Multi-property-preserving Domain Extension Using Polynomial-based Modes of Operation

Jooyoung Lee* and John Steinberger**

Abstract. In this paper, we propose a new double-piped mode of operation for multi-property-preserving domain extension of MACs (message authentication codes), PRFs (pseudorandom functions) and PROs (pseudorandom oracles). Our mode of operation performs twice as fast as the original double-piped mode of operation of Lucks [15] while providing comparable security. Our construction, which uses a class of polynomial-based compression functions proposed by Stam [22, 23], makes a single call to a $3n$-bit to n-bit primitive at each iteration and uses a finalization function f_2 at the last iteration, producing an n-bit hash function $H[f_1, f_2]$ satisfying the following properties.

1. $H[f_1, f_2]$ is unforgeable up to $O(2^n/n)$ query complexity as long as f_1 and f_2 are unforgeable.
2. $H[f_1, f_2]$ is pseudorandom up to $O(2^n/n)$ query complexity as long as f_1 is unforgeable and f_2 is pseudorandom.
3. $H[f_1, f_2]$ is indifferentiable from a random oracle up to $O(2^{2n/3})$ query complexity as long as f_1 and f_2 are public random functions.

To our knowledge, our result constitutes the first time $O(2^n/n)$ unforgeability has been achieved using only an unforgeable primitive of n-bit output length. (Yasuda showed unforgeability of $O(2^{5n/6})$ for Lucks' construction assuming an unforgeable primitive, but the analysis is sub-optimal; in this paper, we show how Yasuda's bound can be improved to $O(2^n)$.)

In related work, we strengthen Stam's collision resistance analysis of polynomial-based compression functions (showing that unforgeability of the primitive suffices) and discuss how to implement our mode by replacing f_1 with a $2n$-bit key blockcipher in Davies-Meyer mode or by replacing f_1 with the cascade of two $2n$-bit to n-bit compression functions.

1 Introduction

The Merkle-Damgård transform has been the most popular method to build a cryptographic hash function from a fixed-size compression function. A major advantage of this construction is that it preserves collision resistance with an appropriate padding algorithm, allowing one to focus on the construction of secure compression functions. However, Joux showed that if computing collisions becomes somehow feasible for the underlying compression function, then the hash function may fail worse than expected: for a hash function based on a compression function of n-bit output, one can find a t-multicollision only with $O(2^{n/2} \log t)$ complexity, which is much smaller than $O(2^{(t-1)n/t})$ required for an ideal random function. This observation led to several generic attacks such as long-message second preimage attacks [13] and herding attacks [12]. Lucks observed that these weaknesses can be mitigated by increasing the size of the internal state and claimed that the internal state size should be seen as a security parameter of its own right [15]. Since a secure compression function of a larger output size might be harder to construct than the hash function itself, Lucks proposed to use a "narrow" compression function in a double-piped mode. In a subsequent paper [24], Yasuda rigorously analyzed the security of the double-piped mode of operation as a multi-property-preserving domain extension. Specifically, he showed that Lucks' double-piped

* The Attached Institute of Electronics and Telecommunications Research Institute, Daejeon, Korea, jlee05@ensec.re.kr

** Institute for Theoretical Computer Science, Tsinghua University, Beijing, China, jpsteinb@gmail.com
Supported by the National Natural Science Foundation of China Grant 60553001 and by the National Basic Research Program of China Grant 2007CB807900, 2007CB807901.

mode of operation preserves unforgeability up to $O(2^{5n/6})$ query complexity, and indistinguishability and indifferentiability both up to $O(2^n)$ query complexity. Moreover it was later noticed by several researchers that Yasuda's unforgeability bound could be increased to $O(2^n)$ with a slightly modified proof. (See Section 7.)

As such Lucks' construction turned out to provide nearly optimal security. However, the fact that Lucks' compression function uses two applications of a (fairly strong) primitive remains a drawback. Stam [22, 23] recently proposed a class of wide-pipe compression functions making a *single* call to an equal primitive (we call these *polynomial-based* compression functions). In this paper we analyze the security properties of double-piped modes using Stam's polynomial-based compression functions, focusing on MAC-preservation, PRF-preservation and PRO-preservation. Except for PRO-preservation (where we only achieve $O(2^{2n/3})$ security), our bounds are comparable to those found by Yasuda for Lucks' original construction (and even better for unforgeability, given the sub-optimality of Yasuda's bound in that case, though the "corrected" unforgeability bound exceeds ours by a factor of n) even though our construction has twice the rate.

