ELSEVIER

Information Processing Letters 68 (1998) 227–233

Information Processing Letters

How to construct efficient signcryption schemes on elliptic curves

Yuliang Zheng [a,*], Hideki Imai [b,1]

[a] *School of Computing and Information Technology, Monash University, Melbourne, VIC 3199, Australia*
[b] *Institute of Industrial Science, University of Tokyo, 7-22-1 Roppongi, Minato-ku, Tokyo 106-8558, Japan*

Received 16 June 1998; received in revised form 9 October 1998
Communicated by K. Iwama

Abstract

Signcryption is a new paradigm in public key cryptography. A remarkable property of a signcryption scheme is that it fulfills both the functions of public key encryption and digital signature, *with a cost significantly smaller than that required by signature-then-encryption*. The purposes of this paper are to demonstrate how to specify signcryption schemes on elliptic curves over finite fields, and to examine the efficiency of such schemes. Our analysis shows that when compared with signature-then-encryption on elliptic curves, signcryption on the curves represents a 58% saving in computational cost and a 40% saving in communication overhead.

Keywords: Cryptography; Digital signature; Elliptic curves; Encryption; Public key cryptography; Signcryption

1. Introduction

Public key cryptography discovered nearly two decades ago [2] has revolutionized the way for people to conduct secure and authenticated communications. Currently the standard approach to achieving both message confidentiality and authenticity is *signature followed by encryption*, namely before a message is sent out, the sender of the message would sign it using a digital signature scheme, and then encrypt the message (and the signature) using a private key encryption algorithm under a randomly chosen message encryption key. The random message encryption key would then be encrypted using the recipient's public key. We call this two-step approach "signature-then-encryption".

Signature generation and encryption consume machine cycles, and also introduce "expanded" bits to an original message. Symmetrically, a comparable amount of computation time is generally required for signature verification and decryption. Hence the cost of a cryptographic operation on a message is typically measured in the message expansion rate and the computational time invested by both the sender and the recipient. With the current standard signature-then-encryption approach, the cost for delivering a message in a secure and authenticated way is essentially the sum of the cost for digital signature and that for encryption.

As realized both by practitioners and theorists in data security, the standard signature-then-encryption approach, together with the fact that cryptanalytic

* Corresponding author. Email: yuliang@pscit.monash.edu.au.
[1] Email: imai@iis.u-tokyo.ac.jp.

0020-0190/98/$ – see front matter
PII: S0020-0190(98)00167-7

attacks have been advancing at a remarkable speed in recent times, is posing an increasingly large problem in security applications where efficiency both in terms of computational time and communication overhead is a critical issue. Such applications include those based on smart cards which usually employ only less powerful CPUs than do their counterparts in desk-top or notebook computers.

To solve the above problem, in [15] (see also [17, 16]) a new paradigm in public key cryptography, called *signcryption*, has been proposed. More specifically, a signcryption scheme is a cryptographic method that fulfills both the functions of secure encryption and digital signature, but *with a cost smaller than that required by signature-then-encryption*. Using the terminology in cryptography, it consists of a pair of (polynomial time) algorithms (S, U), where S is called the *signcryption algorithm*, and U the *unsigncryption algorithm*. S in general is probabilistic, but U is most likely to be deterministic. (S, U) satisfy the following conditions:

(1) *Unique unsigncryptability*—Given a message m of arbitrary length, the algorithm S *signcrypts* m and outputs a *signcrypted text* c. On input c, the algorithm U *unsigncrypts* c and recovers the original message un-ambiguously.
(2) *Security*—(S, U) fulfill, simultaneously, the properties of a secure encryption scheme and those of a secure digital signature scheme. These properties mainly include: confidentiality of message contents, unforgeability, and non-repudiation.
(3) *Efficiency*—The computational cost, which includes the computational time involved both in signcryption and unsigncryption, and the communication overhead or added redundant bits, of the scheme is *smaller* than that required by the best currently known signature-then-encryption scheme with comparable parameters.

