Domain Extension for Enhanced Target Collision-Resistant Hash Functions

Ilya Mironov

Microsoft Research, Silicon Valley Campus

Abstract. We answer the question of Reyhanitabar et al. from FSE'09 of constructing a domain extension scheme for enhanced target collision-resistant (eTCR) hash functions with sublinear key expansion. The eTCR property, introduced by Halevi and Krawczyk [1], is a natural fit for hash-and-sign signature schemes, offering an attractive alternative to collision-resistant hash functions. We prove a new composition theorem for eTCR, and demonstrate that eTCR compression functions exist if and only if one-way functions do.

1 Introduction

Hash functions are the staple of cryptographic protocols. Mapping out the necessary and sufficient assumptions on hash functions is an important research program, with implications for constructions and modes of operations of hash functions. While collision resistance is the strongest and arguably the most universally applicable property one may expect of a hash function, weaker security properties are often sufficient and may be easier to design for.

The focus of this paper is on a new security property of hash functions put forth by Halevi and Krawczyk [1] for the purpose of strengthening hash-and-sign signatures against collision-finding attacks. The standard hash-and-sign paradigm, which is the basis of virtually all practical signature schemes, calls for hashing the message using a collision-resistant hash function and then signing the result. Should collisions in the hash function be found, the hash-and-sign signature becomes obviously insecure as a signature on one of the colliding messages is a valid signature on the other. Halevi and Krawczyk propose hashing the message under a randomly chosen key and then including the key as part of the signature. The new security property of the hash function, under which the scheme preserves security of the underlying fixed input-length signature scheme, is called *enhanced target collision-resistance.*

They demonstrate practical and theoretical advantages of this approach as well as a concrete instantiation of the hash function satisfying this property based on a randomized Merkle-Damgård construction, with a minimal computational overhead. The construction can be proved secure under several non-standard assumptions on the fixed input-length keyless compression function. We argue that a compelling alternative is starting with a keyed eTCR compression function with fixed input length and extending its domain with a dedicated construction.

The question of domain extension for eTCR was first considered by Reyhanitabar et al. [2] who found that only one of many known domain extension schemes preserves the eTCR property and this construction expands the key linearly, rendering it impractical for application in signatures. In the main contribution of this paper we propose a new eTCR-preserving domain extension scheme with logarithmic key expansion thus settling the open question from [2].

The organization of the paper is as follows. In Sections 2, 3, 4 we survey prior and related work, recall standard definitions, and give an overview of domain extension techniques. In Section 5 we consider the question of placing eTCR in the complexity-theoretic hierarchy of hardness assumptions and prove that eTCR compression functions may be constructed from one-way functions. Finally, we introduce a new domain extension scheme in Section 6 and instantiate it in Section 7.

2 Related Work

The hash-and-sign paradigm for cryptographic signatures goes back to the Rabin signature scheme [3], where the hash function was used for message compression and, rather presciently, for input randomization (see also [4]). The notion of collision-resistant hash functions, which are essential for the signatures' security, was formally defined by Damgård [5]. All standards for digital signatures in use today are based on some variant of the hash-and-sign paradigm. In fact, first standards of dedicated hash functions, such as MD2, MD5, or SHA [6,7,8], were explicitly aimed at securing digital signature schemes.

Security of most practical signature schemes cannot be argued based on the collision-resistance property of the hash function alone. Instead, proofs are given in the random oracle model, where the hash function is replaced by an ideal functionality with oracle access [9,10].

As part of a larger research program of reducing hardness assumptions necessary for proving security of various cryptographic primitives, a series of seminal papers reduced existence of secure signatures to that of one-way functions [11,12,13] (see also Katz and Koo [14] for a complete proof and Haitner et al. [15] for an alternative proof of the last reduction). The crucial intermediate step of the construction, proposed by Naor and Yung, is *universal one-way hash functions* (UOWHFs), also known as target collision-resistant (TCR) hashes.

TCR hashes appear to be a fundamentally weaker primitive than collision-resistant hashes, and thus may be easier to construct, as evidenced by the work of Simon [16]. He showed that collision-resistant hashes cannot be built using one-way functions in a black-box manner, as opposed to target collision-resistant hashes that can. In practice, collision-resistance of standardized hash functions has been under assault recently, starting with remarkable attacks on the MD and SHA families by Wang et al. [17,18,19,20].

The attacks spurred interest in using hash functions that are less fragile than collision-resistant hashes. Even before that, the Cramer-Shoup signature scheme [21], which was a first efficient short signature scheme provable in the standard model, included a TCR hash as an option. The main difficulty in using TCR hashes as a drop-in replacement for broken or vulnerable collision-resistant functions is in handling the key, since the TCR hashes are keyed unlike keyless collision-resistant hashes.

Moreover, the key of a TCR hash cannot be chosen ahead of time: to take advantage of its security guarantee the key must be chosen by the signer, signed and communicated as part of the signature. More concretely, if $\sigma(\cdot)$ is a secure (existentially unforgeable [11]) signature scheme for fixed-length inputs and $H_k(\cdot)$ is a TCR hash, $\langle\sigma(k||H_k(M)), k\rangle$ where k is chosen at signing, will be a secure signature scheme for variable-length messages.

To address the problem of inflating the signature length, Mironov [22], generalized by Pasini and Vaudenay [23], suggests reusing the randomness already present in the signature scheme to key the TCR hash. A different approach, which is the main motivation for the present work, is due to Halevi and Krawczyk [1]. They propose a new security definition for hash functions, called *enhanced target collision-resistance* (eTCR), which allows one to leave out the hash function key from the signature's input. Namely, if the keyed hash function $H_k(\cdot)$ satisfies the new definition, the following signature—$\langle\sigma(H_k(M)), k\rangle$—inherits security of the $\sigma(\cdot)$ scheme.

