# Better than BiBa: Short One-time Signatures with Fast Signing and Verifying

Leonid Reyzin* Natan Reyzin

April 30, 2002
(Algorithm 2 on p. 4 clarified October 17, 2007)

**Abstract**

One-time signature schemes have found numerous applications: in ordinary, on-line/off-line, and forward-secure signatures. More recently, they have been used in multicast and broadcast authentication. We propose a one-time signature scheme with very efficient signing and verifying, and short signatures. Our scheme is well-suited for broadcast authentication, and, in fact, can be viewed as an improvement of the BiBa one-time signature (proposed by Perrig in CCS 2001 for broadcast authentication).

## 1 Introduction

In [Per01], Perrig proposes a one-time signature scheme called "BiBa" (for "Bins and Balls"). BiBa's main advantages are fast verification and short signatures. In fact, to the best of our knowledge, BiBa has the fastest verification of all currently-known one-time signature schemes. These desirable properties allow Perrig to design a stream authentication scheme with small communication overhead and fast authentication of each packet (also called BiBa in [Per01]).

BiBa's main disadvantage is the time required to sign a message, which is longer than in most previously proposed one-time signature schemes. We present a simpler one-time signature scheme that maintains BiBa's advantages and removes its main disadvantage. Specifically, in our scheme verifying is as fast as in BiBa, and signing is even faster than verifying. The key and signature sizes are slightly improved, as well (for the same security level).

Like BiBa, our signature scheme can be used $r$ times, instead of just once, for small values of $r$ (in both schemes, security decreases as $r$ increases), which is essential for its use in the stream authentication scheme of [Per01]. The security of our scheme relies only on complexity-theoretic assumptions, and does not require the use of random oracles. This is in contrast to BiBa, in which the use of random oracles seems essential.

We present our scheme in two parts. In Section 2, we slightly generalize a scheme proposed by Bos and Chaum [BC92]), providing a tradeoff that allows us to decrease signature size and verification time at the expense of increasing the public key length. The resulting scheme is not, however, as efficient as we wish. Thus, in Section 3, we present our most efficient construction

*Boston University, 111 Cummington St., Boston, MA 02215; reyzin@cs.bu.edu; http://www.cs.bu.edu/~reyzin

(named “HORS,” for reasons to be explained), whose security is based on stronger assumptions. In HORS, signing requires just one hash function evaluation, and verifying requires 17 hash function evaluations for a high level of security.

### 1.1 Prior Work

One-time signatures based on the idea of committing to secret keys via one-way functions were proposed independently by Lamport [Lam79] and (in an interactive setting) by Rabin [Rab78]. Various improvements were proposed by Meyer and Matyas [MM82, pages 406–409], Merkle [Mer82], Winternitz (as cited in [Mer87]), Vaudenay [Vau92] (in an interactive setting), Bos and Chaum [BC92], and Even, Goldreich and Micali [EGM96]. Bleichenbacher and Maurer considered generalization of the above work in [BM94, BM96a, BM96b].

Perrig’s BiBa [Per01], however, appears to be the first scheme whose primary design goal was fast signature verification (while Bleichenbacher and Maurer concern themselves with “optimal” one-time signature schemes, their notion of optimality translates into fast key generation, rather than efficient signing or verifying). Our scheme preserves BiBa’s verifying efficiency, dramatically increases its signing efficiency, and slightly decreases the sizes of BiBa’s keys and signatures.

### 1.2 Applications of One-Time Signatures

One-time signatures have found applications in constructions of ordinary signature schemes [Mer87, Mer89], forward-secure signature schemes [AR00], on-line/off-line signature schemes [EGM96], and stream/multicast authentication [Roh99], among others. We note that our scheme fits well into all of these applications, including, in particular, the BiBa broadcast authentication scheme of [Per01]. In fact, the BiBa broadcast authentication scheme itself would need no modifications at all if our signature scheme was substituted for the BiBa signature scheme.

