Digital Multi Signature Schemes

Premalatha A Grandhi (pgrandhi@cise.ufl.edu)

- Digital Signatures can be classified into
  - Single Signatures
  - Multiple Signatures (multi-signatures)
- Types of Multiple Signatures
  - Sequential Multi-signatures
  - Parallel/Simultaneous Multi-signatures

Parallel Multi-signatures:

In this scheme, the signature of each signer is on the content of the form but not on the signature of other signers. The mechanism of duplicating the form and distributing to each party is called *Fork.* It can be divided into two types.

- Fork-all - It needs distribution of he form to all the parties.
- Fork-some – It needs distribution of the form to only some of the relevant parties.

The mechanism to collect the signatures is called Join. It can be divided into two types.

- Join-all – Its mandatory that all the signatures are present and valid.
- Join-some – Only the signatures from the obligatory and those who satisfy the additional conditions are needed.

Sequential Multi-signatures:

In this scheme, the first signer signs the form and the second signer signs on the content of the form and the first signer's signature. The form is considered signed when the last signature is appended.

It can be distinguished as follows

- *Independent Sequential Multiple Signatures* where the sequence of signing in not important. The signers sign only on the content of the form.

- *Dependent Sequential Multiple Signatures* where the order of approval is important and has to be respected. The last signer has to sign not only on the content of the form but also the signatures of the form.

Different Modes of Digital Signature Schemes:

- **Appendix mode:** The digital signature is sent to the receiver along with the corresponding message. The message itself is not encrypted and will be used fro verification by the receiver. Example of digital signature scheme using appendix mode is ElGamal digital signature scheme based on discrete logarithms problem.

- **Recovery mode:** The signed message is embedded in the signature and can be recovered from the signature. The RSA digital signature scheme uses Recovery mode, which is based on the difficulty of factoring large integers.

Digital Multi-signatures based on RSA

It overcomes re-blocking problem by assigning individual's modulus according to company seniority. This scheme uses the multiplicative property of exponentiation

- ***Double Signature scheme:***

  It extends basic RSA scheme and generates three keys (one public and 2 private) in order for the cheque C to be signed by two signatories. The company chooses a modulus n to be the product of two large primes as in the RSA scheme. The two private keys r, s are chosen to be random in the range 1 to n such that they are co-prime to $\phi(n)$. The public key is chosen such that

  $$r * s * t = 1 \bmod \phi(n)$$

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

  $$S_1 = C^r \bmod n$$

  And sends S1 to the second signatory. The second signatory can now recover c form S1 in order to see what he is to sign by

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

  Since he knows s and t. If he is satisfied he now signs S1 to form

  $$S_2 = S_1^{s} \bmod n$$

  And sends S2 to the recipient. The recipient and any member of the public can verify the cheque's validity by calculating

  $$C = S_2^{t} \bmod n$$

- ***Extension of the scheme for multi-signatures:***

It can be extended to Multi-signatures by generating n random secret keys $k_1, k_2, ..k_n$ and a public key t such that

$$(k_1 + k_2 + ... + k_n) * t = 1 \bmod \phi(n)$$

Each signatory i takes the message M and signs it by

$$Si = M\, ki \bmod n$$

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

$$S = S_1 * S_2 * S_3 * ....*S_n \bmod n$$

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

$$S^t \bmod n = (S_1 * S_2 *. \ldots *S_n)^t \bmod n$$
$$= M^{[(k1+k2+k3+\ldots kn) * t]} \bmod n$$
$$= M$$

Thus it can be shown that the original message can be verified by any member of public / recipient by using just the public key t.

- The order of signing in this scheme doesn't matter.
- As this multi-signature scheme uses the basic RSA scheme, this scheme is as secure as RSA algorithm.
- Multiplicative property can be used to attack RSA based signatures and can be avoided by using a one-way hash function to transform the message before signing.
- One of the applications of such multi-signature scheme using RSA is the blind signature scheme, which can be implemented as explained as below.

