科研成果

BN 曲线上的 R-ate 双线性对快速计算研究

刘星江 杨竞 敖凌峰

（中国电子科技集团公司第三十研究所 四川成都 610041）

摘要：双线性对是构造各种公钥密码方案的重要工具，其计算性能对公钥密码方案的应用来说至关重要。BN 曲线上的 R-ate 双线性对是各类双线性对中计算效率较优的双线性对，也是各种公钥密码方案中采用的一类重要的双线性对。为了提升 BN 曲线上的 R-ate 双线性对的计算效率，本文提出了一种快速计算算法，可以在各类软硬件平台上实现 BN 曲线上的 R-ate 双线性对时使用，以达到提高密码方案性能水平的目的。

关键词：BN 曲线，R-ate 对，最后幂运算，w-NAF 方法

Research On Efficient Implementation of R-ate Bilinear Pairings over BN Curves

Abstract: Bilinear pairings is an important tool on construction public key cryptographic schemes, its computational performance is crucial to the application of public key cryptography. The R-ate bilinear pairings over BN curves is one of the most efficient bilinear pairings, it is also an important class of bilinear pairings used in various public key cryptographic schemes. In order to improve the computational efficiency of the R-ate bilinear pairings over BN curves, a fast algorithm is proposed in this paper. This algorithm can be used in all kinds of hardware and software platforms to improve performance of the R-ate bilinear pairings over BN curves, so as to improve the performance of cryptographic schemes.

Keywords: BN curves, R-ate pairings, Final exponentiation, w-NAF method

1、引言

在传统的公钥密码体制中，公钥是与用户身份无关的随机字符串，存在如何认证公钥的真实性、有效性的问题。公钥基础设施（Public Key Infrastructure，PKI）通过使用可信任的第三方——签证中心（Certication Authority，CA）颁发公钥证书的形式来绑定公钥和用户身份信息。关于 PKI 及其应用可以参考文献[1]和[2]。不过，PKI 证书管理复杂，需要建设复杂的 CA 系统，证书发布、吊销、验证和保存需要占用较多的资源，这就限制了 PKI 在物联网（Internet of things，IoT）、移动通信网络等场景中的广泛应用。

为了简化对公钥证书的管理，在 1984 年，Shamir 首次提出了基于身份公钥密码（Identity-Based Cryptography，IBC）的概念[3]。在 2001 年，第一个真正实用的 IBE 方案由 D. Boneh 和 M. Franklin 设计出来[4]。这之后，大量使用双线性对构造的基于身份的密码方案相继被提出，双线性对已经成为构造各种密码方案的重要工具。

2、双线性对简介

在密码学中，双线性对是指一个映射函数，它将两个群的笛卡尔乘积映射到另一个群上，两个群在映射中都能保持线性性，即双线性性。实用的双线性对都是定

义上椭圆曲线点群上的，通常有椭圆曲线上的 Weil 对、Tate 对等。自从 D. Boneh 和 M. Franklin 使用双线性对构造 IBE 方案以来，双线性对的计算技术也得到了快速发展。各种类型的双线性对相继被提出，如 Eta 对、Optimal Ate 对等[5][6][7]。

3、幂运算的改进

由第 2 节所示，在使用 Miller 算法计算 BN 曲线上的 R-ate 双线性对时，在 Miller 算法中的最后一步是一个 $F_{p^{12}}$ 上的幂运算，指数为 $(p^{12}-1)/n$，其可以分解为：

$$(p^{12}-1)/n=(p^2+1)\times(p^6-1)\times[(p^4-p^2+1)/n]$$

因此可以将这个幂运算分解如下三个部分。

（1）计算 $q=f^{p^2+1}$。由于 $f^{p^2+1}=f^{p^2}\cdot f$，利用 Frobenius 映射，仅需要一次 $F_{p^{12}}$ 上的乘法运算就可以实现。

（2）计算 $m=f^{(p^2+1)(p^6-1)}=q^{(p^6-1)}$。由于 $q^{(p^6-1)}=q^{p^6}\cdot q^{-1}$，利用 Frobenius 映射，只需要 $F_{p^{12}}$ 上的乘法运算和求逆运算各一次就可以实现。

（3）计算 $m^{(p^4-p^2+1)/n}$。这一部分由于不能直接利用 Frobenius 映射，是最困难的部分。在文献[9]中，M.Scott 给出了快速实现这个部分的一个方法。

