计算 $c=mx_2$. 传送加密数据 (x_1, y_1, c) 给 A.

当实体 B 完成步骤后进行解密过程后, 实体 A 解密实体 B 传输的密文 (x_1, y_1, c), A 执行步骤:

(a) 使用他的私钥 d, 计算点 $(x_2, y_2)=d(x_1, y_1)$;

(b) 通过计算 $m=c\times x_2^{-1}$, 解析出数据 m.

上述过程中, $Q=dP$ 为公开的, 若存在第三者能够解开上述的椭圆曲线上的离散对数问题, 即能从 dP 中求出 d, 就可得到解密的消息.

2 椭圆曲线数乘算法

在有限域中曲线上的两个互异点相加, 需要 1 次求逆, 2 次乘法, 1 次平方, 9 次加法 (在有限域中加法的运算耗时相当少, 通常略去不考虑). 由椭圆曲线密钥交换协议可知, 要提高加解密速度必须提高点的数乘的效率. ECC 的加密、解密过程所要求计算的椭圆曲线数乘运算的计算如下形式:

$$Q=kP=\underbrace{P+\cdots+P}_{k} \tag{3}$$

其中 P 为椭圆曲线上一点, $k=(a_{n-1}, a_{n-2}, \cdots, a_2, a_1, a_0)$.

2.1 典型算法

算法 1. 计算从左到右二进制点乘方法

输入: $k=(a_{n-1}, a_{n-2}, \cdots, a_2, a_1, a_0)$, P 为椭圆曲线上一点.
输出: kP.

1. $Q=O$;
2. 对 i 从 n–1 到 0, 重复执行
(1) $Q=2Q$;
(2) 如 $a_i=1$, 则 $Q=Q+P$;
3. 返回 Q.

该算法平均需要 (m–1) 次倍乘, (m–1)/2 次点加, 其平均计算时间复杂度是记为: $t1=(m/2)A+(m-1)D$.

NAF (Non-Adjacent Form) 表示形式是, 在二进制域椭圆曲线上, 设 $P=(x, y)$, 则 $-P=(x, x+y)$. 大整数 k 表示为 $k=\sum_{i=0}^{n-1} k_i 2^i, k_i\in\{0, 1, -1\}$ 通过利用得到的 NAF(k) 可直接计算 kP. 对 NAF(k) 进行从左到右的扫描, 根据每一位数的正负号判断进行加或减运算. 具体

过程见算法 2.

算法 2. 二进制 NAF 方法的椭圆曲线数乘算法

输入: 正整数 k, 椭圆曲线上一点 P.
输出: kP.

1. 计算NAF(k)= $\sum_{i=0}^{m-1} k_i 2^i, k_i \in \{0,1,-1\}$;
(1) $i \leftarrow 0$;
(2) $k \geq 1$ 时, 重复执行
① 若 k 是奇数, 则 $k_i \leftarrow 2-(k \bmod 4)$, $k \leftarrow k-k_i$;
② 否则, $k_i \leftarrow 0$;
③ $k \leftarrow k/2$, $i \leftarrow i+1$;
(3) 返回 $(k_i, k_{i-1}, k_{i-2}, \cdots, k_1, k_0)$;
2. $Q=O$;
3. 对于 i 从 $m-1$ 到 0, 重复执行
(1) $Q=2Q$;
(2) 如 $a_i=1$, 则 $Q=Q+P$;
(3) 如 $a_i=-1$, 则 $Q=Q-P$;
4. 返回 Q.

NAF 形式的期望权重为 $m/3$, 因而计算其平均计算时间复杂度是 $t2=(m/3)A+(m-1)D$.

算法 3. Montgomery 数乘算法

输入: $k=(a_{n-1}, a_{n-2}, \cdots, a_2, a_1, a_0)$, P 为椭圆曲线上一点.
输出: kP.

1. $Q=O$;
2. 对 i 从 $n-1$ 到 0, 重复执行
(1) $Q=2Q$;
(2) 如 $a_i=1$, 则 $Q=Q+P$;
3. 返回 Q.

算法 3 的平均时间复杂度是 $t3=(m-1)A+(m-1)D$.

结合文献[7]的 NAF 算法本文给出了一个改进的 NAF 算法, 具体见算法 4.

