

## DIGITAL MULTISIGNATURES

C. Boyd  
(British Telecom, Ipswich)

### 1. INTRODUCTION

The idea of digital signature is now well known as a means of replacing written signatures in electronic communications. In many applications a document may need to be signed by more than one party [ 2]. When a signature depends on more than one person we will call it a multisignature.

One way of achieving a multisignature is by using a so-called 'threshold scheme' [ 7] which divides knowledge of the signing key between several people. However in this case all the signatories must meet together at the same place and at the same time in order for the key to be re-assembled, which is clearly impractical in many applications.

Another possibility is successive signing by the various parties, when the signature is verified by removing each signature in turn. This has certain disadvantages however. Firstly the time taken for verification is multiplied by the number of signatories. Secondly, in the RSA signature scheme, which is the most popular scheme, reblocking will be necessary when a subsequent signatory has a larger modulus. Thirdly, any verifier will need to have the relevant verification information for each signatory involved.

In this paper we examine ways of adapting the RSA scheme [ 6] to enable multisignatures to be easily implemented by groups of users such that signature verification will only require a single RSA transformation. The suggested schemes would be particularly applicable for use in companies. Typical applications might be the signing of documents by business partners, or signing of cheques for Electronic Funds Transfer where two or more company representatives may be required to sign any cheque.

In [4] a way of implementing a multisignature scheme using RSA was described. This was essentially a way of overcoming the re-blocking problem mentioned above by assigning an individual's modulus according to company seniority, so that a superior would always have a larger modulus than those under him. This scheme however is only useful when the order of signing is pre-determined (the application envisaged in [4] was the approval of signed documents by a superior). It still suffers from the other drawbacks mentioned above. The idea behind the schemes suggested here is to extend the RSA algorithm to a multiple-key cipher. It is suggested in [1] that for multiple-key ciphers the keys must be inserted simultaneously. We use the multiplicative property of exponentiation to invent a scheme where this is not necessary.

The schemes proposed in this paper are not in fact limited to the RSA algorithm: Any encryption algorithm  $E$  with the multiplicative property  $E(k_1).E(k_2) = E(k_1.K_2)$  will do. For concreteness, however, we will use RSA throughout the paper.

## 2. A SCHEME FOR DOUBLE SIGNATURES

The scheme described here enables two users to sign, say, a cheque which can be verified by any user. The idea is to extend the RSA scheme by having three keys instead of two: two private keys and one public key. The company chooses a modulus  $n$  to be the product of two large primes as in the RSA scheme. The two private keys are then chosen at random in the range 1 to  $n$ , subject to the condition that they are coprime to  $\phi(n)$ . Call these keys  $r$  and  $s$ . The public key  $t$  is then chosen so that

$$r.s.t = 1 \pmod{\phi(n)};$$

$r$  and  $s$  are now issued to the authorised signatories and  $t$  is made public. In order to sign the cheque  $C$  the first signatory calculates

$$S_1 = C^{**r} \pmod{n}$$

and sends  $S_1$  to the second signatory. The second signatory can now recover  $C$  from  $S_1$  in order to see what he is to sign by

$$C = S_1^{**(s.t)} \pmod{n}$$

since he knows both  $s$  and  $t$ . If he is satisfied he now signs  $S_1$  to form

$$S_2 = S_1^{**s} \pmod{n}$$

and sends  $S_2$  to the recipient. Since  $t$  is public, the recipient and any member of the public can verify the validity of the cheque by calculating

$$C = S2^{**t} \text{ mod } n.$$

The cheque must have been signed by the two authorised signatories in order to form S2. Note that the order of signing in this scheme does not matter.

Knowledge of S1 is of no use to (and cannot even be read by) an imposter. It seems clear that forging a message for this scheme is the same as forging an RSA signature with secret key  $r*s$ , and in this sense the scheme is just as secure as the RSA algorithm. However from a key management viewpoint it is certainly safer to divide the key into two parts and keep them separately. There are many applications where dividing the responsibility for authorisation between two people is required. For example, in the SWIFT system two people must authorise each transaction.

The multiplicative property exploited in this scheme can also be used to attack RSA signatures in certain circumstances, see [5]. For example, since  $M1^{**r}.M2^{**r} = (M1.M2)^{**r}$  the signature of M1.M2 can be deduced from those of M1 and M2. Various means are available to avoid these attacks, and they are also applicable in this scheme. One way is to use a one-way hash function  $h$  to transform the message prior to signing, so that  $h(M1.M2) = h(M1).h(M2)$ . This also has the advantage that only one block need be signed however long the message is.

### 3. EXTENDING THE SCHEME

In certain situations the number of possible signatories may be many. For example in a company any two of a number of authorised signatories may be allowed. At the cost of some extra storage and processing the above scheme may be extended to take account of this. If there are  $N$  potential signatories then  $n$  random secret keys,  $k1, k2, \dots, kN$  are chosen. The public key  $t$  is chosen so that

$$k1.k2. \dots .kN.t = 1 \text{ mod } \phi(n).$$

Each signatory is then issued with all the private keys except one. For example the  $j$ -th signatory is given all  $ki$ 's except  $kj$ . Each signatory stores all these keys and also their product. Let  $kj' = k1 \dots k(j-1)k(j+1) \dots kn$ . When the  $j$ -th signatory wishes to sign a cheque  $C$  he signs it to form

$$S1 = C^{**kj'} \text{ mod } n.$$

and appends his identity. Any other signatory can then complete the signing by looking up the missing key, which also allows him to read the message and then to form

$$S2 = S1^{**kj} \text{ mod } n.$$

The recipient and any member of the public can again verify the signature by one decipherment with  $t$ .

