Comparing With RSA

Julien Cathalo[1⋆], David Naccache[2], and Jean-Jacques Quisquater[1]

[1] UCL Crypto Group
Place du Levant 3, Louvain-la-Neuve, B-1348, Belgium
julien.cathalo@uclouvain.be, jean-jacques.quisquater@uclouvain.be
[2] École normale supérieure, Département d'informatique
45, rue d'Ulm, F-75230 Paris Cedex 05, France
david.naccache@ens.fr

Abstract. A multi-set (MS) is a set where an element can occur more than once. MS hash functions (MSHFs) map MSs of arbitrary cardinality to fixed-length strings.

This paper introduces a new RSA-based MSHF. The new function is efficient and produces small hashes. We prove that the proposed MSHF is collision-resistant under the assumption of unforgeability of deterministic RSA signatures.

In many practical applications, programmers need to compare two (unordered) sets of integers. A trivial solution consists in sorting both sets ($\mathcal{O}(n \log n)$) and comparing them linearly. We show how MS hash functions can be turned into a linear-time, constant-space integer set equality test.

1 Introduction

A multi-set (MS) is a set where elements can occur more than once. MS hash functions (MSHFs) were introduced by Clarke *et alii* in [5]. While standard hash functions map arbitrary-length strings to fixed-length strings, MSHFs map MSs of arbitrary cardinality to fixed-length strings.

An MSHF $\mathcal{H}$ is *incremental* if $\mathcal{H}(A \bigcup B)$ can be computed from $\mathcal{H}(A)$ in time proportional to $\sharp B$.

The `MSet-Mu-Hash` MSHF defined in [5] is MS-collision-resistant, produces small hashes (typically $\cong q$ such that solving discrete logarithms modulo q is hard[3]) and is computationally efficient. `MSet-Mu-Hash` is provably secure in the random oracle model under the discrete logarithm assumption.

In this work we introduce a new MSHF. The proposed function is MS-collision resistant, produces hashes of the size of an RSA modulus and is computationally efficient (comparable to `MSet-Mu-Hash`). However, we prove the new MSHF's

⋆ Research supported by the Belgian Walloon Region projects MAIS (Programme WIST) and IRMA (Programme *Réseaux 2*).

[3] *e.g.* 1024 bits.

security under an assumption different than [5]'s, namely: the unforgeability of deterministic RSA signatures.

Moreover, we show that MSHFs provide a practical solution to the *Set Equality Problem* (SEP). SEP consists in deciding whether two (unordered) sets of n integers are equal. Efficient SEP solutions allow, for instance, to check that two hard drives contain the same files, or that two differently indexed databases contain the same fields. The SEP is related to the *Set Inclusion Problem* (SIP) where one needs to decide whether a set A is a subset of another set B.

In the algebraic computation model where the only allowed operation is comparison, the SEP can be solved in $O(n \log n)$ by sorting both lists; Ben-Or showed that this is optimal [2]. In 2004, Katriel [9] proposed a linear-time set equality test in the algebraic computation model where simple algebraic computations are allowed. Katriel maps the sets into $\mathbb{Z}[X]$ and compares polynomials rather than integers. In essence, [9] shows that SEP is easier when the sets contain integers. However, [9] is impractical as it requires to evaluate the polynomial of a huge value.

Using our MSHF, we propose a *practical* linear-time, constant-space integer MS equality test. The algorithm can be used in practice to compare very large MSs and does not yield false negatives. Immunity against false positives is guaranteed if a specific type of RSA signatures is secure.

This non cryptographic application of RSA is quite unusual as, in general, cryptography "borrows" techniques from other fields (such as complexity theory, number theory or statistics) rather than the other way round.

2 The `MSet-Mu-Hash` Function

Let B be a set. We consider a MS $X = \{x_1, \ldots, x_n\} \in B^n$. The `MSet-Mu-Hash` function proposed by Clarke *et alii* [5] is defined as follows: Let q be a large prime and $H : B \to GF(q)$ be a poly-random function[4].

$$\texttt{MSet-Mu-Hash}(X) = \prod_{i=1}^{n} H(x_i) \bmod q$$

This function is proven to be MS-collision resistant in the random oracle model under the discrete logarithm assumption. It produces small hashes (typically, the size of a prime q such that solving discrete logarithms modulo q is hard) and is computationally efficient.