Constrained by backwards compatibility, Halevi and Krawczyk ingeniously replace legacy keyless hash functions in signature schemes such as RSA or DSA with a keyed eTCR function *without changing existing implementations of the signing algorithm.* Instead, they transform the message with a randomly chosen key, sign the output of the transformation, and append the key to the signature. Thus, their signature scheme takes the form of $\langle\sigma(H(\mathrm{RMX}_k(M))), k\rangle$, where RMX is the keyed randomization scheme. A concrete specification of the RMX transform is available as a NIST special publication [24] and an IETF Internet draft [25]. The security of the combined signature scheme follows from existential unforgeability of $\sigma(\cdot)$ and from $H(\mathrm{RMX}_k(\cdot))$'s being eTCR if H is an iterative (based on the Merkle-Damgård scheme [26,27]) hash function whose compression function satisfies one of several non-standard cryptographic properties. For security of the RMX transform applied to blockcipher-based Davies-Meyer hash functions (i.e., most of current standards) see Gauravaram and Knudsen [28].

Given its practical significance, we consider the eTCR property as a natural and intriguing extension of target collision-resistance that can be studied on its own. Two fundamental questions arise in connection with a new definition of security for hash functions: its place in the hierarchy of hardness assumptions, and existence and efficiency of domain extension schemes. In other words, what is the simplest primitive eTCR can be reduced to, and once we have a fixed-length input eTCR, how can we apply it to arbitrary-length inputs? The first question was previously considered by Yasuda [29], which we revisit for *compressing* eTCR in Section 5. The second question was initially raised by Reyhanitabar et al. [2], who found that most known domain extension schemes do not preserve the eTCR property. The only construction on their list that does, achieves so by expanding the key linearly with the size of the message. They conclude the paper by leaving open the problem of constructing a (key-length) efficient eTCR-preserving domain extension scheme. We answer it in Section 6.

3 Definitions

To simplify notation we state standard definitions of security properties for hashes in the asymptotic setting, parameterized with security parameter $\lambda \in \mathbb{N}$, which is omitted when clear from the context. A function of λ is *negligible* if it is less in absolute value than $1/|p(\lambda)|$ for any polynomial p and large enough λ.

Since we deal with both fixed and variable-input length functions, the definitions are given using abstract domain and range sets, which can be substituted for fixed-length binary strings $\{0,1\}^n$ or bounded-length $\{0,1\}^{<n}$ as needed. We always assume inputs to be of length polynomial in λ and functions to be computable in polynomial time.

Definition 1 (Target collision-resistance—TCR [12]). $\{H_\lambda\}_{\lambda\in\mathbb{N}}$ *is a* target collision-resistant *family of functions* $H_\lambda\colon \mathcal{K}_\lambda \times \mathcal{D}_\lambda \to \mathcal{R}_\lambda$ *if for any polynomial-time adversary consisting of two randomized algorithms* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *the probability of outputting* `success` *in the following experiment is negligible:*

```
$\mathbf{Exp}^{\mathrm{tcr}}_{H,\mathcal{A}}(\lambda)$:
$(M, state) \leftarrow \mathcal{A}_1(1^\lambda)$;
$k \xleftarrow{\$} \mathcal{K}_\lambda$;
$M' \leftarrow \mathcal{A}_2(1^\lambda, k, state)$;
output success if $H_\lambda(k,M) = H_\lambda(k,M')$ and $M \neq M'$.
```

The probability is taken over the adversary's random tape and the choice of k.

For notation's brevity, we often write the hash function's key as a subscript also omitting the security parameter, as in $H_k(M)$ instead of $H_\lambda(k, M)$. When the function's domain $\mathcal{D}$ is a direct product of two or more sets, say $\mathcal{X} \times \mathcal{Y}$, we may write the input of the hash function as several arguments, such as $H_k(x, y)$, where $(x, y) \in \mathcal{X} \times \mathcal{Y} = \mathcal{D}$. The difference is purely syntactical.

Definition 2 (Enhanced target collision-resistance—eTCR [1]). *We call* $\{H_\lambda\}_{\lambda\in\mathbb{N}}$ *an* enhanced target collision-resistant *family of functions* $H_\lambda\colon \mathcal{K}_\lambda \times \mathcal{D}_\lambda \to \mathcal{R}_\lambda$ *if for any polynomial-time adversary consisting of two randomized algorithms* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *the probability of outputting* `success` *in the following experiment is negligible:*

```
$\mathbf{Exp}^{\mathrm{etcr}}_{H,\mathcal{A}}(\lambda)$:
$(M, state) \leftarrow \mathcal{A}_1(1^\lambda)$;
$k \xleftarrow{\$} \mathcal{K}_\lambda$;
$(k', M') \leftarrow \mathcal{A}_2(1^\lambda, k, state)$;
output success if $H_\lambda(k,M) = H_\lambda(k',M')$ and $(k,M) \neq (k',M')$.
```

The probability is taken over the adversary's random tape and the choice of k.