Signcryption schemes are compact and particularly suited for efficiency-critical applications such as smart card based systems. We have identified a large number of practical applications of signcryption, including for instance (1) secure and authenticated key establishment in a single small data packet [18], (2) secure multi-casting over the Internet [7], (3) authenticated key recovery [11], (4) secure ATM networks [4], and (5) secure and light weight electronic transaction protocols [5]. We are currently in the process of searching for other novel applications of signcryption in efficient public key solutions.

In [15], it has been shown that the ElGamal signature scheme based on the discrete logarithm problem in finite fields and all its variants can be made shorter, and these shortened signature schemes can all be used to construct efficient signcryption schemes. The aim of this paper is to complete the description of the corresponding signcryption schemes on elliptic curves, and to compare their efficiency with that of signature-then-encryption on elliptic curves.

Organization of the remainder of this paper: Section 2 surveys the necessary background information on the discrete logarithm problem on elliptic curves over finite fields. Section 3 shows how to specify a signcryption scheme on an elliptic curve. The paper is closed by Section 4 where a detailed analysis of the efficiency of the signcryption schemes is carried out, from which we conclude that, when compared with signature-then-encryption, elliptic curve signcryption can save 58% in computational cost and 40% in communication overhead.

2. Elliptic curve cryptography

The original ElGamal public key encryption and digital signature schemes are defined on finite fields. In 1985 Neal Koblitz from the University of Washington and Victor Miller then with IBM observed that discrete logarithm on elliptic curves over finite fields appeared to be intractable and hence ElGamal's encryption and signature schemes have natural counterparts on these curves.

2.1. Elliptic curve groups over a finite field

Let $GF(p^m)$ be the finite field of p^m elements, where p is a prime and m an integer, an elliptic curve over $GF(p^m)$ is defined as the set of solutions (x, y), where $x, y \in GF(p^m)$, to a cubic equation

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

with $a_1, a_2, a_3, a_4, a_6 \in GF(p^m)$, together with a special point $\mathcal{O}$ called the *point at infinity*. In cryptographic practice, we are particularly interested in two types of elliptic curves: (1) curves over $GF(2^m)$ with $m > 150$, and (2) curves over $GF(p)$ with p a large prime.

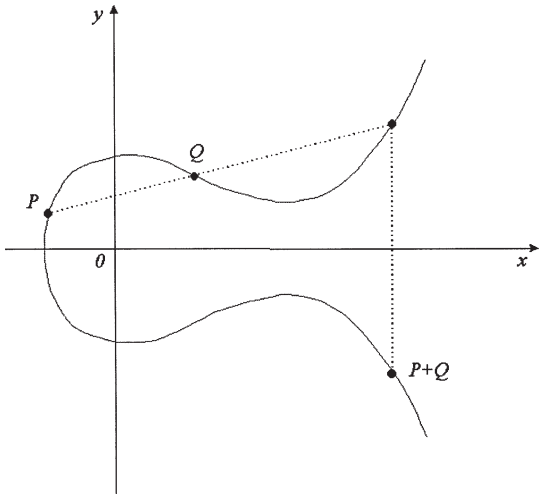

Fig. 1. Addition on an elliptic curve.

For $GF(2^m)$, the cubic equation for an elliptic curve takes the form of

$$\begin{cases} y^2 + cy = x^3 + ax + b, \\ \quad \text{with } a, b, c \in GF(2^m),\ c \neq 0 \text{ and } j\text{-variant } 0, \\ \text{or} \\ y^2 + xy = x^3 + ax^2 + b, \\ \quad \text{with } a, b \in GF(2^m),\ b \neq 0 \\ \quad \text{and } j\text{-variant not } 0. \end{cases} \tag{1}$$

And for $GF(p)$, $p > 3$, the cubic equation takes the form of

$$y^2 = x^3 + ax + b, \quad \text{with } a, b \in GF(p) \text{ and } 4a^3 + 27b^2 \neq 0. \tag{2}$$

An elliptic curve over $GF(p^m)$ forms an abelian group under an addition on the points given by the "tangent and chord method". To be precise, this group should be called an *elliptic curve group* over $GF(p^m)$. In this paper we follow a common practice to call the group an elliptic curve over $GF(p^m)$. The geometric meaning of the addition operation on an elliptic curve is shown in Fig. 1.

