IN4191 Security and Cryptography
Assignment 2

Eric Camellini
4494164
e.camellini@student.tudelft.nl

9 October 2015

Abstract

In this report I analyse the identity-based RSA multisignature scheme proposed by L. Harn and J. Ren in [2]. I also provide an example of application of the algorithm.

1 Introduction

1.1 Multisignatures

A digital multisignature is a digital signature created by multiple people using multiple keys. Every person involved in the process has a personal key pair and uses the private key to build his/her part of the signature. The generated signature can then be verified using all the public keys of the signers. An important feature of multisignatures is that the signing group is not fixed, can be dynamically composed by any set of people. A good digital multisignature should satisfy two properties:

- Fixed length: the length of the signature is always the same, it doesn't depend from the number of signers;
- Constant verification time: the number of modulo exponentiations required to verify the multisignature is the same as the one required to verify a single signature.

In this report I analyse the multisignature scheme described in [2]: it was the first one to satisfy both the properties. This approach is based on the Shamir's IBS (identity-based signature) scheme, a digital RSA-based signature scheme where the identity of the signer is the public key. In an identity-based cryptosystem there is no need for a PKI (public key infrastructure), but in order to be implemented it requires a PKG (private key generator) system that generates private keys based on the identity of the signers. The identity must be a unique identifier of the person in the environment in which the system is used (e.g. the e-mail in a firm).

The system described in this report doesn't take into account the confidentiality of the message, it addresses only its authenticity and integrity.

2 Algorithm

2.1 PKG keys generation

The PKG needs its own key pair. It generates it through the execution of the basic RSA steps:

1. It Chooses two random large primes p and q;
2. Computes and stores $n = p \cdot q$ and $\phi(n) = (p-1) \cdot (q-1)$;
3. Chooses a random public key e such that $e \in Z^*_{\phi(n)}$ (so such that $gcd(e, \phi(n)) = 1$);
4. Computes the private key $d = e^{-1} \, mod \, \phi(n)$.

2.2 Signer's key pair

A prerequisite for the application of the described scheme is that every signer has a key pair. This pair is such that:

1. The public key is i_j, the identity of the signer j;
2. The private key is g_j, computed by the PKG as $g_j = i_j^d \, mod \, n$.

2.3 Multisignature generation

In a scenario with l signers that want to sign a message m, every signer j executes the following steps:

1. Chooses a random integer r_j;
2. Computes $t_j = r_j^e \, mod \, n$;
3. Broadcasts t_j to every other signer;
4. Computes $t = (\prod_{j=1}^{l} t_j) \, mod \, n$ (Note that the formula used to compute t is different from the one used in the paper ($t = (\prod_{j=1}^{l} r_j) \, mod \, n$), but doing it in the way I showed makes the formal verification performed at the end of this section work correctly, even though the numeric example worked in both cases.);
5. Computes $s_j = g_j r_j^{H(t,m)} \, mod \, n$ where $H(t, m)$ is the hash of the message together with t, performed through a two-parameters hash function (which function to use is out of the scope of the paper, it can be chosen during the implementation of the scheme). In this way it is possible to compute the signature in an independent way from the message size, and the message integrity is guaranteed;
6. Broadcasts s_j to every other signer;
7. Computes $s = \prod_{j=1}^{l} (s_j \, mod \, n)$;
8. Obtains the final multisignature $\sigma = (t, s)$ that can be sent together with the message;

2.4 Multisignature verification

The receiver receives the message m (not encrypted, confidentiality is out of the scope of the paper) and the signature σ, and can use the identity of the signers (i.e. their public keys) and the public key of the PKG to verify the following:

$$s^e = (\prod_{j=1}^{l} i_j) \, t^{H(t,m)} \, mod \, n$$

If it holds, the multisignature is valid and the message integrity is guaranteed. This works because:

$$s^e = (s_1 \cdots s_l)^e = (g_1 r_1^{H(t,m)} \cdots g_l r_l^{H(t,m)})^e = (g_1 \cdots g_l)^e \cdot (r_1^e \cdots r_l^e)^{H(t,m)} = (i_1^d \cdots i_l^d)^e \cdot (t_1 \cdots t_l)^{H(t,m)}$$

$$= (i_1^d \cdots i_l^d)^{d^{-1}} \cdot t^{H(t,m)} = (i_1 \cdots i_l) \cdot t^{H(t,m)} \, mod \, n$$

3 Example

In this section I will provide a numerical example with relatively small values. Refer to Section 2 for the description of the various steps. To compute the big exponentiations in modular arithmetic I used the on-line tool linked in [1].