In other words, in the $\mathbf{Exp}^{\mathrm{tcr}}_{H,\mathcal{A}}(\lambda)$ experiment (the *TCR game*) the adversary commits to a message M, receives a randomly sampled key k, and is tasked with producing a message M' such that it collides with M under H_k: $H_k(M) = H_k(M')$. The adversary in the eTCR game is more powerful—after committing to M and receiving k as before, he wins the game if he can find M' and possibly a *different* key k' such that $H_k(M) = H'_k(M')$ subject to the condition that $(k, M) \neq (k', M')$.

We note that our versions of the definitions of TCR and eTCR do not explicitly require the functions to be compressing. In fact, since we allow variable-length inputs, the functions may sometimes map short inputs into longer strings. However, throughout the paper the ranges $\mathcal{R}_\lambda$ are always assumed to consist of bit-strings of some fixed length m (depending on λ) and the domains $\mathcal{D}_\lambda$ typically include strings longer than m.

4 Overview of Domain Extension Schemes for TCR

The most common method of designing hash functions is to construct a fixed input-length compression function and then extend its domain by repeated applications via some composition scheme. By far the best known such scheme is Merkle-Damgård [26,27] that provably and without any loss in exact security extends a collision-resistant compression function. It is important to note limitations of this approach: (1) the extended-domain hash is only as secure as the underlying compression function; (2) the domain extension scheme may not preserve security properties other than those for which it was proved secure; (3) exact security (i.e., the one most relevant in practice) may deteriorate under composition. Several recent papers stress these points regarding the Merkle-Damgård scheme: multi-block collision-finding attacks on MD5 and SHA-0 [18,19,30] span several applications of the compression function; multicollisions, second-preimage and other attacks may take advantage of the Merkle-Damgård iterative structure [31,32,33].

More relevantly and also reinforcing point (2) from the above, target collision-resistance is not preserved under the Merkle-Damgård iterative composition [34], which motivates construction of dedicated TCR-preserving domain extension schemes. Two such schemes appear in the Naor-Yung paper, the first being a Merkle-Damgård-like sequential composition of independently keyed compression functions (linear hash), and the second similar to the Wegman-Carter tree-based method [35] (basic tree, according to [34]). Both schemes expand the key, whose length (the shorter, the better) is an important characteristic of a TCR function. Reducing key expansion in sequential and parallelizable settings was the subject of several papers starting with [34].

Remarkably, both the linear hash and basic tree composition schemes preserve the eTCR property. The first statement was shown in [2] and the second, observed by Dodis and Haitner [36], is implied by the following argument. Each level of the tree construction can be modeled as an independently keyed hash function, which is a concatenation of multiple eTCR and thus an eTCR itself. The basic tree construction is eTCR by applying the same argument as in the proof of the linear hash scheme to the composition of layers.

Let the fixed input-length compression function be $H\colon \{0,1\}^{\ell} \times \{0,1\}^{m} \to \{0,1\}^{n}$, where ℓ is the key length, and let the message length be $L > m$. Linear hash expands the key to $\ell\lceil L/(m-n)\rceil$. Naor-Yung basic tree for hashing messages of size L results in key length $\ell\lceil \log_{m/n}(L/n)\rceil$, improved for large ℓ to $\ell + m\lceil \log_{m/n}(L/n)\rceil$ by Bellare and Rogaway with the XOR tree construction [34]. A sequential composition scheme due to Shoup [37] expands the key to $\ell + n\lceil \log_2(L/(m-n))\rceil$, which is an improvement over the XOR tree construction by approximately a factor of $m/n \log_2 m/n$. The Shoup construction's