The addition on an elliptic curve only involves a few arithmetic operations in $GF(p^m)$, and hence is efficient. Taking an elliptic curve C on $GF(p)$ with $p > 3$ as an example, the addition follows the rules specified below:

(1) $\mathcal{O} + \mathcal{O} = \mathcal{O}$.
(2) $P + \mathcal{O} = P$ for all $P = (x, y) \in C$. Namely, C has $\mathcal{O}$ as its identity element.
(3) $P + Q = \mathcal{O}$ for all $P = (x, y) \in C$ and $Q = (x, -y)$. Namely, the inverse of (x, y) is simply $(x, -y)$.
(4) Adding two distinct points—for all $P = (x_1, y_1) \in C$ and $Q = (x_2, y_2) \in C$ with $x_1 \neq x_2$, $P + Q = (x_3, y_3)$ is defined by

$$x_3 = \lambda^2 - x_1 - x_2,$$
$$y_3 = \lambda(x_1 - x_3) - y_1,$$

where $\lambda = (y_2 - y_1)/(x_2 - x_1)$.
(5) Doubling a point—for any $P = (x, y) \in C$ with $y \neq 0$, $2P = (x^*, y^*)$ is defined by

$$x^* = \lambda^2 - 2x,$$
$$y^* = \lambda(x - x^*) - y,$$

where $\lambda = (3x^2 + a)/(2y)$.

Adding and doubling points on an elliptic curve C over $GF(2^m)$ are defined in a similar way.

Excluding the point at infinity $\mathcal{O}$, every point $P = (x, y)$ on an elliptic curve C over $GF(p^m)$ can be represented as (or "compressed" to) $P = (x, \tilde{y})$ where $\tilde{y}$ is a single bit:

(1) if $x = 0$, then $\tilde{y} = 0$,
(2) if $x \neq 0$, then $\tilde{y}$ is the parity of y when it is viewed as an integer.

An advantage of compressed representation of a point is that when a compressed point is stored internally in a computer or communicated over a network, it takes only one bit more than half of the bits required for storing or transmitting its uncompressed counterpart. This advantage, however, is not for free: recovering the y-coordinate from a compressed point involves a few arithmetic operations in the underlying finite field.

2.2. Elliptic curve discrete logarithms

A result due to Hasse states that the order $\#C$ of an elliptic curve C over $GF(p^m)$, i.e., the number of elements in the group, satisfies the following condition

$$\#C = p^m + 1 - t, \quad \text{with } |t| \leqslant 2\sqrt{p^m}, \tag{3}$$

where t is called the *trace of the elliptic curve* C, or to be more precise, the *trace of the Frobenius endomorphism* of C. Structurally, C is known to be isomorphic to $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2}$, where both n_1 and n_2 are integers, $n_2 | n_1$, $n_2 | (p^m - 1)$ and $\mathbb{Z}_n$ denotes the modular ring of n elements.

Let G be a point on an elliptic curve C over $GF(p^m)$. The order of G is the smallest integer q such that $qG = \mathcal{O}$. For an integer e, the e multiple of G, namely eG, can be readily computed by using a method similar to the "square-and-multiply" for exponentiation in $GF(p)$. The inverse problem corresponding to the computation of a multiple of a point is that given two points G and P in C, one is asked to find an integer e such that $P = eG$, provided that such an integer exists. This is known as the elliptic curve discrete logarithm problem. When the order q of G contains a large prime factor, say of size at least 2^{160}, it is believed that the elliptic curve discrete logarithm problem is infeasible to solve. All elliptic curve based cryptosystems hinge their security on the (purported) hardness of the elliptic curve discrete logarithm problem.

In light of recent developments in cracking the elliptic curve discrete logarithm problem [8,14,13], however, one should be very cautious in designing a cryptosystem based on the elliptic curve discrete logarithm problem. In particular, it has been shown in [8] that the discrete logarithm problem on a super-singular elliptic curve is not more difficult to solve than the discrete logarithm problem in a finite field. Super-singular elliptic curves on $GF(p^m)$ are curves whose trace t satisfies the condition of

$$t = \pm\sqrt{i \cdot p^m} \quad \text{with } i = 0, 1, 2, 3, \text{ or } 4.$$

A more recent breakthrough is dramatic indeed: Nigel Smart at HP Labs in UK, and Takakazu Satoh and Kiyomichi Araki in Japan announced that they have independently broken the discrete logarithm problem on anomalous elliptic curves over $GF(p)$ [14,13] (see also [1]). An anomalous elliptic curve over $GF(p)$ is a curve whose trace is 1, i.e., a curve that has exactly p points. In their preprint, Satoh and Araki present an algorithm that solves the elliptic curve discrete logarithm problem for trace 1 in $O((\log p)^3)$ steps.