Blind Signatures:

It allows a person to get a message signed by another party without revealing any information about the message to the other party.

- Suppose Alice has a message m that she wishes to have signed by Bob, and she does not want Bob to learn anything about m.
- Let (n,e) be Bob's public key and (n,d) be his private key.
- Alice generates a random value r such that $\gcd(r, n) = 1$ and sends $x = (r^e\ m) \bmod n$ to Bob.
- The value x is ``blinded'' by the random value r; hence Bob can derive no useful information from it. Bob returns the signed value $t = x^d \bmod n$ to Alice. Since

$$x^d \equiv (r^e\ m)^d \equiv r\ m^d \bmod n,$$

- Alice can obtain the true signature s of m by computing

$$s = r^{-1}\ t \bmod n.$$

Now Alice's message has a signature she could not have obtained on her own. This signature scheme is secure provided that factoring and root extraction remains difficult.

Digital Multi-signatures based on Discrete Logarithms Problem:

S_I issues the document D and S_1 , S_2 ,.., S_u are the signatories of the document and S_v is a trusted verifier who verifies the document D at the end of the signing process. Each signatory is assumed to hold a secret key x_i and a public key $y_i = g^{xi}$ (i=1,..,u). The Issuer has the secret key x_I and public key y_I and the Verifier has the secret key x_V and public key y_V . It involves three phases.

Phase 1:

- The issuer prepares the document and the list of the prospective signatories of the document.
- The issuer submits the document-list pair to the verifier who selects randomly x_d (secret) associated with the document-list, calculates $y_d = g^{xd}$ and broadcasts y_d
- On receiving the tag from the verifier, the issuer creates an authenticator incorporating the document-list pair and the tag.
- The issuer then sends a message to the first signatory containing the document, pad and other parameters required for authentication process by the signatory S1.

Phase 2:

- S1 tried to establish authenticity of the document-list pair and the integrity of the document-list and pad. If the document hasn't been tampered and is authentic, S1 signs the pad and forwards the document-list pair and pad to signatory S2.
- This process is repeated by each subsequent signatory S_i (i=2,..,n) with the last signatory S_u forwarding the document –list and pad to the verifier.

Phase 3:

- The trusted verifier receives the message and tries to establish its integrity.
- The verifier then checks for the authenticity and of the document list and checks whether all signatories have signed.
- Before starting the verification process, the verifier broadcasts x_d. Usinfg this value, all signatories and public can check whether $g^{xd}=y_d$

[2] describes this in mathematical terms and is straightforward regarding the explanation.

Digital Multi-signatures based on Fiat-Shamir Scheme

This is a Sequential Multi-signature scheme, which is efficient and proven under the difficulty of factoring a large composite number, that the probability of forgery against known or chosen message attack is $\mathbf{1/2^{kt}}$. Where **k** is the number of secret information integers stored in a signer and the signature size is proportional to **t**.

- The improvement of speed over RSA is about 20 times.
- This is an identity based multi-signature scheme.
- The order of signing is not restricted.
- The security level restricts the redundancy of signed message.

Basic Scheme:

Assume there are m users joining the signature system sign the same messages sequentially and convince the verifier that the checked message is signed by each user and hasn't been modified by an intruder. The message is denoted by P and the identification information of user i is ID_i .

The basic steps involved are

1. **Key Generation:** A trusted center publishes a modulus n , which is a product of two secret large primes p and q. It generates integers Sij (1<=j<=k) based on ID_i using a public one-way function f as follows.

$$Sij = 1/\sqrt{f(IDi, j)} \mod n \qquad \text{---------(1)}$$

It issues a smart card to user i after checking its physical identity, containing the set of (n, f, h, S_{i1},..,S_{ik}) where h is another public one-way function.

2. **Multi-signature Generation:**

To sign P , m users execute the following procedure.

1. Repeat while i =1 . . m
   a. Signer i receives X_{i-1}, where $X_0=1$ holds, generates a random integer $R \in Z_n$ where Z_n denotes {0, .. ,n-1}, calculates

$$X_i = R_i^2 * X_{i-1} \bmod n \qquad \text{-------(2)}$$

And sends it to the next signer (i+1) , where signer (m+1) is considered as signer 1.