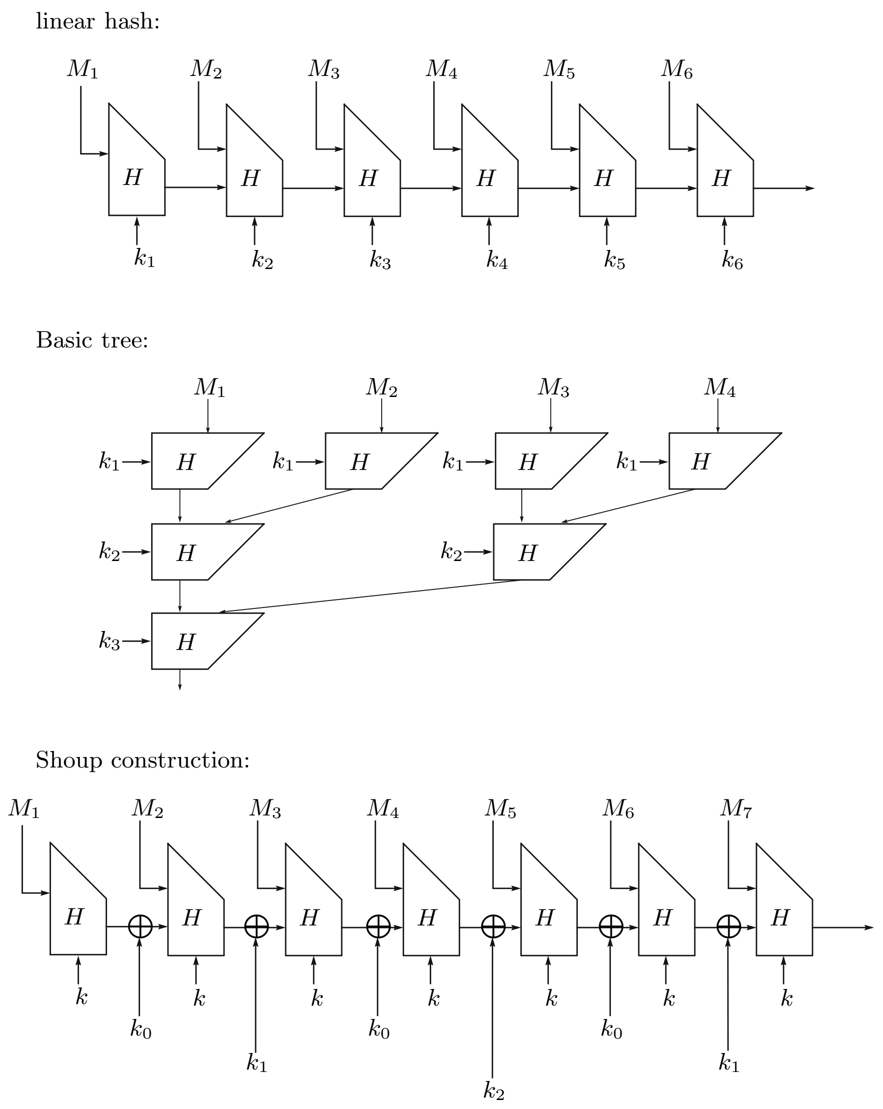

Fig. 1. Extenders for TCR hashes: linear, basic tree, and Shoup constructions

key expansion was proved optimal in a certain restricted model by Mironov [38]. The construction and the proof of optimality were generalized to DAG-based constructions (including tree-based schemes) by Sarkar [39,40].

Depending on the parameters of the compression function and the length of the message, one of the following composition schemes is the least key-expanding: sequential composition, basic tree, or Shoup's construction. The three schemes are illustrated in Figure 1.

Two comments are in order. First, we leave the issue of padding and length encoding for Section 7. Second, we present the linear hash and the Shoup construction without the usual initial value (IV) constant (see also [41]). Instead, the first application of the compression function processes the initial m-bit long block of the message. For the narrowly defined purpose of TCR domain extension these constructions can be proved secure using standard methods. On the upside, our versions have the pleasant property of defaulting to the compression function when the message (after length encoding and padding) is m-bit long and of enabling chaining composition of several such functions (see Figure 2 below).

For more comprehensive treatment of domain extension schemes with an eye towards *simultaneously* preserving many cryptographic properties of hash functions, such as second-preimage resistance, multi-collision resistance, random oracleness, etc., we refer to several recent papers [42,43,44] on domain extension for keyed and keyless functions.

5 From One-Way Hashes to eTCR

Motivated by compatibility with existing standards and APIs, Halevi and Krawczyk describe an elegant eTCR construction framed as a message-preprocessing scheme. Indeed, the composition of the RMX transform (pronounced *remix*), defined as $\mathrm{RMX}(r, M) = (r, M_1 \oplus r, \ldots, M_l \oplus r)$, with a Merkle-Damgård hash function satisfying certain properties is eTCR. They prove security of the composition based on either one of two rather strong assumptions on the compression function, one of which implies existence of collision-resistant hashes, and the other is a property of the composition scheme itself. In light of known difficulty of constructing (plain) TCR functions from one-way functions without key expansion, it is unlikely that RMX can be proved secure based on a property reducible to one-way functions.

We pursue a different approach, starting with TCR hashes, which in turn can be reduced to one-way functions (see also [29] for related results). Before presenting our construction, we recall the Naor-Yung method of building TCR compressing functions (originally introduced as UOWHFs) from one-way permutations, and demonstrate that the resulting function is *not* an eTCR.

Select a bijection between $\mathrm{GF}(2^n)$ and $\{0,1\}^n$ by fixing an irreducible polynomial over $\mathbb{F}_2$ of degree n. Let $\pi\colon \{0,1\}^n \to \{0,1\}^n$ be a one-way permutation and let $g_{a,b}\colon \{0,1\}^n \to \{0,1\}^{n-1}$ be the function defined as $g_{a,b}(x) = \mathit{chop}(ax+b)$, where $x, a, b \in \{0,1\}^n$, arithmetic is done in $\mathrm{GF}(2^n)$, and *chop* drops the last bit of its input. Naor and Yung prove that the following composition function is target collision-resistant: $H_{a,b}(x) = g_{a,b}(\pi(x))$. Indeed, by assuming the opposite, we find a pre-image under π of a given z chosen uniformly at random from $\{0,1\}^n$ as follows. First, the adversary produces some $x \in \{0,1\}^n$, we then choose $a, b \in \{0,1\}^n$ such that $g_{a,b}(\pi(x)) = g_{a,b}(z)$. If the adversary succeeds in finding $y \neq x$ that collides with x under $H_{a,b}(\cdot)$, it means that $\pi(y) = z$, since $H_{a,b}(x)$ has exactly two preimages under $g_{a,b}(\cdot)$, one of which is $\pi(x)$ and the other is z. It contradicts π's one-wayness.

The above construction compresses the input by one bit. Naor and Yung prove that a sequential composition of independently-keyed TCR hashes is also TCR, and thus one can achieve arbitrary (polynomial) compression ratio.

We reproduce the proof here because $h_{a,b}(\cdot)$ fails as enhanced TCR for the same reason it can be proved (plain) TCR. The function is defined in such a way that by choosing the key it can be forced to take any given value on any fixed input. Likewise, a concrete instantiation of a TCR function based on hardness of the subset-sum problem due to Impagliazzo and Naor [45] is trivially not an eTCR.

