Digital Signcryption or How to Achieve Cost(Signature & Encryption) << Cost(Signature) + Cost(Encryption) *

Yuliang Zheng

Monash University, McMahons Road, Frankston, Melbourne, VIC 3199, Australia
Email: yuliang@mars.fcit.monash.edu.au

Abstract. Secure and authenticated message delivery/storage is one of the major aims of computer and communication security research. The current standard method to achieve this aim is "(digital) signature followed by encryption". In this paper, we address a question on the cost of secure and authenticated message delivery/storage, namely, *whether it is possible to transport/store messages of varying length in a secure and authenticated way with an expense less than that required by "signature followed by encryption".* This question seems to have never been addressed in the literature since the invention of public key cryptography. We then present a positive answer to the question. In particular, we discover a new cryptographic primitive termed as "signcryption" which *simultaneously* fulfills both the functions of digital signature and public key encryption in a logically single step, and with a cost *significantly* lower than that required by "signature followed by encryption". For typical security parameters for high level security applications (size of public moduli = 1536 bits), signcryption costs 50% (31%, respectively) less in computation time and 85% (91%, respectively) less in message expansion than does "signature followed by encryption" based on the discrete logarithm problem (factorization problem, respectively).

Keywords

Authentication, Digital Signature, Encryption, Key Distribution, Secure Message Delivery/Storage, Public Key Cryptography, Security, Signcryption.

1 Introduction

To avoid forgery and ensure confidentiality of the contents of a letter, for centuries it has been a common practice for the originator of the letter to sign his/her name on it and then seal it in an envelope, before handing it over to a deliverer.

* Patent pending (PO3234/96, filed on October 25, 1996). The full version of this paper can be obtained from `http://www-pscit.fcit.monash.edu.au/~yuliang/`

Public key cryptography discovered nearly two decades ago [7] has revolutionized the way for people to conduct secure and authenticated communications. It is now possible for people who have never met before to communicate with one another in a secure and authenticated way over an open and insecure network such as Internet. In doing so the same two-step approach has been followed. 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. 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, 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.

In this paper, we address a question on the cost of secure and authenticated message delivery, namely, *whether it is possible to transfer a message of arbitrary length in a secure and authenticated way with an expense less than that required by signature-then-encryption*. This question seems to have never been addressed in the literature since the invention of public key cryptography. We then present a positive answer to the question. In particular, we discover a new cryptographic primitive termed as "signcryption" which *simultaneously* fulfills both the functions of digital signature and public key encryption in a logically single step, and with a cost *significantly* smaller than that required by signature-then-encryption. The saving in cost grows proportionally to the size of security parameters. Hence it will be more significant in the future when larger parameters are required to compensate theoretical and technological advances in cryptanalysis.

2 The Traditional Signature-Then-Encryption Approach

As we mentioned earlier, public key cryptography invented by Diffie and Hellman [7] makes it a reality for one (1) to digitally sign a message, and (2) to send a message securely to another person with whom no common encryption key has been shared. Currently, the standard approach for a user, say Alice, to send a secure and authenticated message to another user Bob is signature-then-encryption. The best example that follows the two-step approach is PEM, a standard for secure e-mail on Internet [15].

To compare the efficiency of two different methods for secure and authenticated message delivery, we examine two types of "cost" involved: (1) computational cost, and (2) communication overhead (or storage overhead for stored messages). The *computational cost* indicates how much computational effort has to be invested both by the sender and the recipient of a message. We estimate

the computational cost by counting the number of dominant operations involved. Typically these operations include private key encryption and decryption, hashing, modulo addition, multiplication, division (inversion), and more importantly, exponentiation. In addition to computational cost, digital signature and encryption based on public key cryptography also require extra bits to be appended to a message. We call these extra "redundant" bits the *communication overhead* involved. We say that a message delivery method is superior to another if (the aggregated value of) the computational cost and communication overhead required by the former is less than that by the latter.

The first part of Table 2 indicates the computational cost and communication overhead of "Schnorr signature-then-ElGamal encryption" against that of "DSS-then-ElGamal encryption" and "RSA signature-then-RSA encryption". Note that, although not shown in the table, other combinations such as "Schnorr signature-then-RSA encryption" and "RSA signature-then-ElGamal encryption" may also be used in practice. As discussed in [21], with the current state of the art, computing discrete logarithm on $GF(p)$ and factoring a composite n of the same size are equally difficult. This simplifies our comparison of the efficiency of a cryptographic scheme based on RSA against that based on discrete logarithm, as we can assume that the moduli n and p are of the same size.