Besides performance, a second concern that arises for Lucks' double-pipe construction is the rather strong primitive it assumes: a $3n$-bit to n-bit function (note that careful consideration is typically already given for the construction of $2n$-bit to n-bit compression functions from smaller or more available primitives). Here we also tackle this problem and show our double-piped polynomial-based mode can be implemented with a blockcipher of $2n$-bit key in Davies-Meyer mode, in either the ideal-cipher model or the weaker "unpredictable cipher" model (see Section 5) without significant loss of security. We also prove MAC-preservation and PRF-preservation for a compression function obtained by replacing the $3n$-bit to n-bit primitive with the cascade of two $2n$-bit to n-bit primitives. This latter result potentially opens the door to implementing the compression function with two calls to an n-bit key blockcipher in Davies-Meyer mode (which would be the first time, to our knowledge, that a $3n$-bit to $2n$-bit compression function using two calls to an n-bit key blockcipher is proved secure nearly up to the birthday bound).

Construction and Results. To keep our construction comparably general to Lucks' [15] and Yasuda's [24], we discuss a hash function obtained by iterating a $(2n+c)$-bit to $2n$-bit compression function $\phi[f_1]$ where the primitive f_1 used by the compression function is a $2n+c$-bit to n-bit compression function (the "expected" setting of the parameters is $c = n$).

The compression function $\phi[f_1]$ is illustrated in Fig. 1(a) for the case $c = n$ of a $3n$-bit to $2n$-bit compression function. Let $u \in \{0,1\}^{2n+c}$ and let $u_d||\ldots||u_0$ be the segmentation of u into n-bit blocks u_0, …, u_{d-1} and a block u_d of no more than n bits (so $d = \lceil \frac{2n+c}{n} \rceil - 1$). Then $\phi[f_1](u)$ is defined by

$$\phi[f_1](u) = x||y,$$

where

$$\begin{aligned} x &= f_1(u), \\ y &= u_d x^d + u_{d-1} x^{d-1} + \cdots + u_1 x + u_0, \end{aligned}$$

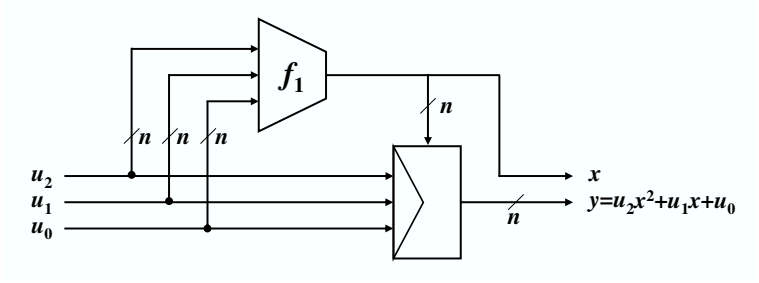
with all field operations taking place in $\mathbb{F}_{2^n}$ (and u_d being viewed as embedded in $\{0,1\}^n$). We call $\phi[f_1]$ a *polynomial-based compression function*. This design is due to Stam [22, 23].

Given an independent compression function $f_2 : \{0,1\}^{2n+c} \to \{0,1\}^n$, we define a hash function

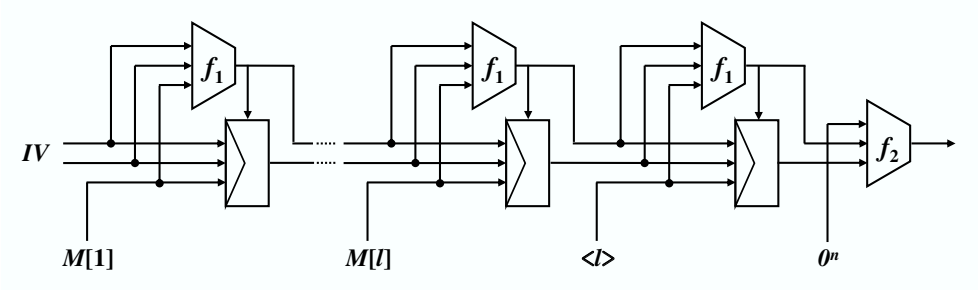
$$\begin{aligned} H[f_1, f_2] : \{0,1\}^* &\longrightarrow \{0,1\}^n \\ M &\longmapsto f_2\left(0^c||v\right), \end{aligned}$$

where $v = MD[\phi[f_1]](M)$, the Merkle-Damgård iteration of $\phi[f_1]$ on message M (with the usual "strengthened" padding for M that appends the length of the message—see Section 2 for details). The scheme is pictured for $c = n$ in Figure 1(b).