[4] H is a poly-random function if no polynomial time (in the logarithm of q) algorithm with oracle access H can distinguish between values of H and true random strings, even when the algorithm is permitted to select the arguments to H. *cf.* to [7].

3 Katriel's Set Equality Test

Let $X = \{x_1, \ldots, x_n\} \in \mathbb{Z}^n$ and $Y = \{y_1, \ldots, y_n\} \in \mathbb{Z}^n$. Katriel [9] defines the polynomials:

$$p(z) = \prod_{i=1}^{n}(z - x_i), \quad q(z) = \prod_{i=1}^{n}(z - y_i) \text{ and } d(z) = p(z) - q(z)$$

As $d(z) \equiv 0 \Leftrightarrow X = Y$, the test ascertains that $d \equiv 0$.

A trivial way to do this would be to check that d has $n+1$ roots. However, this would require $n+1$ evaluations of d and as an evaluation of d is linear in n, the test will become quadratic in n.

Katriel evaluates $d(z)$ only once, at a point α which is too large to be a root of $d(z)$, unless $d(z) \equiv 0$:

$$\alpha = 1 + 2(mn)^n \quad \text{where} \quad m = \max\{x_1, \ldots, x_n, y_1, \ldots, y_n\}$$

The test is simply :

$$\texttt{return}(d(\alpha) \stackrel{?}{=} 0)$$

Complexity: This test requires the computation of:

$$\prod_{i=1}^{n}(\alpha - x_i) - \prod_{i=1}^{n}(\alpha - y_i)$$

Which can be done in $2(n-1)$ multiplications but requires a memory capacity quadratic in n. Indeed:

$$\prod_{i=1}^{n}(\alpha - x_i) \cong \alpha^n \cong ((mn)^n)^n = (mn)^{n^2}$$

Storing this value requires $n^2 \log_2(mn)$ bits. *e.g.*, to compare lists of 2^{20} two-byte integers ($m = 2^{16}$, $n = 2^{20}$), one needs a 36×2^{40} bit memory, *i.e.* ~ 4 terabytes. This makes [9] of little practical use.

4 A New RSA-Based MSHF

We now describe a new RSA-based MSHF and a corresponding MS equality test.

4.1 Digital Signatures

The digital signature of a message m is a string that depends on m and a secret known only to the signer. Digital signatures are traditionally (*e.g.* [8]) defined as follows:

Definition 1 (Signature Scheme). *A signature scheme* {Generate, Sign, Verify} *is a collection of three algorithms:*

- *The key generation algorithm* Generate *is a probabilistic algorithm that, given* 1^k*, outputs a pair of matching public and secret keys,* $\{\mathsf{pk}, \mathsf{sk}\}$.
- *The signing algorithm* Sign *takes the message* m *to be signed and the secret key* sk *and returns a signature* $x = \mathsf{Sign}_{\mathsf{sk}}(m)$. Sign *may be probabilistic.*
- *The verification algorithm* Verify *takes a message* m*, a candidate signature* x' *and the public key* pk*. It returns a bit* $\mathsf{Verify}_{\mathsf{pk}}(m, x')$*, equal to one if the signature is accepted, and zero otherwise. We require that:*

$$\mathsf{Verify}_{\mathsf{pk}}(m, \mathsf{Sign}_{\mathsf{sk}}(m)) = 1$$

The security of signature schemes was formalized in an asymptotic setting by Goldwasser, Micali and Rivest in [8]. Here we use the definitions of [1] that provide a framework for a concrete security analysis of digital signatures and consider resistance against *adaptive chosen-message attacks*; *i.e.* a forger $\mathcal{F}$ who dynamically obtains signatures of messages of his choosing and attempts to output a valid forgery.

A *valid forgery* is a message/signature pair $(\tilde{m}, \tilde{x})$ such that $\mathsf{Verify}_{\mathsf{pk}}(\tilde{m}, \tilde{x}) = 1$ whilst the signature of $\tilde{m}$ was never requested by $\mathcal{F}$.