We close this section by examining the increasingly disproportionate cost for secure and authenticated message delivery in the currently standard signature-then-encryption approach, with an example text of 10000 bits (which corresponds roughly to a 15-line email message). For current and low security level applications, when RSA is used, the computational cost is centered around the execution of four (4) exponentiations modulo 512-bit integers, and the communication overhead is 1024 bits. When Schnorr signature and ElGamal encryption are used, the computational cost consists mainly of six (6) exponentiations modulo 512-bit integers, and the communication overhead is about 750 bits.

However, if the contents of the text are highly sensitive, or a text of the same length will be transmitted in 2010, then very large moduli, say of 5120 bits, might have to be employed. In such a situation, if RSA is used, four (4) exponentiations modulo (very large!) 5120-bit integers have to be invested in computation [2], and the communication overhead is 10240 bits, which is now longer than the original 10000-bit text ! If, instead, Schnorr signature and ElGamal encryption is used, then the computational cost is six (6) exponentiations modulo (again very large!) 5120-bit integers, and the communication overhead of about 5560 bits is more than half of the length of the original message. From this example, one can see that in the signature-then-encryption approach, the cost, especially communication overhead, for secure and authenticated message delivery, is becoming disproportionately large for future, or current but high-level security, applications. This observation serves as further justification on the necessity of inventing a new and more economical method for secure and authenticated message delivery.

[2] The number of bit operations required by exponentiation modulo an integer is a cubic function of the size of the modulo.

3 Digital Signcryption — A More Economical Approach

Over the past two decades since public key cryptography was invented, signature-then-encryption has been a standard method for one to deliver a secure and authenticated message of arbitrary length, and no one seems to have ever questioned whether it is absolutely necessary for one to use the sum of the cost for signature and the cost for encryption to achieve both contents confidentiality and origin authenticity.

Having posed a question that is of fundamental importance both from a theoretical and a practical point of view, we now proceed to tackle it. We will show how the question can be answered positively by the use of a new cryptographic primitive called "signcryption" whose definition follows.

Intuitively, a digital *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 (informal) terminology in cryptography, it consists of a pair of (polynomial time) algorithms (S, U), where S is called the *signcryption algorithm*, while 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, 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.

A direct consequence of having to satisfy both the second and third requirements is that "signcryption $\neq$ signature-then-encryption". These two requirements also justify our decision to introduce the new word *signcryption* which clearly indicates the ability for the new approach to achieve both the functions of digital signature and secure encryption in a logically single operation.

The rest of this section is devoted to seeking for concrete implementations of signcryption. We first identify two (types of) efficient ElGamal-based signature schemes. Then we show how to use a common property of these schemes to construct signcryption schemes.

3.1 Shortening ElGamal-Based Signatures

ElGamal digital signature scheme [9] involves two parameters public to all users: (1) p — a large prime, and (2) g — an integer in $[1, \ldots, p-1]$ with order

$p-1$ modulo p. User Alice's secret key is an integer x_a chosen randomly from $[1,\ldots,p-1]$ with $x_a \not| (p-1)$ (i.e., x_a does not divide $p-1$), and her public key is $y_a = g^{x_a} \bmod p$.

Alice's signature on a message m is composed of two numbers r and s:

$$r = g^x \bmod p$$
$$s = (hash(m) - x_a \cdot r)/x \bmod (p-1)$$

where $hash$ is a one-way hash function, and x is chosen independently at random from $[1,\ldots,p-1]$ with $x \not| (p-1)$ every time a message is to be signed by Alice. Given (m,r,s), one can verify whether (r,s) is Alice's signature on m by checking whether $g^{hash(m)} = y_a^r \cdot r^s \bmod p$ is satisfied.

Since its publication in 1985, ElGamal signature has received extensive scrutiny by the research community. In addition, it has been generalized and adapted to numerous different forms (see for instance [23, 4, 18, 20] and especially [11] where an exhaustive survey of some 13000 ElGamal based signatures has been carried out.) Two notable variants of ElGamal signature are Schnorr signature [23] and DSS or Digital Signature Standard [18]. With DSS, g is an integer in $[1,\ldots,p-1]$ with order q modulo p, where q is a large prime factor of $p-1$. Alice's signature on a message m is composed of two numbers r and s which are defined as

$$r = (g^x \bmod p) \bmod q$$
$$s = (hash(m) + x_a \cdot r)/x \bmod q$$

where x is a random number chosen from $[1,\ldots,q-1]$. Given (m,r,s), one accepts (r,s) as Alice's valid signature on m if $(g^{hash(m)/s} \cdot y_a^{r/s} \bmod p) \bmod q = r$ is satisfied.