However, there is a simple transformation that converts a TCR function into an enhanced TCR. If $h_k(\cdot)$ is a TCR, then $\hat{H}_k(x) = H_k(x)||k$ is eTCR. More formally,

Proposition 1. *If $\{H_\lambda\}$ is a TCR family of functions $H_\lambda\colon \mathcal{K}_\lambda \times \mathcal{D}_\lambda \to \mathcal{R}_\lambda$, the following family $\hat{H}_\lambda\colon \mathcal{K}_\lambda \times \mathcal{D}_\lambda \to \mathcal{R}_\lambda \times \mathcal{K}_\lambda$ defined as*

$$\hat{H}(k, M) = H(k, M)||k$$

is eTCR.

Proof. Assuming the opposite, there is an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that wins the eTCR game against $\hat{H}$. Let the output of $\mathcal{A}_1$ be M, the random key be k, and the output of $\mathcal{A}_2$ be (k', M'). Since $\hat{H}_k(M) = H_k(M)||k$ and $\hat{H}_k(M) = \hat{H}'_k(M')$, it means that $k = k'$, and the adversary $\mathcal{A}$ wins the TCR game against H with the same probability it wins the eTCR game against $\hat{H}$. □

The TCR-to-eTCR transform as applied to the basic Naor-Yung construction *does not* yield a compressing eTCR, because the key of the original construction is longer than the input. For instance, to achieve the compression ratio of two, the key must be quadratic in the input length. However, domain extenders for TCR with sublinear key expansion, surveyed in Section 4, do result in TCR hashes where the length of the key concatenated with the output is less than the hash function's input length. Combining this with Proposition 1, and invoking the result of Rompel [13,14] stating that one-way functions are sufficient for TCR compressing hashes, we establish the following:

Theorem 1. *Compressing eTCR hash functions exist if and only if one-way functions do.*

To the best of our knowledge, the above theorem is the first application of domain extension schemes with sublinear key expansion in complexity-theoretic treatment of cryptographic hash functions.

6 Domain Extension for eTCR

As the main contribution of the paper we construct a domain extender for eTCR. The construction is recursive and uses domain extension schemes for TCR. It is based on the observation that a composition of a TCR function with an independently keyed eTCR function is eTCR. Since an eTCR compression function is also (plain) TCR, whose domain we know how to extend, it suffices to iterate the composition scheme until the input into the eTCR function becomes shorter than the compression function's input or a linear hash extension scheme can be applied.

Suppose we are given a TCR hash H^{tcr} and an eTCR hash F^{etcr}. We argue that the following composition function is eTCR:

$$G_{k_1,k_2}(M) = F^{\mathrm{etcr}}_{k_2}(H^{\mathrm{tcr}}_{k_1}(M), k_1).$$

Indeed, if after committing to M and receiving (k_1, k_2), the adversary finds a collision of the type $G_{k_1,k_2}(M) = G_{k_1',k_2'}(M')$, where $M \neq M'$, $k_1 = k_1'$, and $H^{\mathrm{tcr}}_{k_1}(M) = H^{\mathrm{tcr}}_{k_1}(M')$, it (intuitively) means that the adversary broke target collision-resistance of the H^{tcr} function. If $k_1 \neq k'$ or $H^{\mathrm{tcr}}_{k_1}(M) \neq H^{\mathrm{tcr}}_{k_1}(M')$, we would use the adversary to win the eTCR game against F^{etcr}. More formally:

Theorem 2. *If $H^{\mathrm{tcr}}_\lambda \colon \mathcal{K}^1_\lambda \times \mathcal{D}_\lambda \to \mathcal{R}^1_\lambda$ is a TCR family indexed by security parameter λ and $F^{\mathrm{etcr}}_\lambda \colon \mathcal{K}^2_\lambda \times \mathcal{R}^1_\lambda \times \mathcal{K}^1_\lambda \to \mathcal{R}^2_\lambda$ is eTCR, then the following family of functions $G \colon \mathcal{K}^1_\lambda \times \mathcal{K}^2_\lambda \times \mathcal{D}_\lambda \to \mathcal{R}^2_\lambda$ is eTCR:*

$$G_{k_1,k_2}(M) = F^{\mathrm{etcr}}_{k_2}(H^{\mathrm{tcr}}_{k_1}(M), k_1).$$

Proof. Assume towards a contradiction that there is an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ winning the eTCR game against G. Let (M, state) be the output of $\mathcal{A}_1$, and let $\mathcal{A}_2$ produce a collision $G_{k_1,k_2}(M) = G_{k_1',k_2'}(M')$ given random (k_1, k_2). We classify the collisions into two types: inner collisions, where $M \neq M'$, $k_1 = k_1'$ and $H^{\mathrm{tcr}}_{k_1}(M) = H^{\mathrm{tcr}}_{k_1'}(M')$, and outer collisions (all others). If we guess at random which of these two cases takes place, we succeed with probability at least a half.

Case I: Inner collision. In this case, we may use the adversary to break the H^{tcr} function. Define algorithm $\mathcal{B}^{\mathrm{I}} = (\mathcal{B}^{\mathrm{I}}_1, \mathcal{B}^{\mathrm{I}}_2)$ as follows:

```
Algorithm $\mathcal{B}^{\mathrm{I}}_1(1^\lambda)$:
1. Run $(M, \mathit{state}) \leftarrow \mathcal{A}_1(1^\lambda)$.
2. Output $(M, \mathit{state})$.

Algorithm $\mathcal{B}^{\mathrm{I}}_2(1^\lambda, \mathit{state}, k_1)$:
1. Pick random $k_2 \xleftarrow{\$} \mathcal{K}_2$.
2. $(M', k_1', k_2') \leftarrow \mathcal{A}_2(1^\lambda, \mathit{state}, k_1, k_2)$.
3. Fail if this is not an inner collision.
4. Output $M'$.
```

By the definition of an inner collision (i.e., $M \neq M'$, $k_1 = k_1'$, and $H^{\mathrm{tcr}}_{k_1}(M) = H^{\mathrm{tcr}}_{k_1'}(M')$), the algorithm $\mathcal{B}_{\mathrm{I}}$ outputs a valid collision on H^{tcr}.