## 2 The Construction Based on One-Way Functions

The construction presented in this section relies solely on one-way functions for its security. It is a simple generalization of the construction of Bos and Chaum [BC92], which, in turn, is a generalization of Lamport’s construction [Lam79]. As stated above, this scheme is mainly of theoretical interest due to performance considerations. However, it is a stepping stone to our ultimate signature scheme, and thus we present it first.

Preliminaries. To build a scheme to sign $b$-bit messages[1], pick $t$ and $k$ such that $\binom{t}{k} \geq 2^b$. (Naturally, there are multiple possible choices for $t$ and $k$, and, in fact, a trade-off between them. We note that the public key size will be linear in $t$, and the signature size and verification time will be linear in $k$).

Let $T$ denote the set $\{1, 2, \ldots, t\}$.

Let $S$ be a bijective function that, on input $m$, $0 \leq m < \binom{t}{k}$, outputs the $m$-th $k$-element subset of the $T$ (of course, there are many possibilities for $S$; we do not care how $S$ is chosen as long as it is efficiently implementable and bijective). Our proposal for implementing $S$ is contained in Section 2.1.

[1] In order to sign arbitrary-length messages, one can use the standard technique of applying a collision-resistant hash function to the message and then signing the resulting hash value.

Let $f$ be a one-way function operating on $l$-bit strings, for a security parameter $l$.

THE SCHEME. To generate a key pair given the security parameter $1^l$, generate $t$ random $l$-bit quantities for the secret key: $SK = (s_1, \ldots, s_t)$. Compute the public key as follows: $PK = (v_1, \ldots, v_t)$, where $v_1 = f(s_1), \ldots, v_t = f(s_t)$.

To sign a $b$-bit message $m$, interpret $m$ as an integer value between 0 and $2^b - 1$, and let $\{i_1, \ldots, i_k\}$ be the $m$-th $k$-element subset of $T$ (found using the function $S$). Reveal the values $s_{i_1}, \ldots, s_{i_k}$ as the signature.

To verify a signature $(s'_1, s'_2, \ldots, s'_k)$ on a message, again interpret $m$ as an integer value between 0 and $2^b - 1$, and let $\{i_1, \ldots, i_k\}$ be the $m$-th $k$-element subset of $T$. Verify that $f(s'_1) = v_{i_1}, \ldots, f(s'_t) = v_{i_t}$.

EFFICIENCY. Key generation requires $t$ evaluations of the one-way function. The secret key size is $lt$ bits, and the public key size is $f_l t$ bits, where $f_l$ is the length of the one-way function output on input of length $l$. The signature is $kt$ bits long.

The signing algorithm takes as long as running time of the algorithm for $S$: the time required to find the $m$-th $k$-element subset of a $t$-element set. Our implementations of $S$, in Section 2.1, have running time $O(tk \log^2 t)$, or $O(k^2 \log t \log k)$ with some precomputation. The verifying algorithm takes the same amount of time, plus $k$ evaluations of the one-way function.

We note that [BC92] proposed essentially the same scheme, with the restriction that $k = t/2$. Such choice minimizes the public key size. Our goal (and the goal of BiBa), however, is to minimize the signature size while keeping the public key size reasonable.

SECURITY. Because each message corresponds to a different $k$-element subset of the set $T$, it is trivial to prove that, in order to existentially forge a signature on a new message after a one-time adaptive chosen message attack, the forger would have to invert the one-way function $f$ on at least one of the $t - k$ values in the public key for which the corresponding value in the secret key has not been revealed. Thus, the security of the signature scheme is reduced to the one-wayness of $f$.

If longer messages are being hashed down to $b$ bits before being signed, then the security relies not only on the one-wayness of $f$, but also on the collision-resistance of the hash function.

## 2.1 The Selection Algorithms

Here we present two algorithms that, on input $m$, $0 \le m < \binom{t}{k}$, output the $m$-th $k$-element subset of $T$, where $T = \{1, 2, \ldots, t\}$. The first algorithm is due to Bos and Chaum [BC92]; the second one, to the best of our knowledge, is new.