For most ElGamal based schemes, the size of the signature (r,s) on a message is $2|p|$, $|q|+|p|$ or $2|q|$, where p is a large prime and q is a prime factor of $p-1$. The size of an ElGamal based signature, however, can be reduced by using a modified "seventh generalization" method discussed in [11]. In particular, we can change the calculations of r and s as follows:

1. Calculation of r — Set $r = hash(k,m)$, where $k = g^x \bmod q$ ($k = g^x \bmod (p-1)$ if the original r is calculated modulo $(p-1)$), x is a random number from $[1,\ldots,q]$ (or from $[1,\ldots,p-1]$ with $x \not| (p-1)$), and $hash$ is a one-way hash function such as Secure Hash Standard or SHS [19].
2. Calculation of s — For an *efficient* ElGamal based signature scheme, the calculation of (the original) s from x_a, x, r and optionally, $hash(m)$ involves only simple arithmetic operations, including modulo addition, subtraction, multiplication and division. Here we assume that x_a is the secret key of Alice the message originator. Her matching public key is $y_a = g^{x_a} \bmod p$. We can modify the calculation of s in the following way:
   (a) If $hash(m)$ is involved in the original s, we replace $hash(m)$ with a number 1, but leave r intact. The other way may also be used, namely we change r to 1 and then replace $hash(m)$ with r.

(b) If s has the form of $s = (\cdots)/x$, then changing it to $s = x/(\cdots)$ does not add additional computational cost to signature generation, but may reduce the cost for signature verification.

To verify whether (r, s) is Alice's signature on m, we recover $k = g^x \bmod p$ from r, s, g, p and y_a and then check whether $hash(k, m)$ is identical to r.

To illustrate how to shorten ElGamal based signatures, now we consider DSS. It should be stressed that many other ElGamal based signature schemes, in particular those defined on a sub-group of order q (see for example [11, 20]), can be shortened in the same way and are all equally good candidates for signcryption. Table 1 shows two shortened versions of DSS, which are denoted by SDSS1 and SDSS2 respectively. Here are a few remarks on the table: (1) the first letter "S" in the name of a scheme stands for "shortened", (2) the parameters p, q and g are the same as those for DSS, (3) x is a random number from $[1, \ldots, q]$, x_a is Alice's secret key and $y_a = g^{x_a} \bmod p$ is her matching public key, (4) $|t|$ denotes the size or length (in bits) of t, (5) the schemes have the same signature size of $|hash(\cdot)| + |q|$, (6) SDSS1 is slightly more efficient than SDSS2 in signature generation, as the latter involves an extra modulo multiplication.

Recently Pointcheval and Stern [22] have proven that Schnorr signature is unforgeable by any adaptive attacker who is allowed to query Alice's signature generation algorithm with messages of his choice [10], in a model where the one-way hash function used in the signature scheme is assumed to behave like a random function (the random oracle model). The core idea behind the unforgeability proof by Pointcheval and Stern is based on an observation that the signature has been converted from a 3-move zero-knowledge protocol (for proof of knowledge) with respect to a honest verifier. With such a signature scheme, unforgeability against a non-adaptive attacker who is not allowed to possess valid message-signature pairs follows from the soundness of the original protocol. Furthermore, as the protocol is zero-knowledge with respect to a honest verifier, the signature scheme converted from it can be efficiently simulated in the random oracle model. This implies that an adaptive attacker is not more powerful than a non-adaptive attacker in the random oracle model.

Turning our attention to SDSS1 and SDSS2, both can be viewed as being converted from a 3-move zero-knowledge protocol (for proof of knowledge) with respect to a honest verifier. Thus Pointcheval and Stern's technique is applicable also to SDSS1 and SDSS2. Summarizing the above discussions, both SDSS1 and SDSS2 are unforgeable by adaptive attackers, under the assumptions that discrete logarithm is hard and that the one-way hash function behaves like a random function.

3.2 Implementing Signcryption with Shortened Signature

An interesting characteristic of a shortened ElGamal based signature scheme obtained in the method described above is that although $g^x \bmod p$ is not explicitly contained in a signature (r, s), it can be recovered from r, s and other public parameters. This motivates us to construct a signcryption from a shortened signature scheme.

Shortened schemes	Signature (r, s) on a message m	Recovery of $k = g^x \bmod p$	Length of signature
SDSS1	$r = hash(g^x \bmod p, m)$ $s = x/(r + x_a) \bmod q$	$k = (y_a \cdot g^r)^s \bmod p$	$\|hash(\cdot)\| + \|q\|$
SDSS2	$r = hash(g^x \bmod p, m)$ $s = x/(1 + x_a \cdot r) \bmod q$	$k = (g \cdot y_a^r)^s \bmod p$	$\|hash(\cdot)\| + \|q\|$

p: a large prime (public to all),
q: a large prime factor of $p - 1$ (public to all),
g: a (random) integer in $[1, \ldots, p-1]$ with order q modulo p (public to all),
$hash$: a one-way hash function (public to all),
x_a: Alice's secret key,
y_a: Alice's public key ($y_a = g^{x_a} \bmod p$).