Definition 2. *A forger* $\mathcal{F}$ *is said to* $(t, q_{sig}, \varepsilon)$*-break the signature scheme if after at most* $q_{sig}(k)$ *signature queries and* $t(k)$ *processing time,* $\mathcal{F}$ *outputs a valid forgery with probability at least* $\varepsilon(k)$ *for any* $k > 0$.

Definition 3. *A signature scheme is* EUF-CMA $(t,\ q_{sig},\ \varepsilon)$*-secure if there is no forger capable of* $(t, q_{sig}, \varepsilon)$*-breaking the signature scheme.*

Definition 4. *A signature scheme is* EUF-CMA*-secure if for any forger* $\mathcal{F}$ *that* $(t(k), q_{sig}(k), \varepsilon(k))$*-breaks the scheme, if* $t(k)$ *and* $q_{sig}(k)$ *are polynomial, then* $\varepsilon(k)$ *is negligible.*

4.2 RSA Signatures

RSA [10] is certainly the most famous public-key cryptosystem:

System parameters **:** Two integers $k, \ell \in \mathbb{N}$ and a function $\mu : \{0,1\}^\ell \to \{0,1\}^k$.
Generate **:** On input 1^k,

- Randomly select two distinct $k/2$-bit primes p and q.

– Compute $N = pq$.
– Pick a random encryption exponent $e \in \mathbb{Z}^*_{\phi(N)}$
– Compute the corresponding decryption exponent $d = e^{-1} \bmod \phi(N)$.

The output of the key generation process is $\{N, e, d\}$; the public key is $\mathsf{pk} = \{N, e\}$ and the private key is $\mathsf{sk} = \{N, d\}$.

Sign : Return $y = \mu(m)^d \bmod N$.

Verify : If $y^e \bmod N = \mu(m)$ then return 1 else return 0.

4.3 Coron-Koeune-Naccache Long-Message RSA Encoding

Signing long messages with RSA is possible using a construction proposed by Coron, Koeune and Naccache (CKN) in [6] (and improved in [4]). CKN split a long message into short blocks, encode each block with μ and multiply all the encoded blocks modulo N. Before encoding a block, CKN's procedure appends to each block a 0 and the block's index i. Then the product of so-formed encodings is appended to 1 and re-encoded again with μ.

System parameters : Two integers $k > 0$ and $a \in [0, k-1]$ and a function

$$\mu : \{0,1\}^{k+1} \to \{0,1\}^k$$

Generate : As in standard RSA.

Sign : Split the message m into $(k-a)$-bit blocks such that $m = m[1]|| \ldots ||m[r]$.

Let $\alpha = \prod_{i=1}^{r} \mu(0||i||m[i]) \bmod N$ where i is an a-bit string representing i.

Let $y = \mu(1||\alpha)$ and return $y^d \bmod N$.

Verify : Let $y = x^e \bmod N$ and recompute $\alpha = \prod_{i=1}^{r} \mu(0||i||m[i]) \bmod N$.

If $y = \mu(1||\alpha)$ then return 1 else return 0.

4.4 The New MSHF

Let N, e, d be parameters selected as in sub-section 4.3. We additionally require that e is a prime number and $e > n$. Let $\mu : \{0,1\}^k \to \{0,1\}^k$ be an encoding function. We propose the following MSHF:

Apply μ to all the elements of $X = \{x_1, \ldots, x_n\}$ and multiply all the encoded integers modulo N:

$$\mathcal{H}(X) = \prod_{i=1}^{n} \mu(x_i) \bmod N$$

Note that d and e are not used in the function and that $\mathcal{H}$ is incremental as it can be updated easily if new elements need to be added to the set.

This construction is very similar to CKN's long-message RSA encoding. The difference is that indices are omitted because the order of the elements is not taken into account. The equality test for two MSs X and Y is simply :

$$\texttt{return}(\mathcal{H}(X) \stackrel{?}{=} \mathcal{H}(Y))$$

The setup is a slightly modified RSA since we additionally require that e is a prime number and that $e > n$. The latter requirement is not a problem as n is the size of the compared MSs (*e.g.* $n < 2^{30}$).