#### 4. BLIND MULTISIGNATURES

In certain situations more than two signatories will be required. The above scheme cannot be extended to achieve this. This is because if more than two signatories have secret keys the second signatory (and all others before the last) are unable to read what it is that they are signing. In other words they are only able to create 'blind signatures'.

There are many applications where blind signatures can prove very useful, as discussed in [3] and elsewhere. These include electronic bank notes and secret ballots. We explain here a way that blind multisignatures could be used.

In this scheme the players are a bank and a company. The first problem is to set up the keys. This is done as follows.

The bank chooses a modulus  $n = p \cdot q$ , a random private key  $b$ , and a random public key  $t$ . These are the same for all its customers. The amount debited is defined by the keys  $b$  and  $t$  chosen by the bank, and there can be various keys published by the bank for different denominations.

The company chooses two random keys  $k_1, k_2$ , coprime to  $n$ , and calculates

$$k = k_1 \cdot k_2.$$

The company then sends  $k$  to the bank, which returns a number  $k_3$  satisfying

$$b \cdot t \cdot k \cdot k_3 = 1 \pmod{\phi(n)}.$$

These keys can now be used to form electronic bank notes in the manner described in [3]. The novelty is that the keys  $k_1$  and  $k_2 \cdot k_3$  can be distributed to different people in the company. Let us rename these.

$$\begin{aligned} c_1 &= k_1, \\ c_2 &= k_2 \cdot k_3. \end{aligned}$$

If  $k_1$  and  $k_2$  are chosen to be random primes by the company then the bank has no knowledge of  $c_1$  and  $c_2$ . In order to obtain a single bank note the company chooses a random number  $r$ . This number must include some redundancy, say every other bit is 0. Some authorised member of the company (for example from the finance department) calculates  $r^{c_1} \pmod{n}$  and sends it to the bank. The bank returns  $(r^{c_1})^{b^{-1}}$  to the

company and debits the company's account by the appropriate amount. The finance department can then distribute the note to an authorised spender in possession of  $c_2$ .

When the spender wishes to use the note he validates it by forming  $((r^{**c_1})^{**b})^{**c_2} \bmod m$ . The retailer can verify that it is genuine by recovering  $r$  and checking the redundancy condition. The retailer sends the note to the bank to be cleared where it is checked against, and added to, a list of cleared notes.

Although this procedure puts a certain amount of trust in the bank it allows the company to use its money in a way untraceable by the bank. The company has much better control over its money by dividing the responsibility for forming and for spending the bank notes. At the same time if the bank or the retailer allows access to its records of cleared notes, the company's finance department can audit the use of its notes by checking which random numbers were issued to which spender.

#### 5. ANOTHER SCHEME

In this final section we present an alternative scheme to solve the problem of multisignatures with more than two signatories. However, it requires each signatory to sign separately and the results to be combined which may be inconvenient in many applications. In this scheme the company, or any issuing authority, again selects the secret keys at random. Suppose that there are three signatories required.  $k_1$ ,  $k_2$  and  $k_3$  are selected at random and this time the public key  $t$  is selected to satisfy

$$(k_1 + k_2 + k_3) \cdot t = 1 \bmod \phi(n).$$

Each signatory  $i$  takes the message  $M$  and signs it by

$$S_i = M^{**k_i} \bmod n.$$

The three signed copies are then multiplied by some central authority to form

$$S = S_1 \cdot S_2 \cdot S_3 \bmod n$$

which is then sent to the recipient. The recipient and any member of the public can verify the signature using  $t$  since

$$\begin{aligned} S^{**t} \bmod n &= (S_1 \cdot S_2 \cdot S_3)^{**t} \bmod n \\ &= M^{**[(k_1+k_2+k_3) \cdot t]} \bmod n \\ &= M. \end{aligned}$$

In the obvious way this system can be extended to enable any number of signatories to take part.

#### 6. ACKNOWLEDGEMENTS

I would like to thank E.J. Humphreys for valuable suggestions during the preparation of this paper. I also acknowledge the permission of the directors of British Telecom to publish this paper.

#### 7. REFERENCES

- [ 1] Carroll, J.M., (1984), "The Resurrection of Multiple-Key Ciphers", *Cryptologia*, July.
- [ 2] Chalton, S., (1986), "The Authentication of the Origin and Content of Paperless Transactions, and Questions of Liability in Common Law", *Proceedings of Conference on Paperless Trading and the Law in the EEC, Brussels, March.*
- [ 3] Chaum, D., (1982), "Blind Signatures for Untraceable Payments" *Proceedings of Crypto, Plenum Press.*
- [ 4] Itakura, K. and Nakamura, K., (1983), "A Public Key Cryptosystem Suitable for Digital Multisignatures", *NEC Research and Development*, 71, October.
- [ 5] de Jonge, W. and Chaum, D., (1985), "Attacks on Some RSA Signatures", *Crypto, Springer-Verlag.*
- [ 6] Rivest, R., Shamir, A. and Adelman, L., (1978), "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", *Comm. ACM* 21, 2, 120-126.
- [ 7] Shamir, A., (1979), "How to Share a Secret", *Comm. ACM* 22, 11, 612-613.