Table 1. Examples of Shortened and Efficient Signature Schemes

We exemplify our construction method using the two shortened signatures in Table 1. The same construction method is applicable to other shortened signature schemes based on ElGamal. As a side note, Schnorr's signature scheme, without being further shortened, can be used to construct a signcryption scheme which is slightly more advantageous in computation than other signcryption schemes from the view point of a message originator.

In describing our method, we will use E and D to denote the encryption and decryption algorithms of a private key cipher such as DES [17] and SPEED [25]. Encrypting a message m with a key k, typically in the cipher block chaining or CBC mode, is indicated by $E_k(m)$, while decrypting a ciphertext c with k is denoted by $D_k(c)$. In addition we use $KH_k(m)$ to denote hashing a message m with a key-ed hash algorithm KH under a key k. An important property of a key-ed hash function is that, just like a one-way hash function, it is computationally infeasible to find a pair of messages that are hashed to the same value (or collide with each other). This implies a weaker property that is sufficient for signcryption: given a message m_1, it is computationally intractable to find another message m_2 that collides with m_1. In [2] two methods for constructing a cryptographically strong key-ed hash algorithm from a one-way hash algorithm have been demonstrated. For most practical applications, it suffices to define $KH_k(m) = hash(k, m)$, where $hash$ is a one-way hash algorithm.

Assume that Alice also has chosen a secret key x_a from $[1, \ldots, q]$, and made public her matching public key $y_a = g^{x_a} \bmod p$. Similarly, Bob's secret key is x_b and his matching public key is $y_b = g^{x_b} \bmod p$.

The signcryption and unsigncryption algorithms constructed from a shortened signature are remarkably simple. For Alice to signcrypt a message m for Bob, she carries out the following:

Signcryption by Alice the Sender

1. Pick x randomly from $[1, \ldots, q]$, and let $k = y_b^x \bmod p$. Split k into k_1 and k_2 of appropriate length. (Note: one-way hashing, or even simple folding, may be applied to k prior splitting, if k_1 or k_2 is too long to fit in E or KH, or one wishes k_1 and k_2 to be dependent on all bits in k.)
2. $r = KH_{k_2}(m)$.
3. $s = x/(r + x_a) \bmod q$ if SDSS1 is used, or
   $s = x/(1 + x_a \cdot r) \bmod q$ if SDSS2 is used instead.
4. $c = E_{k_1}(m)$.
5. Send to Bob the signcrypted text (c, r, s).

The unsigncryption algorithm works by taking advantages of the property that $g^x \bmod p$ can be recovered from r, s, g, p and y_a by Bob. On receiving (c, r, s) from Alice, Bob unsigncrypts it as follows:

Unsigncryption by Bob the Recipient

1. Recover k from r, s, g, p, y_a and x_b:
   $k = (y_a \cdot g^r)^{s \cdot x_b} \bmod p$ if SDSS1 is used, or
   $k = (g \cdot y_a^r)^{s \cdot x_b} \bmod p$ if SDSS2 is used.
2. Split k into k_1 and k_2.
3. $m = D_{k_1}(c)$.
4. accept m as a valid message originated from Alice only if $KH_{k_2}(m)$ is identical to r.

In the following, the two examples of signcryption schemes will be denoted by SCS1 and SCS2 respectively. For the purpose of a detailed comparison, the cost of these signcryption schemes has been analyzed and listed, along with other signature-then encryption schemes, in Table 2.

Finally two remarks follow: (1) signcryption schemes can also be derived from shortened signature schemes based on the discrete logarithm problem on elliptic curves [13]. (2) the functions, especially non-repudiation and unforgeability, of signcryption may not be fully implemented by the use of a shared key between Alice and Bob, such as $g^{x_a \cdot x_b} \bmod p$ or a key obtained via a Key Pre-distribution Scheme [16], unless tamper-resistant devices and/or trusted third parties are involved.

3.3 Working with Signature-Only and Encryption-Only Modes

Not all messages require both confidentiality and integrity. Some messages may need to be signed only, while others may need to be encrypted only. For the two digital signcryption schemes SCS1 and SCS2, when a message is sent in clear, they degenerate to signature schemes with verifiability by the recipient

only. As will be argued in Section 6, limiting verifiability to the recipient only still preserves non-repudiation, and may represent an advantage for some applications where the mere fact that a message is originated from Alice needs to be kept secret. Furthermore, if Alice uses g instead of Bob's public key y_b in the calculation of k, the schemes becomes corresponding shortened ElGamal based signature schemes with universal verifiability.