算法 4. 改进的 NAF 算法

输入: $k=a_m(a_{m-1}, \cdots, a_0)a_{-1}a_{-2}$, $a_m a_{-1}a_{-2}=0$(为了方便计算, 我们在最左边添加 0, 同时在末尾添加 00, 表示不产生进位), P 为椭圆曲线上一点.
输出: kP.

1. s, t$\leftarrow m+1$; $change, begin, end, count \leftarrow 0$; $Q=P$;
2. 对于 t 从 $m+1$ 到 -1, 重复执行
如 $a_t a_{t-1}=01$
$begin \leftarrow 1$; $Q=2Q$; $Q=Q+a_{begin}P$;
如 $a_t a_{t-1}$!=00, 重复
如 $a_t a_{t-1}=11$,
$change \leftarrow 1$; $end \leftarrow t-1$; $t=t-1$;
如 $change=1$,
当 $a_{begin}a_{begin-1}a_{begin}=010$,
判断 R_{count};
如 R_{count}! =0
$begin \leftarrow t-2$; $count \leftarrow count+1$; $a_{begin}=1$; $Q=2Q$; $Q=Q+a_{begin}P$.
$begin=begin-1$.
如 $begin$!=end, 重复
$a_{begin}=a_{begin}-1$;
$a_{beigin}=-1$.
如 $begin$!=t, 重复
$Q=2Q$; $Q=Q+a_{begin}P$;
$begin=begin-1$.
否则 $begin=begin-1$.
如 $begin$!=end, 重复
$Q=2Q$; $Q=Q+a_{begin}P$;
否则 $Q=2Q$.
$begin \leftarrow 0$; $change \leftarrow 0$; $end \leftarrow 0$.
3.返回 Q.

改进算法使用随机的带符号二进制表示, 其随机性主要通过用 011 替换二进制表示的10 $\bar{1}$. 对于不同的随机数, 带符号的二进制表示的操作过程也有所不同. 新算法不需要引入随机变量产生的额外的计算资源, 只判断随机数的位 R 是 1 或 0, 判断发生替换字符的步骤是否发生, 确保了计算速度. 其平均计算时间复杂度是 $t4=(m/3)A+(m-1)D$.

2.2 算法比较

表 1 对上述算法进行了平均计算时间复杂度的比较.

表 1 几种算法的时间复杂度比较

算法名称	时间复杂度
算法1. 从左到右二进制点乘方法	$t1=(m/2)A+(m-1)D$
算法2. 二进制NAF方法	$t2=(m/3)A+(m-1)D$
算法3. Montgomery数乘算法	$t3=(m-1)A+(m-1)D$
算法4. 改进的NAF编码算法	$t4=(m/3)A+(m-1)D$

改进的算法不仅确保了 ECC 的安全性, 而且还具有更加优秀的平均计算时间复杂度. 另外也不需要存储 NAF 码直接进行点乘法, 节省了计算资源.

3 椭圆曲线密码体制与 MD5 的融合方案

提出的方案包含了应用改进的 NAF 编码算法的椭圆曲线密码体制以及 MD5 哈希生成算法. 在原有步骤中添加对信息 M 运用 MD5 进行 128 位密码的生成. 具体步骤如图 1 所示, 首先在进行加密过程中, 将信息 M 在进行 ECC 加密的同时, 进行 MD5 加密得到 128 位密码. 传输过程由只发送信息 M 替换为发送信息 M 以及 128 位密码. 原有加密过程不变, 而在解密时, 在通过 ECC 解密得到解密信息 M 后对信息 M 进

行 MD5 哈希生成算法. 通过对比解密得到的 128 位密码与接收到的密码来判断是否接受信息 M.