### Algorithm 1

The first algorithm is based on the following equation:

$$\binom{t}{k} = \binom{t-1}{k-1} + \binom{t-1}{k}.$$

In other words, if the last element of $T$ belongs to the subset, then there are $t-1$ elements remaining from which $k-1$ need to be chosen. Otherwise, there are $t-1$ elements remaining from which $k$ need to be chosen.

Thus, on input $(m, k, t)$, the algorithm checks if $m < \binom{t-1}{k-1}$. If so, it adds $t$ to the output subset, and recurses on $(m, k-1, t-1)$. Else, it adds nothing to the output subset, and recurses on $\left(m - \binom{t-1}{k-1}, k, t-1\right)$. Note that $\binom{t}{k}$ can be precomputed once and for all, and then at each recursive level, the algorithm simply needs to do one division and one multiplication to compute $\binom{t-1}{k-1} = k\binom{t}{k}/t$, plus possibly one subtraction to compute $\binom{t-1}{k} = \binom{t}{k} - \binom{t-1}{k-1}$.

Thus, the cost per recursive level is one multiplication of an $O(k \log t)$-bit number by an $O(\log k)$-bit number, followed by one division of an $O(\log k + k \log t)$-bit number by an $O(\log t)$-bit number, for a total of $O(k \log k \log t) + O((\log k + k \log t)(\log t)) = O(k \log^2 t)$. There are $t$ levels, for the total cost of $tk \log^2 t$.

**Algorithm 2**

The second algorithm is slightly more complicated, and is based on the following equation:

$$\begin{aligned}\binom{t}{k} &= \binom{\lceil t/2 \rceil}{0}\binom{\lfloor t/2 \rfloor}{k} + \binom{\lceil t/2 \rceil}{1}\binom{\lfloor t/2 \rfloor}{k-1} + \ldots + \binom{\lceil t/2 \rceil}{k}\binom{\lfloor t/2 \rfloor}{0} \\ &= \sum_{i=0}^{k} \binom{\lceil t/2 \rceil}{i}\binom{\lfloor t/2 \rfloor}{k-i}.\end{aligned}$$

In other words, if $k$ elements are selected from $T$, then, for some $j$, $j$ elements come from the first half of $T$, and $k - j$ elements come from the second half of $T$.

Thus, on input $(m, k, t)$, the algorithm finds $j$ $(0 \le j \le k)$ such that

$$\sum_{i=0}^{j-1} \binom{\lceil t/2 \rceil}{i}\binom{\lfloor t/2 \rfloor}{k-i} \le m\,, \text{ but } \sum_{i=0}^{j} \binom{\lceil t/2 \rceil}{i}\binom{\lfloor t/2 \rfloor}{k-i} > m\,.$$

Then let $m' = m - \sum_{i=0}^{j-1} \binom{\lceil t/2 \rceil}{i}\binom{\lfloor t/2 \rfloor}{k-i}$; let $m_1 = m' \operatorname{div} \binom{\lceil t/2 \rceil}{j}$ and $m_2 = m' \bmod \binom{\lceil t/2 \rceil}{j}$ (where div stands for integer division, and mod stands for the remainder). The algorithm recurses on $(m_1, j, \lceil t/2 \rceil)$ to find the $m_1$-th $j$-element subset of $\{1, 2, \ldots, \lceil t/2 \rceil\}$ and on $(m_2, k-j, \lfloor t/2 \rfloor)$ to find the $m_2$-th $(k-j)$-element subset of $\{\lceil t/2 \rceil + 1, \lceil t/2 \rceil + 2, \ldots, t\}$. It then combines the subsets that the two recursive calls return.