Case II: Outer collision. In this case, we attack the F^{etcr} function. Let $\mathcal{B}^{\mathrm{O}} = (\mathcal{B}^{\mathrm{O}}_1, \mathcal{B}^{\mathrm{O}}_2)$ be the following:

```
Algorithm $\mathcal{B}_1^{\mathrm{O}}(1^\lambda)$:
1. Run $(M, state) \leftarrow \mathcal{A}_1(1^\lambda)$.
2. Pick random $k_1 \overset{\$}{\leftarrow} \mathcal{K}_1$.
3. Output $(\langle H_{k_1}^{\mathrm{tcr}}(M), k_1\rangle, k_1||state)$.
```

```
Algorithm $\mathcal{B}_2^{\mathrm{O}}(1^\lambda, k_1||state, k_2)$:
1. $(M', k_1', k_2') \leftarrow \mathcal{A}_2(1^\lambda, state, k_1, k_2)$.
2. Fail if this is not an outer collision.
3. Output $(k_2', \langle H_{k_1'}^{\mathrm{tcr}}(M'), k_1'\rangle)$.
```

If $\mathcal{A}$ succeeds, we know that $G_{k_1,k_2}(M)=G_{k_1',k_2'}(M')$ and $(k_1, k_2, M) \neq (k_1', k_2', M')$. To conclude, we must verify that this gives a valid collision on F^{etcr}, where the colliding key-message pairs is $(k_2, \langle H_{k_1}^{\mathrm{tcr}}(M), k_1\rangle)$ and $(k_2', \langle H_{k_1'}^{\mathrm{tcr}}(M'), k_1'\rangle)$. By definition of an outer collision, at least one of the following holds true: $M = M'$, or $k_1 \neq k_1'$, or $H_{k_1}^{\mathrm{tcr}}(M) \neq H_{k_1'}^{\mathrm{tcr}}(M')$. In the last two cases, the inputs into F^{etcr} are obviously distinct. If $M = M'$, then by definition of the eTCR game won by $\mathcal{A}$, we know that $(k_1, k_2) \neq (k_1', k_2')$, which results in a valid collision on F^{etcr}. □

Remark 1. The domain of the outer function F^{etcr} in the theorem statement is $\mathcal{R}^1 \times \mathcal{K}^1$, whereas in practice the function's input is most likely to be a bit string. Thus, to apply the theorem one has to ensure that F^{etcr}'s input can be uniquely parsed into the output of H^{tcr} and its key. It is indeed the case when either the output of the function (the most common option) or the key have fixed length, or alternatively, by using an encoding scheme where the boundary between the two strings can be unambiguously identified.

The above composition theorem gives a domain extension scheme for eTCR except for one potential problem: Since there are no known domain extenders for TCR hashes without key expansion, the key k_1, which is concatenated with H's output and fed into the eTCR function F, may be longer than the compression function's input length. The construction may be applied recursively, or it may terminate by using the linear hash domain extension scheme for eTCR (see below). If one uses the Shoup construction for H and the linear hash for F, the total key expansion is $(\ell + n\lceil \log_2(L/(m-n))\rceil) \times (1 + \ell/(m-n))$.

7 Length Variability and Concrete Scheme

Before we present a concrete instantiation of the scheme proven secure in Theorem 2 we must address the question of message padding and length encoding by the underlying schemes. As is, domain extenders for TCR hashes from Figure 1 either require message pre-processing or become insecure when applied to variable-length messages. A generic method for domain extension to variable-length inputs is described by Bellare and Rogaway [34] and requires one additional application of an independently keyed compression function. To streamline the construction, we encode the message length into the input of the eTCR function F^{etrc}, which allows us to use domain extenders satisfying a weaker definition of TCR* (defined in [2]), where the function accepts variable-length inputs but the adversary in the TCR game is restricted to finding a collision of equal-length messages. Namely, assuming that the keyed function of three inputs F^{etcr} is eTCR and H^{tcr} is TCR*, the following function is eTCR:

$$G^*_{k_1,k_2}(M) = F^{\text{etcr}}_{k_2}(H^{\text{tcr}}_{k_1}(M), k_1, |M|).$$

The proof is analogous to Theorem 2. In our construction below we make sure that the input to F^{etcr} be uniquely parsed into three parts.