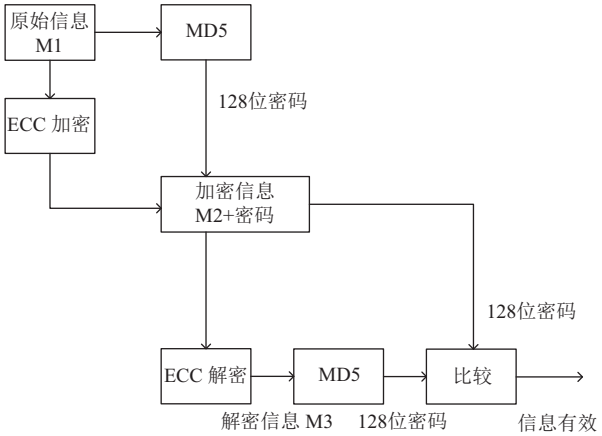

图 1 融合方案流程图

3.1 安全性分析

改进方案通过 MD5 算法生成密钥代替了原有的简单步骤, 相对原有数字签名所使用的 Hash 算法, MD5 进一步增强了算法的安全性. 而对于 ECC 的安全性, 我们通过对 ECC 与 RSA、DSA 对比分析. 在目前整数因子分解技术与方法的不断改善, 计算机运行速度的提高的大环境下, RSA 加解密安全保障的大整数要求日益变高, 为保证安全性, 就必须不断增加 RSA 的密钥长度. 目前普遍采用的安全字长为 1024 位以上, 但密钥长度的增加会直接导致解密速度的降低, 硬件实现也越发困难. 安全性分析如表 2 所示. 表中 MIPS 表示每秒执行 100 万条指令的计算机运行一年. 一般认为破译时间到达 1012 MIPS 年为安全. 从表中可以知道: 保证安全性的情况下, ECC 只需要 160 密钥长度, 而 RSA 和 DSA 则需要 1024 位. ECC 的安全性递增也大大快于 RSA 和 DSA. 攻击有限域上离散对数问题有指数积分法, 而它对椭圆曲线上的离散对数问题并不有效. 对 RSA 和 DSA 而言, 均存在指数时间算法, 而对于 ECC 至今还没有发现指数时间算法. 故在综合比较下 ECC 的安全性更高、抗攻击性更强.

表 2 破译时间比较

破译所需时间/MIPS年	RSA/DSA 密钥长度	ECC 密钥长度	RSA/ECC 密钥长度之比
104	512	106	5:1
108	764	132	6:1
1012	1024	160	7:1
1020	2048	210	10:1
1078	21 000	600	35:1

3.2 融合算法性能评估

本节将对性能进行评估并列出. 对样本文件进行随机抽取进行加解密操作. 以增加文件大小为基础进行不同的实验. 表 3 表示了有关所有加密文件的信息, 包括文件序号, 存储, 时间, 内存信息和所有加密文件的文件大小. 图 2 以及图 3 各自对表格内的内存消耗以及时间消耗给出更为直观的展示. 其中内存消耗表示所需的处理加密算法的主内存量.

表 3 文件加密时间, 内存和文件大小

序号	存储	时间(ms)	内存消耗(KB)	文件大小
1	1.0	234.0	232 158.456 24	101.0
2	1.0	93.0	50 801.514 565	100.0
3	1.0	278.0	24 634.985 215	110.0
4	17.0	187.0	56 229.657 954	191.0
5	17.0	360.0	36 057.202 975	191.0
6	17.0	203.0	18 501.985 45	45.0
7	17.0	328.0	18 405.101 5598	32.0
8	0.0	0.0	0.0	0.0
9	17.0	15.0	10 341.954 23	12.0
10	17.0	15.0	10 341.954 23	12.0
11	17.0	203.0	60 056.014 5521	198.0

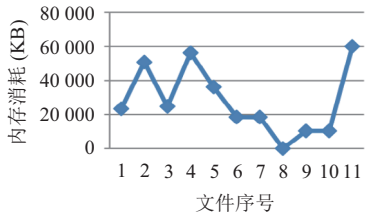

图 2 加密内存消耗

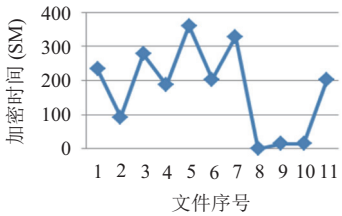

图 3 加密时间消耗

表 4 描述了有关所有解密文件的信息. 解密文件的准确性, 时间, 内存和文件大小给出的信息. 图 4 对解密的内存消耗进行了展示, 图 5 则对于解密时间消耗给出直观的展示. 图 6 中的存储开销是指待加密的原始文件额外的密码的数量. 图 7 展示了数据解密的准略率, 数据解密准确率是在加密文件解密之后准确恢复的数据量.