If $t$ is a power of two, and $k$ is not too large, then it is possible to precompute $\binom{t/2^a}{b}$, for all $a < \log t$ and $b < k$ (this would be $O(k \log t)$ values, for $O(k^2 \log^2 t)$ bits). The recursive calls form a binary tree with of depth $\log t$. The cost of each invocation within this binary tree is dominated by multiplications of precomputed values of the form $\binom{t/2^a}{b}$; moreover, there are at most as many multiplications as the size of the subset that the particular invocation is looking for. Note that the sum of the sizes of subsets across each *level* of the binary tree is at most $k$. Hence, summing up over all $\log t$ levels, there are a total of $O(k \log t)$ multiplications of $O(k \log t)$-bit values. Thus, the computational cost is $O(k^2 \log^2 t)$.

Moreover, for a random $m$, the expected depth of the tree is actually $O(\log k)$ (because, intuitively, the subset is likely to be divided fairly evenly between the two halves of $T$). Therefore, the expected running time for a random $m$ is actually $O(k^2 \log t \log k)$.

Finally, we note that, in practice, it is better to change the order in which the sum is computed: instead of going from $i = 0$ to $k$, it is better to start with $i = k/2$ and go in both directions (to 0 and to $k$). This is so because we are likely to stop sooner, because $k$ is more likely to be split evenly. Moreover, it is more efficient to have the base case of $k = 1$ than $k = 0$. Our implementation of this algorithm (based on Wei Dai's Crypto++ library for multiprecision arithmetic) takes .09 milliseconds on a 1700 MHz Pentium 4 for $t = 1024$, $k = 16$, and a random $m$.

# 3 Construction Based On "Subset-Resilient" Functions

This section presents our most efficient construction.

In the above scheme, the function $S$ makes it *impossible* to find two distinct messages that will result in the same $k$-element subset of $T$ (because $S$ is bijective, it guarantees that such two messages do not exist). To improve efficiency, we propose to replace $S$ with a function $H$ that provides no such iron-clad guarantee, but merely makes it *infeasible* to find two such messages. Moreover, we will relax the requirement that the subset of $T$ corresponding to each message contain *exactly* $k$ elements, and rather require it to contain *at most* $k$ elements. The requirement on $H$ now will be that it is infeasible to find two messages, $m_1$ and $m_2$, such that $H(m_2) \subseteq H(m_1)$.

This relaxation of the requirements allows us to consider using a single public key to provide not one, but several signatures. If the signer provides up to $r$ signatures, then, for the scheme to be secure, it should be infeasible to find $r+1$ messages $m_1, m_2, \ldots, m_{r+1}$ such that $H(m_{r+1}) \subseteq H(m_1) \cup H(m_2) \cup \ldots \cup H(m_r)$.

Using $H$ instead of $S$ also allows us to remove the restrictions on message length, and with it the need for collision-resistant hashing in order to sign longer messages.

It should be clear how to formalize the precise requirements on $H$ in order to prove security of our one-time signature scheme (naturally, in order for the requirements to be meaningful, $H$ has to be selected at random from a family of functions, and which particular $H$ is chosen should be part of the public key). We do so in Appendix A. We note that the formalizations are different depending on whether the adversary for the signature scheme is allowed a chosen message attack or a random message attack. We also note that if $H$ is modeled as a random oracle, then it trivially satisfies the definitions.

The Efficient Scheme. We propose using a cryptographic hash function *Hash*, for example SHA-1 [NIS95] or RIPEMD-160 [DBP96], to construct $H$ as follows:

1. split the output of the hash function into $k$ substrings of length $\log t$ each;
2. interpret each $(\log t)$-bit substring as integer written in binary;
3. combine these integers to form the subset of $T$ of size at most $k$.

We believe that such $H$ satisfies our definition (it certainly does if the cryptographic hash function is modeled as a random oracle).

Such construction of $H$ results in the scheme we call HORS (for "Hash to Obtain Random Subset"). We present it in Figure 1. HORS possesses extremely efficient signing, requiring just one evaluation of the cryptographic hash function, and efficient verifying, requiring one evaluation of the hash function in addition to $k$ evaluations of the one-way function $f$. (We note that in every signature scheme used in practice, both signing and verifying always require at least one evaluation of a cryptographic hash function if the messages are long.)