The following construction, which consists of pre-processing followed by application of independently keyed compression functions, is very similar to the linear hash scheme of [34]. They differ in two important aspects: the IV, which is replaced with input material in our construction, and handling of the message length, which is encoded in the last block. A proof of the claim that the resulting construction is eTCR a straightforward adaption of Theorem 7 from the full version of [2].

```
**Pre-processing function**
Input: message $M$, length $len < 2^d$
Output: blocks $M_1, \ldots, M_t$
    format $M$ as $M_1, \ldots, M_t$, where (a) $|M_1| = |M|$ if $|M| < m$
                                            (b) $|M_1| = m$ if $|M| \geq m$
                                            (c) $|M_i| = m - n$ for $1 < i < t$
                                            (d) $|M_t| \leq m - n$
    $s \leftarrow \begin{cases} m & \text{if } t = 1 \\ m - n & \text{otherwise} \end{cases}$
    if $|M_t| > s - d$ then
        pad $M_t$ with zeros to $s$ bits
        $M_{t+1} \leftarrow$ empty
        $t \leftarrow t + 1$
    pad $M_t$ with zeros to $m - n - d$ bits
    $M_t \leftarrow M_t || [len]^d_2$, where $[len]^d_2$ is $len$ encoded as $d$-bit binary
```

The following example (Figure 2) is a domain extension scheme for the compression function $H\colon \{0,1\}^{128} \times \{0,1\}^{640} \to \{0,1\}^{256}$, which can be based on the compression function of SHA-256 with 128 input bits allocated for the key. The composite function takes input of length $256 + 4 \times 384 = 1792$ bits and expands the key to $3 \times 128 + 2 \times 256 = 896$ bits. The example illustrates two features of our construction. First, its two main components, the TCR* and eTCR functions, may be selected independently from each other. In particular,

```
**Linear hashing**
Given: function $F\colon \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^n$
Input: blocks $M_1, \ldots, M_t$; keys $k_1, \ldots, k_t$
Output: hash of length $n$ bits
    $h_1 \leftarrow F_{k_1}(M_1)$
    for $i := 2$ to $t$ do
        $h_i \leftarrow F_{k_i}(h_{i-1} || M_i)$
    output $h_t$
```

the choice of the inner function TCR* may depend on the message length (for instance, in our example a more keysize-efficient choice of the inner function would have been the linear hash; we prefer the Shoup construction for illustrative purposes). Second, rather than separately encoding the length of its input, as prescribed by the linear hash scheme, and the length of the message, called for by the construction of G^*, the outer eTCR function may only do the latter as long as the message length uniquely determines the length of eTCR's input. In fact, our pre-processing function accepts the length of the message as a separate input, which allows this optimization.

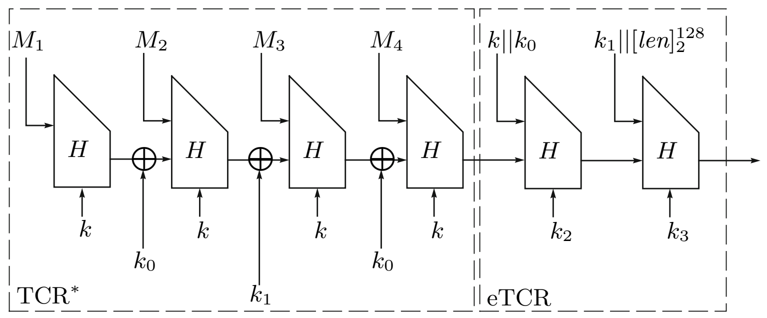

Fig. 2. Example of an eTCR domain extension function for $H\colon \{0,1\}^{128} \times \{0,1\}^{640} \to \{0,1\}^{256}$

8 Conclusion

We study the enhanced target collision-resistant (eTCR) property of hash functions introduced by Halevi and Krawczyk as a method of securing signature schemes in lieu of traditionally used collision-resistant hash functions [1]. While the definition was initially proposed to facilitate proof of security of the RMX transform, it is an interesting variant of the TCR property that may have applications on its own.

In our first contribution, we explore connections between TCR and eTCR hash functions, demonstrating that the TCR construction of Naor-Yung is *provably* not eTCR. On the other hand, eTCR hashes can be constructed from TCR compressing functions, placing them in the same complexity-theoretic class of functions that can be based on one-way functions in a black-box manner. It separates them from collision-resistant hashes that cannot be reduced to one-way functions or permutations via a black-box construction.

Secondly, we answer the question raised in [2] on constructing a key length-efficient domain extender for eTCR hashes by presenting a domain extension scheme with logarithmic key expansion.

Acknowledgements

We thank anonymous reviewers and attendees of the FSE'10 conference for their comments, and Yevgeniy Dodis and Iftach Haitner for their valuable observation that the basic tree construction preserves the eTCR property.

References

1. Halevi, S., Krawczyk, H.: Strengthening digital signatures via randomized hashing. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 41–59. Springer, Heidelberg (2006)
2. Reyhanitabar, M.R., Susilo, W., Mu, Y.: Enhanced target collision resistant hash functions revisited. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 327–344. Springer, Heidelberg (2009); Full version available at Cryptology ePrint Archive, Report 2009/506
3. Rabin, M.O.: Digitalized signatures and public-key functions as intractable as factorization. Technical Memo MIT/LCS/TR-212, MIT (January 1979)
4. Davies, D.W., Price, W.L.: The application of digital signatures based on public-key cryptosystems. In: Salz, J. (ed.) Proceedings of the Fifth Intl. Conference on Computer Communications, pp. 525–530 (1980)
5. Damgård, I.: Collision free hash functions and public key signature schemes. In: Price, W.L., Chaum, D. (eds.) EUROCRYPT 1987. LNCS, vol. 304, pp. 203–216. Springer, Heidelberg (1988)
6. Kaliski Jr., B.S.: The MD2 message-digest algorithm. RFC 1115, The Internet Engineering Task Force (April 1992)
7. Rivest, R.L.: The MD5 message-digest algorithm. RFC 1321, The Internet Engineering Task Force (April 1992)
8. National Institute of Standards and Technology: Secure hash standard (SHS) (May 1993)
9. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
10. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security, pp. 62–73 (1993)
11. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing 17, 281–308 (1988)
12. Naor, M., Yung, M.: Universal one-way hash functions and their cryptographic applications. In: Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing, May 15–17, pp. 33–43 (1989)
13. Rompel, J.: One-way functions are necessary and sufficient for secure signatures. In: Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing, May 14–16, 1990, pp. 387–394 (1990)
14. Katz, J., Koo, C.Y.: On constructing universal one-way hash functions from arbitrary one-way functions. J. Cryptology (to appear); Available on Cryptology ePrint Archive, Report 2005/328
15. Haitner, I., Holenstein, T., Reingold, O., Vadhan, S., Wee, H.: Universal one-way hash functions via inaccessible entropy. In: Advances in Cryptology—EUROCRYPT 2010 (to appear, 2010); Available on Cryptology ePrint Archive, Report 2010/120