5 MS Collision-Resistance Proof

We now prove the MS collision-resistance of our MSHF. We show that computing a collision in time t with probability ε implies forging μ-encoded RSA signatures in polynomially-related t' and ε'.

Let $\mathsf{Generate'}(k, n)$ be an algorithm that, given two positive integers k and n, returns (N, e, d, μ) such that (N, e, d) are RSA parameters, $|N| = k$ and e is a prime number such that $e > n$. Moreover, $\mu : \{0,1\}^k \to \{0,1\}^k$ is such that the deterministic-padding RSA scheme obtained using μ and (N, e, d) is EUF-CMA secure.

Given k and n and an output of $\mathsf{Generate'}(k, n)$, we consider two different games.

In the first game Game_1, a forger $\mathcal{F}$ is given $\{N, e, \mu\}$. $\mathcal{F}$ has access to a signing oracle $\mathcal{S}$ that, when given $m_i \in \{0,1\}^k$, answers $\mu(m_i)^d \bmod N$. After $q_1(k, n)$ signature requests to $\mathcal{S}$ and $t_1(k, n)$ computing time, $\mathcal{F}$ outputs with probability $\varepsilon_1(k, n)$ a forgery (m, s) such that $s = \mu(m)^d \bmod N$ and m was never signed by $\mathcal{S}$. When $n = n(k)$ is a polynomial in k, the security of the μ-based RSA deterministic-encoding signatures implies that, for any $\mathcal{F}$, if $t_1(k)$ and $q_1(k)$ are polynomial then $\varepsilon_1(k)$ is negligible.

In the second game Game_2, an adversary $\mathcal{A}$ is given $\{N, e, \mu, n\}$. This adversary's goal is to produce a collision. It wins if it can find two sets $X = \{x_1, \ldots, x_{n'}\}$ and $Y = \{y_1, \ldots, y_{n''}\}$ where $x_i, y_i \in [0, 2^k[$ such that $X \neq Y$, $n' \leq n$, $n'' \leq n$ and

$$\prod_{i=1}^{n'} \mu(x_i) \equiv \prod_{i=1}^{n''} \mu(y_i) \pmod N$$

$\mathcal{A}$ runs in time $t_2(k, n)$ and succeeds with probability $\varepsilon_2(k, n)$.

Theorem 1. *If there exists an adversary $\mathcal{A}$ that finds a collision in time $t_2(k, n)$ with probability $\varepsilon_2(k, n)$, then there exists a forger $\mathcal{F}$ that finds a forgery after $q_1(k, n) < 2n$ queries to $\mathcal{S}$ and $t_1(k, n) = t_2(k, n) + \mathcal{O}(n^2) + \mathcal{O}(nk^2)$ computing time, with probability $\varepsilon_1(k, n) = \varepsilon_2(k, n)$.*

Proof. Let $\mathcal{A}$ be an adversary that finds a collision in time t_2 with probability ε_2. We construct a forger $\mathcal{F}$ as follows.

$\mathcal{F}$ first uses $\mathcal{A}$ to try to obtain a collision. If, after t_2 time units, $\mathcal{A}$ does not succeed, $\mathcal{F}$ stops. This happens with probability $1 - \varepsilon_2$.

Otherwise, $\mathcal{A}$ returns a collision. This happens with probability ε_2 and in this case $\mathcal{F}$ learns $X = \{x_1, \ldots, x_{n'}\}$ and $Y = \{y_1, \ldots, y_{n''}\}$ such that $X \neq Y$ and:

$$\prod_{i=1}^{n'} \mu(x_i) \equiv \prod_{i=1}^{n''} \mu(y_i) \pmod N$$

We denote by $\sharp X(x)$ number of occurrences of an element x in a MS X.

Since $X \neq Y$, there exists x_{i_0} such that $\sharp X(x_{i_0}) \neq \sharp Y(x_{i_0})$ and without loss of generality, we assume that $\sharp X(x_{i_0}) > \sharp Y(x_{i_0})$.

Let $a = \sharp X(x_{i_0}) - \sharp Y(x_{i_0})$ (note that $1 \leq a \leq n'$).