The use of cryptographic hash function with 160-bit outputs results in practically convenient parameters. One can take $k = 16$ and $t = 2^{10} = 1024$, or $k = 20$ and $t = 2^8 = 256$. We analyze the security of these specific parameters in the Section 4.

Security. The security proof for this scheme follows directly from subset-resilience of the hash function, as defined in Appendix A, and one-wayness of $f$. Specifically, the signatures are existentially unforgeable against $r$-time chosen-message (or random-message) attacks (in the sense

**Key Generation**

**Input:** Parameters $l, k, t$

Generate $t$ random $l$-bit strings $s_1, s_2, \ldots, s_t$

Let $v_i = f(s_i)$ for $1 \leq i \leq t$

**Output:** $PK = (k, v_1, v_2, \ldots, v_t)$ and $SK = (k, s_1, s_2, \ldots, s_t)$

---

**Signing**

**Input:** Message $m$ and secret key $SK = (k, s_1, s_2, \ldots, s_t)$

Let $h = Hash(m)$

Split $h$ into $k$ substrings $h_1, h_2, \ldots, h_k$, of length $\log_2 t$ bits each

Interpret each $h_j$ as an integer $i_j$ for $1 \leq j \leq k$

**Output:** $\sigma = (s_{i_1}, s_{i_2}, \ldots, s_{i_k})$

---

**Verifying**

**Input:** Message $m$, signature $\sigma = (s'_1, s'_2, \ldots, s'_k)$, and public key $PK = (k, v_1, v_2, \ldots, v_t)$

Let $h = Hash(m)$

Split into $k$ substrings $h_1, h_2, \ldots, h_k$, of length $\log_2 t$ bits each

Interpret each $h_j$ as an integer $i_j$ for $1 \leq j \leq k$

**Output:** "accept" if for each $j$, $1 \leq j \leq k$, $f(s'_j) = v_{i_j}$; "reject" otherwise

Figure 1: *HORS one-time signature scheme, where $f$ is a one-way function and Hash is a hash function. Both $f$ and Hash may be implemented using a standard hash function, such as SHA-1 or RIPEMD-160. Suggested parameter values are $l = 80$, $k = 16$ and $t = 1024$, or $l = 80$, $k = 20$ and $t = 256$.*

of [GMR88]) if the hash function is $r$-subset-resilient (or, respectively, $r$-target-subset-resilient) and $f$ is one-way. The proof trivially reduces a signature forgery to a break of the one-way function or the hash function, depending on whether the forgery uses a new element of $T$ or only re-uses the ones already revealed during the attack.

The exact security against non-adaptive chosen-message attack is analyzed in the next section.

## 4 Comparison with BiBa

BiBa requires about $2t$ calls to the random oracle for signing messages, while, in contrast, HORS requires only one call to $H$. Verification in BiBa requires $k$ calls to the one-way function $f$, just like in HORS. However, BiBa verification also requires $k$ calls to the random oracle, while HORS requires only one call to $H$. Thus, our scheme is significantly more efficient in signing and slightly more efficient in verifying.

Another advantage of HORS is that slightly smaller values of $t$ and $k$ can be used to achieve the same security levels (or, alternatively, more security can be obtained from the same values of $t$ and $k$), as shown below.

One cannot compare the security of our scheme directly to the security of BiBa, because BiBa relies on the assumption that *Hash* is a random oracle, whereas HORS does not. Therefore, for a fair comparison, we have to make the same assumptions as [Per01] makes: that *Hash* is a random oracle. Given this, we analyze the exact security of HORS against $r$-non-adaptive-message attack: that is, just like [Per01], we assume that the adversary obtains signatures on $r$ messages of its choice (but the choice is independent of *Hash*), and then tries to forge a signature on a any new message $m$ of its choice. We are interested in the probability that the adversary is able to do so without inverting the one-way function $f$. It is quite easy to see that this probability is at most $(rk/t)^k$ for each invocation of *Hash*, i.e., the probability that after $rk$ elements of $T$ are fixed, $k$ elements chosen at random are a subset of them.