16. Simon, D.R.: Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 334–345. Springer, Heidelberg (1998)
17. Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the hash functions MD4 and RIPEMD. In: [46], pp. 1–18
18. Wang, X., Yu, H.: How to break MD5 and other hash functions. In: [46], pp. 19–35
19. Wang, X., Yu, H., Yin, Y.L.: Efficient collision search attacks on SHA-0. In: [48], pp. 1–16
20. Wang, X., Yin, Y.L., Yu, H.: Finding collisions in the full SHA-1. In: [48], pp. 17–36
21. Cramer, R., Shoup, V.: Signature schemes based on the strong RSA assumption. ACM Trans. on Information and System Security (TISSEC) 3(3), 161–185 (2000)
22. Mironov, I.: Collision-resistant no more: Hash-and-sign paradigm revisited. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 140–156. Springer, Heidelberg (2006)
23. Pasini, S., Vaudenay, S.: Hash-and-sign with weak hashing made secure. In: Pieprzyk, J., Ghodosi, H., Dawson, E. (eds.) ACISP 2007. LNCS, vol. 4586, pp. 338–354. Springer, Heidelberg (2007)
24. Dang, Q.: Randomized hashing for digital signatures. NIST Special Publication 800-106, National Institute of Standards and Technology (February 2009)
25. Halevi, S., Krawczyk, H.: Strengthening digital signatures via randomized hashing. Internet Draft draft-irtf-cfrg-rhash-01, Internet Engineering Task Force (October 2007) (Work in progress)
26. Merkle, R.C.: One way hash functions and DES. In: [47], pp. 428–446
27. Damgård, I.: A design principle for hash functions. In: [47], pp. 416–427
28. Gauravaram, P., Knudsen, L.R.: On randomizing hash functions to strengthen the security of digital signatures, pp. 88–105
29. Yasuda, K.: How to fill up Merkle-Damgård hash functions. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 272–289. Springer, Heidelberg (2008)
30. Biham, E., Chen, R., Joux, A., Carribault, P., Lemuet, C., Jalby, W.: Collisions of SHA-0 and reduced SHA-1. In: [46], pp. 36–57
31. Joux, A.: Multicollisions in iterated hash functions. Application to cascaded constructions. In: Franklin, M.K. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 306–316. Springer, Heidelberg (2004)
32. Kelsey, J., Schneier, B.: Second preimages on n-bit hash functions for much less than 2^n work. In: [46], pp. 474–490
33. Kelsey, J., Kohno, T.: Herding hash functions and the Nostradamus attack. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 183–200. Springer, Heidelberg (2006)
34. Bellare, M., Rogaway, P.: ion-resistant hashing: Towards making UOWHFs practical. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 470–484. Springer, Heidelberg (1997)
35. Wegman, M.N., Carter, L.: New hash functions and their use in authentication and set equality. J. Comput. Syst. Sci. 22(3), 265–279 (1981)
36. Dodis, Y., Haitner, I.: Private communication
37. Shoup, V.: A composition theorem for universal one-way hash functions. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 445–452. Springer, Heidelberg (2000)
38. Mironov, I.: Hash functions: From Merkle-Damgård to Shoup. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 166–181. Springer, Heidelberg (2001)

39. Sarkar, P.: Masking based domain extenders for UOWHFs: Bounds and constructions. IEEE Transactions on Information Theory 51(12), 4299–4311 (2005)
40. Sarkar, P.: Construction of universal one-way hash functions: Tree hashing revisited. Discrete Applied Mathematics 155(16), 2174–2180 (2007)
41. Sarkar, P.: Domain extender for collision resistant hash functions: Improving upon Merkle-Damgård iteration. Discrete Applied Mathematics 157(5), 1086–1097 (2009)
42. Bellare, M., Ristenpart, T.: Multi-property-preserving hash domain extension and the EMD transform. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 299–314. Springer, Heidelberg (2006)
43. Andreeva, E., Neven, G., Preneel, B., Shrimpton, T.: Seven-property-preserving iterated hashing: ROX. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 130–146. Springer, Heidelberg (2007)
44. Bellare, M., Ristenpart, T.: Hash functions in the dedicated-key setting: Design choices and MPP transforms. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 399–410. Springer, Heidelberg (2007)
45. Impagliazzo, R., Naor, M.: Efficient cryptographic schemes provably as secure as subset sum. J. Cryptology 9(4), 199–216 (1996)
46. Cramer, R. (ed.): EUROCRYPT 2005. LNCS, vol. 3494. Springer, Heidelberg (2005)
47. Brassard, G. (ed.): CRYPTO 1989. LNCS, vol. 435. Springer, Heidelberg (1990)
48. Shoup, V. (ed.): CRYPTO 2005. LNCS, vol. 3621. Springer, Heidelberg (2005)