3.1 PKG keys generation

PKG selects p and q, computes n and $\phi(n)$. It then selects the public key e and computes the private key d:

$$p = 7 \qquad q = 11$$

$$n = p \cdot q = 77 \qquad \phi(n) = (p-1) \cdot (q-1) = 60$$

$$e = 17 \in Z_{60}^{*} \qquad d = e^{-1} \, mod \, 60 = 53$$

3.2 Multisignature generation

Now I show an example of a multisignature generation procedure with three signers. The corresponding identities i_j (public keys) are selected as three small values, then the PKG computes their private keys g_j using its own one. The signers then compute the multisignature values t and s using their random number r_j (selected as a small values). To do example calculations I just set the hash value $H(t, m) = 15$, where m is the message to be signed (the hash function to use is an implementation choice and it has no restrictions, so this situation is possible).

$$i_1 = 3 \qquad i_2 = 4 \qquad i_3 = 5$$

$$g_j = i_j^d \, mod \, n \quad \rightarrow \quad g_1 = 3^{53} \, mod \, 77 = 5 \qquad g_2 = 4^{53} \, mod \, 77 = 9 \qquad g_3 = 5^{53} \, mod \, 77 = 59$$

$$r_1 = 6 \qquad r_2 = 7 \qquad r_3 = 8$$

$$t_j = r_j^e \, mod \, n \quad \rightarrow \quad t_1 = 6^{17} \, mod \, 77 = 41 \qquad t_2 = 7^{17} \, mod \, 77 = 28 \qquad t_3 = 8^{17} \, mod \, 77 = 57$$

$$t = \prod_{j=1}^{l} (t_j \, mod \, n) \quad \rightarrow \quad t = 41 \cdot 28 \cdot 57 \, mod \, 77 = 63$$

$$H(t, m) = 15$$

$$s_j = g_j r_j^{H(t,m)} \, mod \, n \quad \rightarrow \quad s_1 = 5 \cdot 6^{15} \, mod \, 77 = 72 \qquad s_2 = 9 \cdot 7^{15} \, mod \, 77 = 35 \qquad s_3 = 59 \cdot 8^{15} \, mod \, 77 = 73$$

$$s = \prod_{j=1}^{l} (s_j \, mod \, n) \quad \rightarrow \quad s = 72 \cdot 35 \cdot 73 \, mod \, 77 = 7$$

$$\sigma = (63, 7)$$

3.3 Multisignature verification

$$s^e = (\prod_{j=1}^{l} i_j) \, t^{H(t,m)} \, mod \, n$$

$$7^{17} \, mod \, 77 = 3 * 4 * 5 * 63^{15} \, mod \, 77$$

$$28 = 28 \quad \rightarrow \quad valid.$$

4 Analysis

The proposed scheme respects the two properties mentioned in Section 1.1 because the size of the signature σ does not depend on the number of signers and because we use only one modular exponentiation in the verification procedure. It has also some other relevant properties:

- It is based on RSA, that is one of the most well-known and easy to implement asymmetric signature schemes;
- It is based on the Identity based RSA, so we also don't have the problem of public key authenticity after distribution, there is no need for a PKI. This because the public keys are supposed to be known being the identities of the signers;
- Shamir's IBS scheme is computationally secure against chosen-message forgeability attack, and since this scheme is based on it it can be proven that it is also computationally secure under the same attack;
- The probability that two groups of people's identities combined together generate the same multisignature (multi-signer collusion) is negligible (considering real identities factors such as the e-mail, not small numbers as in the example.);

The scheme has also some drawbacks:

- What to use as identity must be chosen carefully: for example, using a simple ID of the employee (a small number) could result in an increase in the probability of having multi-signer collusion;
- Being and identity based scheme, it inherits all the problems of this kind of approach:
  - The PKG module is the trusted element of the system since it generates and holds all the public keys;
  - For the same reason also the connection between the signers and the PKG must be secure, confidential;
  - If for some reasons a public key must be changed then it is difficult to manage the related process, because it should reflect into a change of the user identification (e.g. the e-mail).

5 Conclusion

In conclusion, the multisignature approach described in [2] is easy to implement, based on the well-known RSA IBS scheme (from which it also inherits some drawbacks as described in Section 4), computationally secure, and it adheres to the two properties mentioned in Section 1.1.

References

[1] Math crypto calculator - brown university. `http://www.math.brown.edu/~jhs/MA0158/MathCryptoCalc.02.html`. Accessed: October 5, 2015.

[2] Lein Harn and Jian Ren. Efficient identity-based {RSA} multisignatures. *Computers & Security*, 27(1–2):12 – 15, 2008.