首先将 $(p^4-p^2+1)/n$ 表示为 p 进制数 $\lambda_3p^3+\lambda_2p^2+\lambda_1p+\lambda_0$，其中

$$\lambda_3=1$$

$$\lambda_2=6u^2+1$$

$$\lambda_1=-36u^3-18u^2-12u+1$$

$$\lambda_0=-36u^3-30u^2-18u-2$$

于是，$m^{(p^4-p^2+1)/n}=y_0\cdot y_1{}^2\cdot y_2{}^6\cdot y_3{}^{12}\cdot y_4{}^{18}\cdot y_5{}^{30}\cdot y_6{}^{36}$

其中，

$$y_0=m^p\cdot m^{p^2}\cdot m^{p^3}$$

$$y_1=1/m$$

$$y_2=(m^{u^2})^{p^2}$$

$$y_3=1/(m^u)^p$$

$$y_4=1/[(m^{u^2})^p\cdot(m^u)]$$

$$y_5=1/m^{u^2}$$

$$y_6=1/[m^{u^3}\cdot(m^{u^2})^p]$$

注意到 $m=q^{(p^6-1)}$，因此 $m^{(p^6+1)}=q^{(p^6-1)(p^6+1)}=q^{(p^{12}-1)}=1$。于是 $m^{-1}=m^{p^6}$，利用 Frobenius 映射容易计算。所以，在计算这个困难部分时，主要的计算量在计算 m^u、m^{u^2}、m^{u^3} 上。而这三个幂运算通常可以采用平方-乘算法实现。

$F_{p^{12}}$ 上的平方-乘幂运算算法

输入：$m\in F_{p^{12}}$，正整数 $u=(u_{l-1}\cdots u_1u_0)_2$。

输出：m^u。

计算过程如下：

（1）设置 $f=1$；

（2）对 i 从 $l-1$ 到 0，执行：

（2.1）计算 $f=f^2$；

（2.2）若 $u_i=1$，则计算 $f=f\cdot m$；

（3）返回 f。

由上所述，$m^{-1}=m^{p^6}$，利用 Frobenius 映射容易计算。因此，在进行 m^u、m^{u^2}、m^{u^3} 幂运算时，可以采用正整数的非邻近表示型方法，减少平方-乘算法中乘法的次数。关于正整数的非邻近表示型可参考文献[10]。

一个正整数 k 的二进制表示为 $k=\sum_{i=0}^{l-1}k_i2^i=(k_{l-1}\cdots k_1k_0)_2$，这里 $k_i\in\{0,1\}$。而一个整数 k 的非邻近表示型（Non-Adjacent Form，NAF）为 $NAF(k)=\sum_{i=0}^{l-1}k_i2^i=(k_{l-1},\cdots,k_1,k_0)_2$，这里

$k_i \in \{0, \pm 1\}$。例如，$61 = (111101)_2$，而 $NAF(61) = (1,0,0,0,-1,0,1)_2$。

NAF 还可以进一步推广为 w-NAF，其中w为大于等于 2 的正整数。一个正整数k的 w-NAF 表示为 $NAF_w(k) = \sum_{i=0}^{l-1} k_i 2^i = (k_{l-1}, \cdots, k_1, k_0)_2$，这里 $k_i \in \{0, \pm 1, \pm 3, \cdots, \pm(2^{w-1} - 1)\}$。容易看出 w-NAF 有如下性质：

（1）k有唯一的 w-NAF 表示；

（2）$NAF_2(k) = NAF(k)$；

（3）$NAF_w(k)$的长度比k的二进制表示长度至多大 1；

（4）$NAF_w(k)$中非零位数大约占总位数的 $1/(w+1)$。

给定一个正整数，计算其 w-NAF 表示型的算法如下。

计算正整数的 w-NAF 表示型算法

输入：正整数k和w。

输出：$NAF_w(k)$。

计算过程如下：

（1）设置$i = 0$；

（2）如果$k \geq 1$，则循环执行：

（2.1）若k为奇数，则设置k_i为k模2^w的绝对值最小剩余，并设置$k = k - k_i$；

（2.2）否则，设置$k_i = 0$；

（2.3）设置$k = k/2$，$i = i + 1$；

（3）返回$(k_{i-1}, \cdots, k_1, k_0)_2$。

采用正整数的 w-NAF 表示型计算幂运算的算法如下。

$F_{p^{12}}$上的 w-NAF 幂运算算法

输入：$m \in F_{p^{12}}$，正整数u。

输出：m^u。

计算过程如下：

（1）将正整数u表示为 $NAF_w(u) = \sum_{i=0}^{l-1} u_i 2^i = (u_{l-1}, \cdots, u_1, u_0)_2$；