To work with the encryption-only mode, one may simply switch to the ElGamal encryption, or any other public key encryption scheme.

Various schemes	Computational cost	Communication overhead (in bits)
signature-then-encryption based on RSA	EXP=2, HASH=1, ENC=1 [EXP=2, HASH=1, DEC=1]	$\|n_a\| + \|n_b\|$
signature-then-encryption based on DSS + ElGamal encryption	EXP=3, MUL=1, DIV=1 ADD=1, HASH=1, ENC=1 [EXP=3, MUL=1, DIV=2 ADD=0, HASH=1, DEC=1]	$2\|q\| + \|p\|$
signature-then-encryption based on Schnorr signature + ElGamal encryption	EXP=3, MUL=1, DIV=0 ADD=1, HASH=1, ENC=1 [EXP=3, MUL=1, DIV=0 ADD=0, HASH=1, DEC=1]	$\|KH.(\cdot)\| + \|q\| + \|p\|$
signcryption SCS1	EXP=1, MUL=0, DIV=1 ADD=1, HASH=1, ENC=1 [EXP=2, MUL=2, DIV=0 ADD=0, HASH=1, DEC=1]	$\|KH.(\cdot)\| + \|q\|$
signcryption SCS2	EXP=1, MUL=1, DIV=1 ADD=1, HASH=1, ENC=1 [EXP=2, MUL=2, DIV=0 ADD=0, HASH=1, DEC=1]	$\|KH.(\cdot)\| + \|q\|$

where
EXP = the number of modulo exponentiations,
MUL = the number of modulo multiplications,
DIV = the number of modulo division (inversion),
ADD = the number of modulo addition or subtraction,
HASH = the number of one-way or key-ed hash operations,
ENC = the number of encryptions using a private key cipher,
DEC = the number of decryptions using a private key cipher,
Parameters in the brackets indicate the number of operations involved in "decryption-then-verification" or "unsigncryption".

Table 2. Cost of Signature-Then-Encryption v.s. Cost of Signcryption

4 Cost of Signcryption v.s. Cost of Signature-Then-Encryption

The most significant advantage of signcryption over signature-then-encryption lies in the dramatic reduction of computational cost and communication overhead which can be symbolized by the following inequality:

$$\text{Cost(signcryption)} < \text{Cost(signature)} + \text{Cost(encryption)}.$$

With SCS1 and SCS2, this advantage is shown in Tables 3 and 4.

Note that when comparing with RSA based signature-then-encryption, we have assumed that a relatively small public exponent e is employed for encryption or signature verification, although cautions should be taken in light of recent progress in cryptanalysis against RSA with an small exponent (see for example [6]). Therefore the main computational cost for RSA based signature-then-encryption is in decryption or signature generation which generally involves a modulo exponentiation with a *full size* exponent d. We have further assumed that the Chinese Remainder Theorem is used, so that the computational expense for RSA decryption can be reduced, theoretically, to a quarter of the expense with a full size exponent.

security parameters $\lvert p\rvert, \lvert q\rvert, \lvert KH.(\cdot)\rvert (= \lvert hash(\cdot)\rvert)$	saving in comp. cost	saving in comm. overhead
768, 152, 80	50%	76.8%
1024, 160, 80	50%	81.0%
2048, 192, 96	50%	87.7%
4096, 256, 128	50%	91.0%
8192, 320, 160	50%	94.0%
10240, 320, 160	50%	96.0%

$$\text{saving in comp. cost} = \frac{\text{3 modulo exponentiations}}{\text{6 modulo exponentiations}} = 50\%$$

$$\text{saving in comm. cost} = \frac{\lvert hash(\cdot)\rvert+\lvert q\rvert+\lvert p\rvert-(\lvert KH.(\cdot)\rvert+\lvert q\rvert)}{\lvert hash(\cdot)\rvert+\lvert q\rvert+\lvert p\rvert}$$

Table 3. Saving of Signcryption over Signature-Then-Encryption Using Schnorr Signature and ElGamal Encryption

4.1 How the Parameters are Chosen

Advances in fast computers help an attacker in increasing his capability to break a cryptosystem. To compensate this, larger security parameters, including $|n_a|$, $|n_b|$, $|p|$, $|q|$ and $|KH.(\cdot)|$ must be used in the future. From an analysis by Odlyzko [21] on the hardness of discrete logarithm, one can see that unless there is an algorithmic breakthrough in solving the factorization or discrete logarithm

security parameters $\lvert p\rvert(=\lvert n_a\rvert=\lvert n_b\rvert), \lvert q\rvert, \lvert KH.(\cdot)\rvert$	advantage in comp. cost	advantage in comm. overhead
768, 152, 80	0%	84.9%
1024, 160, 80	6.25%	88.3%
2048, 192, 96	43.8%	93.0%
4096, 256, 128	62.0%	95.0%
8192, 320, 160	77.0%	97.0%
10240, 320, 160	81.0%	98.0%

$$\text{advantage in comp. cost} = \frac{0.375(|n_a|+|n_b|)-4.5|q|}{0.375(|n_a|+|n_b|)}$$

$$\text{advantage in comm. cost} = \frac{|n_a|+|n_b|-(|KH.(\,)|+|q|)}{|n_a|+|n_b|}$$