Let us assume, optimistically, that the effectiveness of the algorithms reported in [8,14,13] is limited to super-singular and anomalous elliptic curves. Then the fastest known algorithm for the discrete logarithm problem on other elliptic curves appears to take time in the order of $O(\sqrt{p^m})$ which grows exponentially with the size of the elliptic curve group. In other words, on elliptic curves which are not super-singular or with trace 1, the discrete logarithm problem appears to share a similar degree of hardness with the discrete logarithm problem in a sub-group of comparable order modulo a large prime. This point is the origin of signcryption schemes to be introduced in the next section.

3. Elliptic curve signcryption schemes

As we mentioned earlier, ElGamal public key encryption and digital signature schemes and their variants can all be extended to elliptic curves in a straightforward way (see for instance, [6]). For the sake of completeness, Table 1 summarizes an elliptic curve version of the Digital Signature Standard or DSS [10], together with its shortened variants. The elliptic curve DSS will be called ECDSS, and its two shortened versions SECDSS1 and SECDSS2, respectively. Note that in the computation of $r = (vG) \bmod q$ with ECDSS, $vG = K$ which is a point on an elliptic curve is viewed as an integer. Similarly, in $r = hash(vG, m)$ with SECDSS1 and SECDSS2, vG is viewed as a binary string. Also note that instead of vG, one may involve only its x-coordinate in the computation of r, as the y-coordinate carries essentially only one bit of information and hence may be excluded.

Parameters for elliptic curve based signcryption schemes are summarized in Table 2, and two signcryption schemes built on SECDSS1 and SECDSS2 are described in Table 3. These signcryption schemes are called ECSCS1 and ECSCS2, respectively. Similarly to elliptic curve signature schemes described in Table 1, points on an elliptic curve, namely vP_a, $uP_a + urG$ and $uG + urP_a$, are regarded as binary strings when involved in hashing. The *bind_info* part in the computation of r may contain, among other data, the public keys or public key certificates of both Alice the sender and Bob the recipient.

4. A comparison with elliptic curve signature-then-encryption

4.1. Saving in computational cost

With the signature-then-encryption based on SECDSS1 or SECDSS2 and elliptic curve ElGamal encryption, the number of computations of multiples of points is three, both for the process of

Table 1
Elliptic curve DSS and its shortened and efficient variants

Shortened schemes	Signature (r, s) on a message m	Verification of signature	Length of signature
ECDSS	$r = (vG) \bmod q$ $s = ((hash(m) + v_a r)/v) \bmod q$	$K = s'(hash(m)G + rP_a)$ where $s' = (1/s) \bmod q$, check whether $K \bmod q = r$	$2\|q\|$
SECDSS1	$r = hash(vG, m)$ $s = (v/(r + v_a)) \bmod q$	$K = s(P_a + rG)$ check whether $hash(K, m) = r$	$\|hash(\cdot)\| + \|q\|$
SECDSS2	$r = hash(vG, m)$ $s = (v/(1 + v_a \cdot r)) \bmod q$	$K = s(G + rP_a)$ check whether $hash(K, m) = r$	$\|hash(\cdot)\| + \|q\|$

C: an elliptic curve over $GF(p^m)$, either with $p \geqslant 2^{150}$ and $m = 1$ or $p = 2$ and $m \geqslant 150$ (public to all).
q: a large prime whose size is approximately of $|p^m|$ (public to all).
G: a point with order q, chosen randomly from the points on C (public to all).
hash: a one-way hash function (public to all).
v: a number chosen uniformly at random from $[1, \ldots, q-1]$.
v_a: Alice's private key, chosen uniformly at random from $[1, \ldots, q-1]$.
P_a: Alice's public key ($P_a = v_a G$, a point on C).