In other words, we get $k(\log t - \log k - \log r)$ bits of security. Thus, we use $k = 16$, $t = 1024$ and $r = 1$, then the security level is $2^{-96}$. After four signatures are given, the probability of forgery is $2^{-64}$. This is in contrast to $2^{-58}$ for Perrig's scheme with the same parameters (see Section 5 of [Per01]). Alternatively, we could reduce the size of the public key $t = 790$ to match the $2^{-58}$ security level.

For the example of $k = 20$ and $t = 256$, our security level is $2^{-73}$ if $r = 1$ and $2^{-53}$ if $r = 2$.

We note that the above does not analyze the security of HORS against *adaptive* chosen-message attacks, where the adversary gets to decide the messages on which to obtain signatures *after* evaluating *Hash* on multiple messages. The security of HORS against such an attack is lower than above (because, similarly to the "birthday paradox," the adversary has a higher chance of being able to choose the $r+1$ messages such that the union of the subsets corresponding to the first $r$ of them covers the subset corresponding the last one). However, HORS is more secure against such an attack than BiBa is. A precise analysis will be provided in the full version of this paper.

## Acknowledgments

We are thankful to Kevin Fu and Michael Freedman for helpful discussions, to Wei Dai for his Crypto++ library, to Jonathan Katz for pointing us to the [BC92] paper, and Yasuharu Sugano for finding a typo in Algorithm 2.

## References

[AR00] Michel Abdalla and Leonid Reyzin. A new forward-secure digital signature scheme. In Tatsuaki Okamoto, editor, *Advances in Cryptology—ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 116–129, Kyoto, Japan, 2000. Springer-Verlag. Full version available from the Cryptology ePrint Archive, record 2000/002, `http://eprint.iacr.org/`.

[BC92] Jurjen N. E. Bos and David Chaum. Provably unforgeable signatures. In Ernest F. Brickell, editor, *Advances in Cryptology—CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 1–14. Springer-Verlag, 1993, 16–20 August 1992.

[BM94] Daniel Bleichenbacher and Ueli M. Maurer. Directed acyclic graphs, one-way functions and digital signatures. In Yvo G. Desmedt, editor, *Advances in Cryptology—CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 75–82. Springer-Verlag, 21–25 August 1994.

[BM96a] Daniel Bleichenbacher and Ueli M. Maurer. On the efficiency of one-time digital signatures. In Kwangjo Kim and Tsutomu Matsumoto, editors, *Advances in Cryptology—ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*, pages 145–158, Kyongju, Korea, 3–7 November 1996. Springer-Verlag.

[BM96b] Daniel Bleichenbacher and Ueli M. Maurer. Optimal tree-based one-time digital signature schemes. In Claude Puech and Rüdiger Reischuk, editors, *Symposium on Theoretical Aspects of Computer Science*, volume 1046 of *Lecture Notes in Computer Science*, pages 363–374. Springer-Verlag, 1996.

[DBP96] Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. RIPEMD-160: A strengthened version of RIPEMD. In D. Gollmann, editor, *Fast Software Encryption. Third International Workshop Proceedings*, volume 1039 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.

[EGM96] Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital signatures. *Journal of Cryptology*, 9(1):35–67, Winter 1996.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988.

[Lam79] Leslie Lamport. Constructing digital signatures from a one way function. Technical Report CSL-98, SRI International, October 1979.

[Mer82] Ralph C. Merkle. *Secrecy, Authentication, and Public Key Systems*. UMI Research Press, 1982.

[Mer87] Ralph C. Merkle. A digital signature based on a conventional encryption function. In Carl Pomerance, editor, *Advances in Cryptology—CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 369–378. Springer-Verlag, 1988, 16–20 August 1987.

[Mer89] Ralph C. Merkle. A certified digital signature. In G. Brassard, editor, *Advances in Cryptology—CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer-Verlag, 1990, 20–24 August 1989.