（2）预计算$m^{-1}, m^{\pm 3}, \cdots, m^{\pm(2^w-1)}$；

（3）设置$f = 1$；

（4）对i从$l-1$到 0，执行：

（4.1）计算$f = f^2$；

（4.2）若$u_i \neq 0$，则计算$f = f \cdot m^{u_i}$；

（5）返回f。

4、性能分析

在一次 BN 曲线上的 R-ate 对的计算过程中需要三次幂运算，这里针对计算一次$F_{p^{12}}$上的幂运算m^u进行分析，假设这里的u是一个t比特的随机正整数。

在采用标准的平方-乘算法计算时，平均来说需要$F_{p^{12}}$上的$t/2$次乘法运算和t次平方运算。

在采用 w-NAF 算法计算时，平均来说需要$F_{p^{12}}$上的 $t/(w+1)$次乘法运算和t次平方运算。另外，预计算步骤还额外需要$F_{p^{12}}$上的$2^{w-2} - 1$次乘法运算和 1 次平方运算，$F_{p^{12}}$上的平方运算通常会比乘法运算略小，为了便于计算，这里不妨假设这 1 次平方运算与乘法运算相同。因此，总计需要$F_{p^{12}}$上的$t/(w+1) + 2^{w-2}$次乘法运算和t次平方运算。

由此可以看出，在采用 w-NAF 幂运算算法时，并不是w取值越大越好。虽然随着w的增大，$t/(w+1)$一直在减小，但是2^{w-1}却在指数级增大。

在t取 64、96、128 时，幂运算m^u的运算量如下表所示，其中 M 代表$F_{p^{12}}$上的乘法运算，D 代表$F_{p^{12}}$上的

平方运算。

	$t=64$	$t=96$	$t=128$
标准平方-乘算法	32M+64D	48M+96D	64M+128D
2-NAF 算法	22.3M+64D	33M+96D	43.7M+128D
3-NAF 算法	18M+64D	26M+96D	34M+128D
4-NAF 算法	16.8M+64D	23.2M+96D	29.6M+128D
5-NAF 算法	18.7M+64D	24M+96D	29.3M+128D
6-NAF 算法	25.1M+64D	29.7M+96D	34.3M+128D

由上表可以看出，当t取 64 和 96 时，4-NAF 算法最优；当t取 128 时，5-NAF 算法最优。

另外，如果一个密码方案中的椭圆曲线参数固定的情况下，这里的参数u将是一个固定的正整数，可以根据其具体的取值来评估 w-NAF 中的w取何值为最优。

5、结语

本文介绍了计算 BN 曲线上的 R-ate 对的 Miller 算法，并介绍了 M.Scott 等人所提出的最后一步幂运算的快速计算算法。在此基础之上，分析了幂运算中底数具有的逆容易计算的特性，并提出了一种基于 w-NAF 方法的快速幂运算算法。在各种不同的软硬件平台上实现 BN 曲线上的 R-ate 对时均可以采用该方法，以提高实现的性能。

参考文献:

[1]陆强，朱万红，李昊. PKI 技术的算法实现及体系结构[J]. 信息安全与通信保密, 2006(9): 131-135.

[2]尹晓晖. PKI 技术在应用系统中的应用[J]. 信息安全与通信保密, 2008(3): 64-65.

[3]A. Shamir. Identity-based Cryptosystems and Signature Schemes[J]. Advances in cryptology. Springer Berlin Heidelberg, 1985: 47-53.

[4]D. Boneh, M. Franklin. Identity-based encryption from the Weil pairing[J]. Advances in Cryptology-CRYPTO 2001, vol. 2139 of LNCS, pages 213-29. Springer-Verlag, 2001.

[5]F. Vercauteren. Optimal pairings[J]. IEEE transactions on information Theory, 56(1): 455-461, January 2010.

[6]P. Barreto, H. Kim, B. Lynn, M. Scott. Efficient algorithms for pairing-based cryptosystems[J]. Advances in Cryptology-CRYPTO 2002, Springer LNCS, Vol. 2442, pages 354-368, 2002.

[7]P. Barreto, S. Galbraith, C. Eigeartaigh, M. Scott. Efficient pairing computation on supersingular Abelian varieties[J]. Designs, Codes and Cryptography, 42: 239-271, 2007.

[8]P. Barreto, M. Naehrig. Pairing-friendly elliptic curves of prime order[J]. Selected Areas in Cryptography-SAC2005, vol. 3897 of LNCS, pages 319-331. Springer 2006.

[9]M. Scott, N. Benger, M. Charlemagne, L.J. Dominguez Perez, E.J. Kachisa. On the Final Exponentiation for Calculating Pairings on Ordinary Elliptic Curves.

[10]D. Hankerson, A. Menezes, S. Vanstone. Guide to Elliptic Curve Cryptography[M]. New York: Springer-Verlag, 2004.