Table 2
Parameters for elliptic curve signcryption

Parameters public to all:
C—an elliptic curve over $GF(p^m)$, either with $p \geqslant 2^{150}$ and $m = 1$ or $p = 2$ and $m \geqslant 150$ (public to all).
q—a large prime whose size is approximately of $\|p^m\|$ (public to all).
G—a point with order q, chosen randomly from the points on C (public to all).
hash—a one-way hash function whose output has, say, at least 128 bits.
KH—a keyed one-way hash function.
(E, D)—the encryption and decryption algorithms of a private key cipher.
Alice's keys:
v_a—Alice's private key, chosen uniformly at random from $[1, \ldots, q-1]$.
P_a—Alice's public key ($P_a = v_a G$, a point on C).
Bob's keys:
v_b—Bob's private key, chosen uniformly at random from $[1, \ldots, q-1]$.
P_b—Bob's public key ($P_b = v_b G$, a point on C).

signature-then-encryption and that of decryption-then-verification.

We note that the "square-and-multiply" method for fast exponentiation can be adapted to a "doubling-and-addition" method for the fast computation of a multiple of a point on an elliptic curve. Namely a multiple can be obtained in about $1.5|q|$ point additions.

Among the three multiples for decryption-then-verification, two are used in verifying a signature.

Table 3
Implementations of signcryption on elliptic curves

Signcryption of m by Alice the Sender		Unsigncryption of (c, r, s) by Bob the Recipient
$v \in_R [1, \ldots, q-1]$		$u = sv_b \bmod q$
$(k_1, k_2) = hash(vP_b)$		$(k_1, k_2) = hash(uP_a + urG)$
$c = E_{k_1}(m)$		if SECDSS1 is used, or
$r = KH_{k_2}(m, bind\_info)$	$\Rightarrow c, r, s \Rightarrow$	$(k_1, k_2) = hash(uG + urP_a)$
$s = (v/(r + v_a)) \bmod q$		if SECDSS2 is used.
if SECDSS1 is used, or		$m = D_{k_1}(c)$
$s = (v/(1 + v_a \cdot r)) \bmod q$		Accept m only if
if SECDSS2 is used.		$KH_{k_2}(m, bind\_info) = r$

More specifically, these two multiples are spent in computing $e_1G + e_2P_a$ for two integers e_1 and e_2. Shamir's technique for fast computation of the product of multiple exponentials with the same modulo (see [3] as well as Algorithm 14.88 on p. 618 of [9]) can be adapted to the fast computation of $e_1G + e_2P_a$. Thus on average, the computational cost for $e_1G + e_2P_a$ is $(1 + 3/4)|q|$ point additions, or equivalently 1.17 point multiples. That is, the number of point multiples involved in decryption-then-verification can be reduced from 3 to 2.17. Consequently, the combined computational cost of the sender and the recipient is 5.17 point multiples.

In contrast, with ECSCS1 and ECSCS2, the number of point multiples is one for the process of signcryption and two for that of unsigncryption respectively. Applying Shamir's technique, one reduces the computational cost of unsigncryption from 2 multiples to 1.17 on average. The total average computational cost for signcryption is therefore 2.17 point multiples. This represents a

$$\frac{5.17 - 2.17}{5.17} = 58\%$$

reduction in average computational cost.

4.2. Saving in communication overhead

To simplify our discussions, we assume that $|q| \approx |p^m|$. Namely the order q of G is of comparable size to p^m. In addition we assume that

$$\left|hash(\cdot)\right| = \left|KH.(\cdot)\right| = \tfrac{1}{2}|q|.$$

Furthermore, we assume that a point on an elliptic curve is represented in a compressed way.

Under these reasonable assumptions, the communication overhead measured in bits is $|hash(\cdot)| + |q| + |p^m + 1| \approx |hash(\cdot)| + 2|q|$ for signature-then-encryption based on SECDSS1 or SECDSS2 and elliptic curve ElGamal encryption, and $|KH.(\cdot)| + |q|$ for the two signcryption schemes ECSCS1 and ECSCS2. This gives rise to the saving in communication overhead as follows

$$\frac{|hash(\cdot)| + 2|q| - (|KH.(\cdot)| + |q|)}{|hash(\cdot)| + 2|q|} = \frac{|q|}{(1/2)|q| + 2|q|} = 40\%.$$

In conclusion, when compared with signature-then-encryption on elliptic curves, signcryption on the curves represents a 58% saving in computational cost and a 40% saving in communication overhead. A final remark is that the signcryption schemes can be built also on hyperelliptic curves ([12] is a good source on how to select secure hyperelliptic curves), and all the above analysis remains valid for these schemes.