2. Repeat while i = 1, . . , (m-1)
   a. Signer i receives (P, I_m, X_m, Y_{i-1}) from signer (i-1) where Y_0=1 holds, calculates

$$(e_1, .. , e_k) = h(P, I_m, X_m) \qquad \text{-------(3)}$$

And

$$Y_i = Y_{i-1} * R_i \prod_{ej-1} Sij \bmod n \qquad \text{-------(4)}$$

And sends (P, I_m, X_m, Y_m) to the next signer (i+1) where

$I_m = ID_1 * ID_2 * ID_3 * \ldots ID_m$

Where * is the concatenation operator.

3. Signer m receives (P, I_m, X_m, Y_{m-1}) from signer (m-1), calculates equations (3) and (4) and sends (P, I_m, (e_1,..e_k),Y_m) to a verifier.

3. Multi-signature Verification: A verifier verifies the message with the multi-signature (P, I_m, (e_1,..e_k),Y_m) from the signer m using public modulus n and one-way functions f and h as follows.

- The verifier calculates V_{ij} with I_m as follows

$$V_{ij} = f(ID_i, j) \qquad (1 \le i \le m)(1 \le j \le k)$$

- The verifier calculates Z_m with V_{ij} , (e_1,..e_k) and Y_m as follows

$$Z_m = Y_m^2 * \prod_{i=1}^{m} \prod_{ej=1} Vij \bmod n$$

- The verifier calculates h(P, I_m, Z_m) and checks whether the following equation holds

$$(e_1, .. e_k) = h(P, I_m, Z_m)$$

If the above equation holds, the multi-signature is considered to be valid.

Note:

If all m signers follow this procedure, a verifier will accept the multi-signature is valid

By definition,

$$S_{ij} = 1/Vij \bmod n \qquad (1 \le i \le m)(1 \le j \le k)$$

$$Z_m = (Y_{m-1}^2 * R_m^2 \prod_{ej=1} 1/Vmj) * \prod_{i=1}^{m} \prod_{ej=1} Vij \bmod n$$

$$= (Y_{m-1}^2 * \prod_{i=1}^{m-1} \prod_{ej=1} Vij) * R_m^2 \bmod n$$

$$= R_1^2 \ldots R_m^2$$

$$= X_m$$

And thus (e_1,..e_k) = (P, I_m, Z_m) = (P, I_m, X_m)

Here t is assumed to be 1 , but it can be greater than equal to 2, which is called Extended Fiat Shamir Scheme.

The security of Fiat-Shamir Scheme against passive attack is the same as the Fiat-Shamir Scheme

Bibliography:

1. Boyd, Colin. "Digital Signatures". Cryptography and Coding. H.J.Beker and F.C.Piper Eds., Oxford University Press, 1989, pp241-246. URL: http://sky.fit.qut.edu.au/~boydc/papers/ima89.pdf

2. Hardjono, Thomas; Zheng, Yulliang. "A Practical Multisignature Scheme Based on Discrete Logarithms". 1993. http://citeseer.nj.nec.com/hardjono93practical.html

3. Ohta, Okamoto, "A Digital Multisignature Scheme based on the Fiat-Shamir Scheme", ASICRYPT'91, pp139-148

4. Shieh, Shiuh-Pying; Lin, Chern-Tang; Yang, Wei-Bon; Sun, Hung-Min. "Digital Multisignature Schemes for Authenticating Delegates in Mobile Code Systems". July 2000.http://dsns.csie.nctu.edu.tw/ssp/docs/Digital%20multisignature%20schemes%20for%20authenticating%20delegates%20in%20mobile%20code%20systems.pdf

5. Rivest, Shamir and Adelman,L.,(1978), "A Method for Obtaining Digital MultiSignatures and Public Key CryptoSystems ", Comm.ACM21, 2 , 120-126