[MM82] Carl H. Meyer and Stephen M. Matyas. *Cryptography: A New Dimension in Computer Data Security*. John Wiley & Sons, 1982.

[NIS95] FIPS publication 180-1: Secure hash standard, April 1995. Available from `http://csrc.nist.gov/fips/`.

[Per01] Adrian Perrig. The BiBa one-time signature and broadcast authentication protocol. In P. Samarati, editor, *Eighth ACM Conference on Computer and Communication Security*, pages 28–37. ACM, 2001.

[Rab78] Michael O. Rabin. Digitalized signatures. In Richard A. Demillo, David P. Dobkin, Anita K. Jones, and Richard J. Lipton, editors, *Foundations of Secure Computation*, pages 155–168. Academic Press, 1978.

[Roh99] Pankaj Rohatgi. A compact and fast hybrid signature scheme for multicast packet authentication. In *Sixth ACM Conference on Computer and Communication Security*, pages 93–100. ACM, November 1999.

[Vau92] Serge Vaudenay. One-time identification with low memory. In P. Camion, P. Charpin, S. Harari, and G. Cohen, editors, *Proceedings of EUROCODE '92*, Lecture Notes in Computer Science, pages 217–228. Springer-Verlag, 1992.

# A Definitions of Subset-Resilience

Let $\mathcal{H} = \{H_{i,t,k}\}$ be a family of functions, where $H_{i,t,k}$ maps an input of arbitrary length to a subset of size at most $k$ of the set $\{0, 1, \ldots, t-1\}$. (Note that for each $t$ and $k$, $\mathcal{H}$ contains a number of functions, which are indexed by $i$.) Moreover, assume that there is a polynomial-time algorithm that, given $i, 1^t, 1^k$ and $M$, computes $H_{i,t,k}(M)$.

**Definition 1.** We say that $\mathcal{H}$ is *$r$-subset-resilient* if, for every probabilistic polynomial-time adversary $A$,

$$\Pr_i \left[ (M_1, M_2, \ldots, M_{r+1}) \leftarrow A(i, 1^t, 1^k) \text{ s.t. } H_{i,t,k}(M_{r+1}) \subseteq \bigcup_{j=1}^{r} H_{i,t,k}(M_j) \right] < negl(t,k) \,.$$

Fix a distribution $D$ on the space of all inputs to $H$ (i.e., on the space of messages).

**Definition 2.** We say that $\mathcal{H}$ is *$r$-target-subset-resilient* if, for every probabilistic polynomial-time adversary $A$,

$$\Pr_i \left[ M_1, M_2, \ldots, M_r \xleftarrow{R} D;\ M_{r+1} \leftarrow A(i, 1^t, 1^k, M_1, \ldots, M_r) \text{ s.t. } H_{i,t,k}(M_{r+1}) \subseteq \bigcup_{j=1}^{r} H_{i,t,k}(M_j) \right] < negl(t,k) \,.$$

Note that subset-resilience is a stronger property than target-subset-resilience. The former is needed to prove security of our scheme against adaptive chosen message attacks, while the latter suffices for random message attacks.

For $r = 1$, the above definitions can be realized using only collision-resilient hash families. Namely, if $\mathcal{H}$ is a collision-resilient hash function family, and $S$ is the subset-selection algorithm of Section 2, then $\{H(S)\}_{H \in \mathcal{H}}$ is a 1-subset-resilient family. Similarly, if $\mathcal{H}$ is target-collision-resilient (a.k.a. "universal one-way"), then $\{H(S)\}_{H \in \mathcal{H}}$ is 1-target-subset-resilient.

For $r > 1$, realizing the above definitions using only common complexity-theoretic assumptions (without random oracles) is an open problem. We believe, however, that it is reasonable to assume that selecting subsets via cryptographic hash functions, such as SHA-1 or RIPEMD-160, will satisfy these definitions for small values of $r$.