We comment at this point that our mode of operation uses two distinct primitives instead of a single primitive f_1 as do Lucks and Yasuda. As such, our construction explicitly follows the



(a) The compression function $\phi[f_1]$.



(b) The hash function $H[f_1, f_2]$.

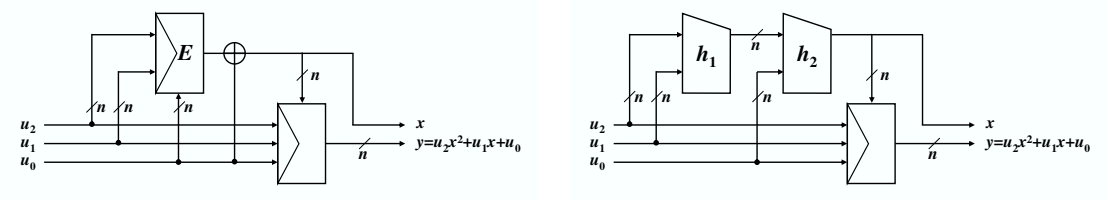
Fig. 1. The polynomial-based mode of operation for $c = n$.

framework of An and Bellare [1] for proving unforgeability whereas Yasuda adopts it implicitly: with some extra work, one can use $f_1 = f_2$ because the f_2-queries are (with very high likelihood) all independent from f_1-queries, due to the presence of the 0^c input segment. (This technique for reducing key material was first used by Maurer and Sjödin [19].) We opt for using two primitives because it simplifies the proofs and allows separation of the security properties required by f_1 and f_2 (the security requirements for f_1 being often much less than those for f_2).

The following points summarize our results on $\phi[f_1]$ and $H[f_1, f_2]$. For this summary we say that f_i is *unforgeable* to mean that a computationally bounded adversary with oracle access to f_i has low probability of predicting the output of f_i on an unqueried value when f_i is sampled from a keyed function family (as low as for a random function of the same range). The *query complexity* of an attack on a variable input length (VIL) function is the number of queries to the underlying primitive necessary to compute the answers to the adversary's queries.

1. We prove that $\phi[f_1]$ is collision resistant up to $O(2^n/n)$ queries to f_1 as long as f_1 is unforgeable. This result also implies the collision resistance of $\phi[f_1]$ against an information-theoretic adversary if f_1 is a random function. It also implies $H[f_1, f_2]$ is unforgeable up to $O(2^n/n)$ query complexity as long as f_1 and f_2 are unforgeable, and that $H[f_1, f_2]$ is weakly collision resistant up to $O(2^n/n)$ query complexity as long as f_1 is unforgeable and f_2 is weakly collision resistant.
2. We prove that $H[f_1, f_2]$ is pseudorandom up to $O(2^n/n)$ query complexity as long as f_1 and f_2 are pseudorandom. In the complexity-theoretic model, we can weaken the assumption so that f_1 is unforgeable.
3. We prove that $\phi[f_1]$ is preimage aware[1] up to $O(2^{2n/3})$ query complexity as long as f_1 is a public random function. By the results of [6], this implies $H[f_1, f_2]$ is indifferentiable from a random oracle up to $O(2^{2n/3})$ query complexity as long as f_1 and f_2 are public random functions.

[1] *Preimage awareness* is a security notion introduced by Dodis, Ristenpart and Shrimpton [6]. The Merkle-Damgård iteration of a preimage aware compression function composed with a random function results in a construction that is indifferentiable from a random oracle, up to the preimage awareness security of the compression function and the maximum message length queried to the iterated construction.



(a) The quadratic blockcipher-based function. (b) The quadratic cascade function.

Fig. 2. Variants of the quadratic compression function.

Refinements. As mentioned, we also investigate two variants of the $3n$-to-$2n$ bit polynomial-based compression function (a.k.a. the "quadratic" compression function) with a view towards concrete implementations of the mode. These alternate constructions are shown in Figure 2. The first variant replaces f_1 by a blockcipher E of $2n$-bit key in Davies-Meyer mode. We show this compression function $\psi[E]$ is collision resistant up to $O(2^n/n)$ queries as long as E is "unpredictable", a notion we discuss in Section 5. Similar corollaries follow on the security of the hash function obtained by iterating $\psi[E]$.