Table 4. Advantage of Signcryption over RSA based Signature-Then-Encryption with *Small Public Exponents*

problem, $|q|$ and $|KH.(\cdot)|$ can be increased at a smaller pace than can $|n_a|$, $|n_b|$ and $|p|$. Thus, as shown in Tables 3 and 4, the saving or advantage in computational cost and communication overhead by signcryption will be more significant in the future when larger parameters must be used.

The selection of security parameters $|p|$, $|q|$, $|n_a|$ and $|n_a|$ in Tables 3 and 4, has been partially based on recommendations made in [21]. The parameter values in the tables, however, are indicative only, and can be determined flexibly in practice. We also note that choosing $|KH.(\cdot)| \approx |q|/2$ is due to the fact that using Shank's baby-step-giant-step or Pollard's rho method, the complexity of computing discrete logarithms in a sub-group of order q is $O(\sqrt{q})$ (see [14]). Hence choosing $|KH.(\cdot)| \approx |q|/2$ will minimize the communication overhead of the signcryption schemes SCS1 and SCS2. Alternatively, one may decide to choose $KH.(\cdot) \in [1, \ldots, q]$ which can be achieved by setting $|KH.(\cdot)| = |q| - 1$. This will not affect the computational advantage of the signcryption schemes, but slightly increase their communication overhead.

5 Applications of Signcryption

As discussed in the introduction, a major motivation of this work is to search for a more economical method for secure and authenticated transactions/message delivery. If digital signcryptions are applied in this area, the resulting benefits are potentially significant: for every single secure and authenticated electronic transaction, we may save 50% in computational cost and 85% in communication overhead.

The proposed signcryption schemes are compact and particularly suitable for smart card based applications. We envisage that they will find innovative applications in many areas including digital cash payment systems, EDI and personal heath cards. Of particular importance is the fact that signcryption

may be used to design more efficient digital cash transaction protocols that are often required to provide with both the functionality of digital signature and encryption.

In the full paper we also show how to adapt a signcryption scheme into one for broadcast communication which involves multiple recipients. Such an adapted scheme shares a comparable computational cost with a broadcast scheme proposed in RFC1421. The communication overhead required by the scheme based on signcryption, however, is multiple times lower than that required the scheme in RFC1421.

Another surprising property of the proposed signcryption schemes is that it enables us to carry out fast, secure, unforgeable and non-repudiatable key transport *in a single block whose size is smaller than* $|p|$. In particular, using either of the two signcryption schemes, we can transport highly secure and authenticated keys in a single ATM cell (48 byte payload + 5 byte header). A possible combination of parameters is $|p| \geq 512$, $|q| = 160$, and $|KH.(\cdot)| = 80$, which would allow the transport of an unforgeable and non-repudiatable key of up to 144 bits. Advantages of such a key transport scheme over interactive key exchange protocols such as those proposed in [8] are obvious, both in terms of computational efficiency and compactness of messages. Compared with previous attempts for secure, but un-authenticated, key transport based on RSA (see for example [1, 12]), our key transport scheme has a further advantage in that it offers both unforgeability and non-repudiation. In a similar way, a multi-recipient signcryption scheme can be used as a very economical method for generating conference keys among a group of users.

6 Unforgeability, Non-repudiation and Confidentiality of Signcryption

Like any cryptosystem, security of signcryption in general has to address two aspects: (1) to protect what, and (2) against whom. With the first aspect, we wish to prevent the contents of a signcrypted message from being disclosed to a third party other than Alice, the sender, and Bob, the recipient. At the same time, we also wish to prevent Alice, the sender, from being masquerade by other parties, including Bob. With the second aspect, we consider the most powerful attackers one would be able to imagine in practice, namely adaptive attackers who are allowed to have access to Alice's signcryption algorithm and Bob's unsigncryption algorithm.

We say that a signcryption scheme is secure if the following conditions are satisfied:

1. Unforgeability — it is computationally infeasible for an adaptive attacker (who may be a dishonest Bob) to masquerade Alice in creating a signcrypted text.
2. Non-repudiation — it is computationally feasible for a third party to settle a dispute between Alice and Bob in an event where Alice denies the fact that she is the originator of a signcrypted text with Bob as its recipient.