表 4 文件解密时间, 内存和文件大小

序号	准确率(%)	时间(ms)	内存(KB)	文件大小
1	100.0	38.0	35 942.958 4625	39.0
2	100.0	38.0	35 942.958 4625	39.0
3	100.0	69.0	40 761.875 3545	110.0
4	100.0	23.0	17 511.057 5685	39.0
5	100.0	297.0	58 830.359 8455	191.0
8	100.0	140.0	21 682.795 4395	45.0
9	100.0	141.0	21 015.242 4865	45.0
10	100.0	31.0	21 617.978 535	32.0
11	100.0	16.0	14 095.985 755	12.0

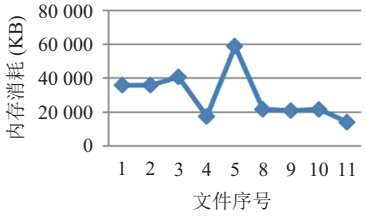

图 4 解密内存消耗

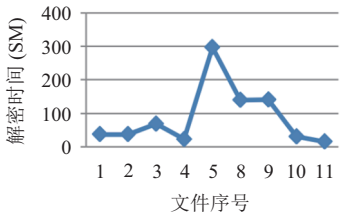

图 5 解密时间消耗

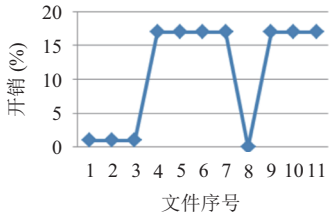

图 6 加密存储开销

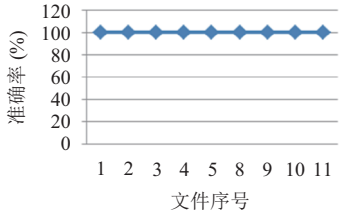

图 7 解密准确率

分析可得加解密所消耗的时间与内存都在可接受范围内. 由上图分析可得: 存储开销随着原始文件大小增加而增加, 并且总体上开销较少. 结果表明系统在解密过程中可恢复 100% 的数据. 因而算法在保证安全性的同时, 消耗较少的内存资源以及时间资源从而表明所提出的技术的有效性.

4 结论与展望

SET 协议是电子支付系统中的关键, 本文通过对 SET 协议核心技术中的加密算法体制进行研究, 使用椭圆曲线算法取代目前所流行的 RSA 算法, 从而提高了协议的安全性和工作效率. 同时本文对传统的 ECC 安全算法进行了比较, 使用一种改进的 NAF 算法. 对几种加密算法的性能进行了评估内存消耗量和时间消耗. 同时结合了椭圆曲线密码体制与 MD5 提出了改进性方案, 使用 MD5 算法代替原有随机生成密钥方法. 比较证明了改进性方案消耗较少的资源并提高了安全性.

参考文献

1 Lu S, Smolka SA. Model checking the secure electronic transaction (SET) protocol. Proceedings of the 7th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems. College Park, MD, USA. 1999. 358–365.

2 SetCo. SET secure electronic transaction specification: Business description. http://www.mastercard.com/set/set.htm. 1997.

3 魏来, 杨朵. 一个改进的椭圆曲线密码体制在物联网传输中的研究应用. 信息技术与信息化, 2015, (10): 209–212. [doi: 10.3969/j.issn.1672-9528.2015.10.079]

4 Koblitz N. Elliptic curve cryptosystems. Mathematics of Computation, 1987, 48(177): 203–209. [doi: 10.1090/S0025-5718-1987-0866109-5]

5 Miller VS. Use of elliptic curves in cryptography. Advances in Cryptology-CRYPTO '85. Santa Barbara, CA, USA. 1986. 417–426.

6 Kobayashi T, Morita H, Kobayashi K, *et al*. Fast elliptic curve algorithm combining Frobenius map and table reference to adapt to higher characteristic. Advances in Cryptology-EUROCRYPTO'99. Prague, Czech Republic. 1999. 176–189.

7 Wang X, Wang LP, Bai YC, *et al*. Optimization of elliptic curve cryptography resisting power attack scalar multiplication algorithm in security system on chip. 2015 IEEE 12th International Conference on Ubiquitous Intelligence and Computing and 2015 IEEE 12th International Conference on Autonomic and Trusted Computing and 2015 IEEE 15th International Conference on Scalable Computing and Communications and its Associated Workshops (UIC-ATC-ScalCom). Beijing, China. 2015. 1397–1401.