The second is obtained by replacing f_1 with the cascade of two $2n$-bit to n-bit compression functions h_1 and h_2. We show this compression function, denoted $\tau[h_1, h_2]$, is collision resistant up to $O(2^n/n^3)$ queries as long as h_1 and h_2 are unforgeable. It follows that the hash function $G[h_1, h_2, f_2]$ obtained by iterating $\tau[h_1, h_2]$ (defined like $H[f_1, f_2]$ but substituting $\tau[h_1, h_2]$ for $\phi[f_1]$) has unforgeability security up to $O(2^n/n^3)$ query complexity when h_1, h_2 and f_2 are unforgeable, has collision security up to $O(2^n/n^3)$ query complexity when h_1, h_2 are unforgeable and f_2 is collision resistant, and is indistinguishable from a PRF up to $O(2^n/n^3)$ query complexity when h_1, h_2 are unforgeable and f_2 is pseudorandom.

Related Work. All the compression functions discussed in this paper, including the cascaded and blockcipher variants, were proposed by Stam [22, 23]. In [23] Stam proves the collision resistance of polynomial-based compression functions of degrees two and three in the random function model, and also proves the collision security of the quadratic blockcipher mode in the ideal cipher model. Here our contribution is that we weaken the model by showing collision resistance is already assured when f_1 and E are unforgeable/unpredictable rather than random. (It is this weakening of the model that allows us to prove MAC-preservation results for the resulting hash functions.) Regarding the quadratic cascade compression function, Stam proves collision resistance for a special class of non-adaptive adversaries assuming random primitives. Our analysis supports fully adaptive adversaries and once again weakens the model to unforgeable primitives.

Lucks [16] recently proposed a double-pipe hash function iterating a $3n$-bit to $2n$-bit compression function which, like the quadratic blockcipher-based mode, uses a single call to a blockcipher of $2n$-bit key. However, by contrast to the quadratic blockcipher compression function, Lucks' compression function is neither collision resistant nor preimage resistant. As a consequence, collision and preimage security can only be proved in the iteration (higher security notions like indifferentiability are unaddressed). On the other hand, for $n = 128$ Lucks gives a better explicit collision security bound than we do for the quadratic blockcipher compression function: 2^{122} versus 2^{119} queries, respectively.

This paper can be seen as an extension of Yasuda's work [24] since our main achievement is to double the rate of that construction while maintaining comparable MAC-preservation and PRF-preservation properties. However, from a technical standpoint we owe most to Dodis and Steinberger [7], whose "multicollision-to-forgery" (MTF) balls-in-bins games are used in nearly all of our analyses (the sole exception being the preimage awareness bound for polynomial-based compression functions). Indeed, the main "message" of this paper may well be the versatility and power of MTF games.

2 Preliminaries

$\mathbb{F}_{2^n}$ denotes a finite field of order 2^n. Throughout our work, we will identify $\mathbb{F}_{2^n}$ and $\{0,1\}^n$, assuming a fixed mapping between the two sets. For $u \in \{0,1\}^*$, $|u|$ is the length in bits of u. For two bitstrings u and v, $u||v$ denotes the concatenation of u and v. For a set U, we write $u \xleftarrow{\$} U$ to denote uniform sampling from U and assignment to u.

Let $M \in \{0,1\}^*$ and let $c \geq 1$ be message block length (as c will denote throughout the paper). Then $\mathsf{pad}(M) := M||10^b||\langle l \rangle$ where b is the least integer such that $|M||10^b|$ is a multiple of c and where l is the number of c-bit blocks in $M||10^b$. (This representation is possible as long as $l < 2^c$, but this is not a restriction for most applications.) The main property of $\mathsf{pad}(\cdot)$ is that it gives a suffix-free encoding of messages.

The (strengthened) *Merkle-Damgård transform* produces a VIL-function $MD[F] : \{0,1\}^* \to \{0,1\}^n$ from a FIL-function $F : \{0,1\}^{n+c} \to \{0,1\}^n$. Given a predetermined constant $IV \in \{0,1\}^n$, the function $MD[F]$ is defined as follows.