3. Confidentiality — it is computationally infeasible for an adaptive attacker (who may be any party other than Alice and Bob) to gain any partial information on the contents of a signcrypted text.

A detailed description of the proofs/arguments of the security of the signcryption schemes SCS1 and SCS2 can be found in the full paper. Here are the key ideas used in the proofs/arguments:

1. Unforgeability — this can be done using the technique of Pointcheval and Stern [22].
2. Non-repudiation — A dispute between Alice and Bob can be settled by a trusted third party (say a judge), by the use of a zero-knowledge proof protocol between the judge and Bob. In particular, they can use a very simple 4-move zero-knowledge interactive proof protocol proposed by Chaum in [5].
3. Confidentiality — We achieve our goal by reduction: we will reduce the confidentiality of another encryption scheme called C_{kh}, whose confidentiality is relatively well-understood, to the confidentiality of a signcryption scheme (say SCS1). With the encryption scheme C_{kh}, the ciphertext of a message m is defined as ($u = g^x \bmod p$, $c = E_{k_1}(m)$, $r = KH_{k_2}(m)$) where k_1 and k_2 are defined in the same way as in SCS1. C_{kh} is a slightly modified version of a scheme that has received special attention in [24, 3] (see also earlier work [26].)
   Now assume that there is an attacker for SCS1. Call this attacker A_{SCS1}. We show how A_{SCS1} can be translated into one for C_{kh}, called $A_{C_{kh}}$. Note that for a message m, the input to A_{SCS1} includes q, p, g, $y_a = g^{x_a} \bmod p$, $y_b = g^{x_b} \bmod p$, $u = g^x \bmod p$, $c = E_{k_1}(m)$, $r = KH_{k_2}(m)$. With the attacker $A_{C_{kh}}$ for C_{kh}, however, its input includes: q, p, g, $y_b = g^{x_b} \bmod p$, $u = g^x \bmod p$, $c = E_{k_1}(m)$, and $r = KH_{k_2}(m)$. One immediately identifies that two numbers that correspond to y_a and s which are needed by A_{SCS1} as part of its input are currently missing from the input to $A_{C_{kh}}$. Thus, in order for $A_{C_{kh}}$ to "call" the attacker A_{SCS1} "as a sub-routine", $A_{C_{kh}}$ has to create two numbers corresponding to y_a and s in the input to A_{SCS1}. Call these two yet-to-be-created numbers y'_a and s'. y'_a and s' have to have the right form so that $A_{C_{kh}}$ can "fool" A_{SCS1}. It turns out that such y'_a and s' can be easily created by $A_{C_{kh}}$ as follows: (1) pick a random number s' from $[1, \ldots, q]$. (2) let $y'_a = u^{1/s'} \cdot g^{-r} \bmod p$.

A final note on signcryption follows. Unlike signature-then-encryption, the verifiability of a signcryption is in normal situations limited to Bob the recipient, as his secret key is required for unsigncryption. At the first sight, the limited verifiability of a signcryption, namely the direct verifiability by the sender only (and indirect verifiability by a judge with the cooperation of Bob), may be seen as a drawback of signcryption. Here we argue that the limited direct verifiability will not pose any problem in practice and hence should not be an obstacle to practical applications of signcryption. In the real life, a message sent to Bob in a secure and authenticated way is meant to be readable by Bob only. Thus if there is no dispute between Alice and Bob, direct verifiability by Bob only is

precisely what the two users want. In other words, in normal situations where no disputes between Alice and Bob occur, the full power of universal verifiability provided by digital signature is never needed. (For a similar reason, traditionally one uses signature-then-encryption, rather than encryption-then-signature !) In a situation where repudiation does occur, interactions between Bob and a judge would follow. This is very similar to a dispute on repudiation in the real world, say between a complainant (Bob) and a defendant (Alice), where the process for a judge to resolve the dispute requires in general interactions between the judge and the complainant, and furthermore between the judge and an expert in hand-written signature identification, as the former may rely on advice from the latter in correctly deciding the origin of a message.

7 Conclusion

We have introduced a new cryptographic primitive called signcryption for secure and authenticated message delivery, which fulfills all the functions of digital signature and encryption, but with a far smaller cost than that required by the current standard signature-then-encryption methods. Security of the signcryption schemes has been proven, and extensions of the schemes to multiple recipients has been carried out. We believe that the new primitive will open up a number of avenues for future research into more efficient security solutions.

The signcryption schemes proposed in this paper have been based on ElGamal signature and encryption. We have not been successful in searching for a signcryption scheme employing RSA or other public key cryptosystems. Therefore it remains a challenging open problem to design signcryption schemes based factorization or other computationally hard problems.