The forger $\mathcal{F}$ finds x_{i_0} and a by sorting and comparing X and Y. We have:

$$\prod_{i \in V} \mu(x_i) \times \mu(x_{i_0})^{\sharp X(x_{i_0})} \equiv \prod_{i \in W} \mu(y_i) \times \mu(x_{i_0})^{\sharp Y(x_{i_0})} \pmod N$$

where V is the subset of $\{1, \ldots, n'\}$ corresponding to the indices of the x_i not equal to x_{i_0} and W is the subset of $\{1, \ldots, n''\}$ corresponding to the indices of the y_i not equal to x_{i_0}.

We get:

$$\mu(x_{i_0})^a \equiv \prod_{i \in V} \mu(x_i)^{-1} \times \prod_{i \in W} \mu(y_i) \pmod N \tag{1}$$

The integer x_{i_0} does not appear on the right side of this equation.

$\mathcal{F}$ computes u and k such that $au = ke + 1$ (since e is prime and $0 < a \leq n' \leq n < e$, a is invertible modulo e). $\mathcal{F}$ obtains from the signing oracle $\mathcal{S}$ the signatures $s_i = \mu(x_i)^d \bmod N$ of all x_i such that $i \in V$ and the signatures $s'_i = \mu(y_i)^d \bmod N$ of all y_i for $i = 1, \ldots, n''$. Then $\mathcal{F}$ computes:

$$s = \left(\left(\prod_{i \in V} s_i \right)^{-1} \times \prod_{i \in W} s'_i \right)^u \times \mu(x_{i_0})^{-k} \pmod N$$

One can show that $s = \mu(x_{i_0})^d \bmod N$ using equation (1):

$$\begin{aligned}
\mu(x_{i_0})^{au} &\equiv \prod_{i \in V} \mu(x_i)^{-u} \times \prod_{i \in W} \mu(y_i)^u && \pmod N \\
\mu(x_{i_0})^{ke+1} &\equiv \prod_{i \in V} \mu(x_i)^{-u} \times \prod_{i \in W} \mu(y_i)^u && \pmod N \\
\mu(x_{i_0}) &\equiv \prod_{i \in V} \mu(x_i)^{-u} \times \prod_{i \in W} \mu(y_i)^u \times \mu(x_{i_0})^{-ke} && \pmod N \\
\mu(x_{i_0})^d &\equiv \prod_{i \in V} \mu(x_i)^{-ud} \times \prod_{i \in W} \mu(y_i)^{ud} \times \mu(x_{i_0})^{-k} && \pmod N \\
\mu(x_{i_0})^d &\equiv s && \pmod N
\end{aligned}$$

This implies that (x_{i_0}, s) is a valid (message, signature) pair. Since the message x_{i_0} was never sent to $\mathcal{S}$, $\mathcal{F}$ succeeds in finding a forgery. The number of queries to $\mathcal{S}$ is $\#V + n'' = (n' - a) + n'' < 2n$.

We now evaluate $\mathcal{F}$'s running time. First, $\mathcal{F}$ runs $\mathcal{A}$ in time $t_2(k, n)$. Finding x_{i_0} and a by sorting and comparing X and Y takes $\mathcal{O}(n \log n)$ time[5]. Computing u and k takes $\mathcal{O}(n^2)$ time[6]; s can be computed in $2n$ modular multiplications (*i.e.* $\mathcal{O}(nk^2)$ time), n modular inversions (still $\mathcal{O}(nk^2)$ time), two modular exponentiations ($\mathcal{O}(k^2 \log n)$ dominated by $\mathcal{O}(nk^2)$) and an evaluation of μ, that we omit. All in all, the total running time of $\mathcal{F}$ is:

$$t_1(k, n) = t_2(k, n) + \mathcal{O}(n^2) + \mathcal{O}(nk^2)$$

□

Using Theorem 1 we infer that no algorithm can efficiently find collisions:

Theorem 2. *If $n = n(k)$ is polynomial, the success probability of any adversary that running in polynomial time $t_2(k)$ is negligible.*

Proof. Assume that $n(k)$ is polynomial and that $\mathcal{A}$ finds a collision in polynomial time $t_2(k)$. We want to show that $\varepsilon_2(k)$ is negligible.