Acknowledgment

The authors would like to thank anonymous referees for helpful comments.

References

[1] K. Araki, K. Satoh, S. Miura, Overview of elliptic curve cryptography, in: Proc. 1998 International Workshop on Practice and Theory in Public Key Cryptography (PKC '98), Lecture Notes in Comput. Sci., Vol. 1431, Springer, Berlin, 1998, pp. 29–49.

[2] W. Diffie, M. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory IT-22 (6) (1976) 472–492.

[3] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Trans. Inform. Theory IT-31 (4) (1985) 469–472.

[4] C. Gamage, J. Leiwo, Y. Zheng, A block-based approach to secure ATM networking, 1997 (submitted for publication).

[5] G. Hanaoka, Y. Zheng, H. Imai, LITESET: A light-weight secure electronic transaction protocol, in: Information Security and Privacy—Proceedings of ACISP '98, Lecture Notes in Comput. Sci., Vol. 1438, Springer, Berlin, 1998, pp. 215–226.

[6] IEEE, Standard Specifications For Public Key Cryptography (P1363), The Institute of Electrical and Electronics Engineers, August 1998 (Draft Version 6, http://grouper.ieee.org/groups/1363/).

[7] K. Matsuura, Y. Zheng, H. Imai, Compact and flexible resolution of cbt multicast key-distribution, in: Proc. Second International Conference on Worldwide Computing and Its Applications (WWCA '98), Lecture Notes in Comput. Sci., Vol. 1368, Springer, Berlin, 1998, pp. 190–205.

[8] A. Menezes, T. Okamoto, S. Vanstone, Reducing elliptic curve logarithms to logarithms in a finite field, IEEE Trans. Inform. Theory IT-39 (5) (1993) 1639–1646.

[9] A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.

[10] National Institute of Standards and Technology, Digital signature standard (DSS), Federal Information Processing Standards Publication FIPS PUB 186, U.S. Department of Commerce, May 1994.

[11] T. Nishioka, K. Matsuura, Y. Zheng, H. Imai, A proposal for authenticated key recovery system, in: Proc. 1997 Joint Workshop on Information Security and Cryptography (JW-ISC '97), Seoul, South Korea, 1997, pp. 189–196.

[12] Y. Sakai, K. Sakurai, H. Ishizuka, Secure hyperelliptic cryptosystems and their performance, in: Proc. 1998 International Workshop on Practice and Theory in Public Key Cryptography (PKC '98), Lecture Notes in Comput. Sci., Vol. 1431, Springer, Berlin, 1998, pp. 164–181.

[13] T. Satoh, K. Araki, Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves, October 1997.

[14] N. Smart, A message posted to the number theory list <nmbrthry@listserv.nodak.edu>, September 1997.

[15] Y. Zheng, Digital signcryption or how to achieve cost(signature & encryption) << cost(signature) + cost(encryption), in: Advances in Cryptology—CRYPTO '97, Lecture Notes in Comput. Sci., Vol. 1294, Springer, Berlin, 1997, pp. 165–179.

[16] Y. Zheng, Shortened digital signature, signcryption and compact and unforgeable key agreement schemes, August 1998 (a submission to IEEE P1363: Standard Specifications For Public Key Cryptography, http://grouper.ieee.org/groups/1363/).

[17] Y. Zheng, Signcryption and its applications in efficient public key solutions, in: Information Security—Proc. 1997 Information Security Workshop (ISW '97), Lecture Notes in Comput. Sci., Vol. 1396, Springer, Berlin, 1998, pp. 291–312.

[18] Y. Zheng, H. Imai, Compact and unforgeable session key establishment over an ATM network, in: Proc. IEEE INFOCOM '98, IEEE, San Francisco, 1998, pp. 411–418.