References

1. Basturk, E., Bellare, M., Chow, C.-S., Guerin, R.: Secure transport protocols for high-speed networks. IBM Research Report Report RC 19981 IBM T. J. Watson Research Center Yorktown Heights, NY 10598 1994.
2. Bellare, M., Canetti, R., Krawczyk, H.: Keying hash functions for message authentication. In Advances in Cryptology - CRYPTO'96 (Berlin, New York, Tokyo, 1996) vol. 1109 of Lecture Notes in Computer Science Springer-Verlag pp. 1–15.
3. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In Proceedings of the First ACM Conference on Computer and Communications Security (New York, November 1993) The Association for Computing Machinery pp. 62–73.
4. Brickell, E., McCurley, K.: Interactive identification and digital signatures. AT&T Technical Journal (1991) 73–86.
5. Chaum, D.: Zero-knowledge undeniable signatures. In Advances in Cryptology - EUROCRYPT'90 (Berlin, New York, Tokyo, 1990) vol. 473 of Lecture Notes in Computer Science Springer-Verlag pp. 458–464.
6. Coppersmith, D., Franklin, M., Patarin, J., Reiter, M.: Low-exponent RSA with related messages. In Advances in Cryptology - EUROCRYPT'96 (Berlin, 1996) vol. 1070 of Lecture Notes in Computer Science Springer-Verlag pp. 1–9.

7. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Transactions on Information Theory **IT-22** (1976) 472–492.
8. Diffie, W., Oorschot, P. V., Wiener, M.: Authentication and authenticated key exchange. Designs, Codes and Cryptography **2** (1992) 107–125.
9. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory **IT-31** (1985) 469–472.
10. Goldwasser, S., Micali, S., Rivest, R.: A digital signature scheme secure against adaptively chosen message attacks. SIAM J. on Computing **17** (1988) 281–308.
11. Horster, P., Michels, M., Petersen, H.: Meta-ElGamal signature schemes. In Proceedings of the second ACM Conference on Computer and Communications Security (New York, November 1994) ACM pp. 96–107.
12. Johnson, D., Matyas, S.: Asymmetric encryption: Evolution and enhancements. CryptoBytes **2** (1996) 1–6.
13. Koblitz, N.: Elliptic curve cryptosystems. Mathematics of Computation **48** (1987) 203–209.
14. Lenstra, A. K., Lenstra, H. W.: Algorithms in Number Theory vol. A of Handbook in Theoretical Computer Science. Elsevier and the MIT Press 1990.
15. Linn, J.: Privacy enhancement for internet electronic mail: Part I: Message encryption and authentication procedures. Request for Comments RFC 1421 IAB IRTF PSRG, IETF PEM WG 1993.
16. Matsumoto, T., Imai, H.: On the key predistribution systems: A practical solution to the key distribution problem. In Advances in Cryptology - CRYPTO'87 (Berlin, New York, Tokyo, 1987) vol. 239 of Lecture Notes in Computer Science Springer-Verlag pp. 185–193.
17. National Bureau of Standards: Data encryption standard. FIPS PUB 46 U.S. Department of Commerce January 1977.
18. National Institute of Standards and Technology: Digital signature standard (DSS). FIPS PUB 186 U.S. Department of Commerce May 1994.
19. National Institute of Standards and Technology: Secure hash standard. FIPS PUB 180-1 U.S. Department of Commerce April 1995.
20. Nyberg, K., Rueppel, R.: Message recovery for signature schemes based on the discrete logarithm problem. Designs, Codes and Cryptography **7** (1996) 61–81.
21. Odlyzko, A.: The future of integer factorization. CryptoBytes **1** (1995) 5–12.
22. Pointcheval, D., Stern, J.: Security proofs for signature schemes. In Advances in Cryptology - EUROCRYPT'96 (Berlin, New York, Tokyo, 1996) vol. 1070 of Lecture Notes in Computer Science Springer-Verlag pp. 387–398.
23. Schnorr, C. P.: Efficient identification and signatures for smart cards. In Advances in Cryptology - CRYPTO'89 (Berlin, New York, Tokyo, 1990) vol. 435 of Lecture Notes in Computer Science Springer-Verlag pp. 239–251.
24. Zheng, Y.: Improved public key cryptosystems secure against chosen ciphertext attacks. Technical Report 94-1 University of Wollongong Australia January 1994.
25. Zheng, Y.: The SPEED cipher. In Proceedings of Financial Cryptography'97 (Berlin, New York, Tokyo, 1997) Lecture Notes in Computer Science Springer-Verlag.
26. Zheng, Y., Seberry, J.: Immunizing public key cryptosystems against chosen ciphertext attacks. IEEE Journal on Selected Areas in Communications **11** (1993) 715–724.