By virtue of Theorem 1, there exists an $\mathcal{F}$ that finds a forgery after $q_1(k)$ signature queries in time $t_1(k)$. The number of queries $q_1(k) < 2n(k)$ is polynomial. As we have seen that $t_1(k, n) = t_2(k, n) + \mathcal{O}(n^2) + \mathcal{O}(nk^2)$ is also polynomial. Therefore the attacker's success probability $\varepsilon_1(k)$ is negligible. By virtue of Theorem 1, $\varepsilon_1(k) = \varepsilon_2(k)$ and hence $\varepsilon_2(k)$ is negligible. □

Theorem 2 validates the MSHF's asymptotic behavior.

In practical terms the above means that if a modulus N and a deterministic encoding function μ can be safely used to produce RSA signatures, they can also be used to compute MS hashes.

6 Conclusion & Further Research

In this paper, we proposed a new MSHF whose collision-resistance is directly linked to the security of deterministic-encoding RSA signatures.

The function allows to test if two integer sets are equal using moderate memory and computational resources. The test does not yield false negatives and with carefully chosen parameters, it does not yield false positives either since we prove that a single false positive would imply the insecurity of deterministic RSA signature encoding.

[5] Dominated by $\mathcal{O}(n^2)$.

[6] Extended Euclidean algorithm.

While this gives a practical answer to a theoretical question asked by Katriel, we still do not know how to generalize the proposed construction to solve the SIP *i.e.* test if a set A is a subset of a set B.

Another open question is whether a similar construction based on an aggregate signature scheme (*e.g.* [3]) could also be used to provide MSHF functions. The idea would be to multiply hashes[7] of set elements and prove the resulting MSHF's collision-resistance under the assumption that the aggregate signature scheme is secure.

Acknowledgements

The authors thank Sylvie Baudine, Fabien Laguillaumie, Mark Manulis, David Pointcheval and an anonymous ICALP referee for their helpful comments and suggestions.

References

1. M. Bellare and P. Rogaway. The exact security of digital signatures - How to sign with RSA and Rabin. In U. Maurer, editor, *Advances in Cryptology - Eurocrypt 96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer-Verlag, 1996.
2. M. Ben-Or. Lower bounds for algebraic computation trees. In STOC*'83 - Proceedings of the fifteenth annual* ACM *symposium on the Theory of Computing*, pages 80–86. ACM, 1983.
3. D. Boneh, B. Lynn, and H. Shacham. Short Signatures from the Weil Pairing. In *Journal of Cryptology*, volume 17, pages 297-319, 2004.
4. J. Cathalo, J.-S. Coron, and D. Naccache. From fixed-length to arbitrary-length RSA encoding schemes revisited. In S. Vaudenay, editor, *8th International Workshop on Practice and Theory in Public-Key Cryptography (*PKC *2005)*, volume 3386 of *Lecture Notes in Computer Science*, pages 234–243. Springer, January 2005.
5. D. Clarke, S. Devadas, M. van Dijk, B. Gassend, and G. Suh. Incremental multiset hash functions and their application to memory integrity checking. In Chi-Sung Laih, editor, *Asiacrypt*, volume 2894 of *Lecture Notes in Computer Science*, pages 188–207. Springer, 2003.
6. J.-S. Coron, F. Koeune, and D. Naccache. From fixed-length to arbitrary-length RSA padding schemes. In T. Okamoto, editor, *Advances in Cryptology - Asiacrypt 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 90–96. Springer-Verlag, 2000.
7. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. In *Journal of the* ACM, volume 33(4), pages 210-217, 1986.
8. S. Goldwasser, S. Micali, and R. Rivest. A Digital signature scheme secure against adaptive chosen-message attacks. In SIAM *Journal of computing*, volume 17, pages 281–308, April 1988.

[7] Obtained using the hash function of the aggregate signature scheme.

9. I. Katriel. On the algebraic complexity of set equality and inclusion. In *Information Processing Letters*, volume 92, pages 175–178, November 2004.
10. R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, volume 21 (2), pages 120-126, 1978.