Function $MD[F](M)$

$z[0] \leftarrow IV$
Break $\mathsf{pad}(M)$ into c-bit blocks, $\mathsf{pad}(M) = M[1]|| \ldots ||M[l+1]$
for $i \leftarrow 1$ to $l+1$ **do**
  $z[i] \leftarrow F(z[i-1]||M[i])$
return $z[l+1]$

3 Security Definitions

Unforgeability and Weak Collision Resistance. A *function family* is a map $f : \{0,1\}^\kappa \times \mathsf{Dom}(f) \to \{0,1\}^n$ where $\mathsf{Dom}(f) \subset \{0,1\}^*$. The bitstrings in $\{0,1\}^\kappa$ are the *keys* of f and we write $f_k(M)$ for $f(k,M)$ for $k \in \{0,1\}^\kappa$ and $M \in \mathsf{Dom}(f)$. The function f_k is called a *member* of f. The unforgeability of f as a secure message authentication code (MAC) is estimated by the experiment $\mathbf{Exp}^{\mathsf{mac}}_{\mathcal{A}}$ described in Figure 3(a). In the experiment, an adversary $\mathcal{A}$ has oracle access to $f_k(\cdot)$ and tries to produce a valid tag z for a new message M. Here we call a message M "new" if it has not been queried to oracle $f_k(\cdot)$. The *forgery advantage* of $\mathcal{A}$ is defined by

$$\mathbf{Adv}^{\mathsf{mac}}_f(\mathcal{A}) = \mathbf{Pr}\left[\mathbf{Exp}^{\mathsf{mac}}_{\mathcal{A}} = 1\right]. \tag{1}$$

The probability is taken over the random choice of k and $\mathcal{A}$'s coins (if any). We define $\mathbf{Adv}^{\mathsf{mac}}_f(t,q,\mu)$ as the maximum of $\mathbf{Adv}^{\mathsf{mac}}_f(\mathcal{A})$ over all adversaries $\mathcal{A}$ making at most q queries whose total combined length is at most μ bits (including the forgery produced by $\mathcal{A}$) and of "running time" at most t. The "running time" is defined to be the total running time of the experiment, including the time required to compute the answers to $\mathcal{A}$'s queries. We write $\mathbf{Adv}^{\mathsf{mac}}_f(t,q)$ for $\mathbf{Adv}^{\mathsf{mac}}_f(t,q,\mu)$ if f is a family of fixed input length functions, as in this case μ is automatically determined by q.

The weak collision resistance (WCR) of f is estimated by the experiment $\mathbf{Exp}^{\mathsf{wcr}}_{\mathcal{A}}$ described in Figure 3(b). In contrast to the definition of collision resistance (in the dedicated-key setting) where $\mathcal{A}$ is provided key k, $\mathcal{A}$ is allowed only oracle access to $f_k(\cdot)$. Let

$$\mathbf{Adv}^{\mathsf{wcr}}_f(\mathcal{A}) = \mathbf{Pr}\left[\mathbf{Exp}^{\mathsf{wcr}}_{\mathcal{A}} = 1\right]. \tag{2}$$

Then the weak collision resistance of f, denoted $\mathbf{Adv}^{\mathsf{wcr}}_f(t,q,\mu)$, is defined to be the maximum of $\mathbf{Adv}^{\mathsf{wcr}}_f(\mathcal{A})$ over all adversaries $\mathcal{A}$ making at most q queries whose total combined length is at most μ bits and of running time at most t. When f is a family of fixed input length functions we likewise write $\mathbf{Adv}^{\mathsf{wcr}}_f(t,q)$ instead of $\mathbf{Adv}^{\mathsf{wcr}}_f(t,q,\mu)$.

Our security proof for unforgeability will follow the approach developed by An and Bellare [1]. One of their results is that $f_2 \circ MD[f_1]$ is a VIL-MAC if f_1 is a FIL-WCR and f_2 is a FIL-MAC. With a slight modification, we summarize Lemma 4.2 and Lemma 4.3 in [1] as the following lemma.

Lemma 1. *Let* $f^1 : \{0,1\}^\kappa \times \{0,1\}^{n+c} \to \{0,1\}^n$ *and* $f^2 : \{0,1\}^{\kappa'} \times \{0,1\}^n \to \{0,1\}^m$ *be function families. Then,*

$$\mathbf{Adv}^{\mathsf{mac}}_{f^2 \circ MD[f^1]}(t, \tilde{q}, \mu) \leq \mathbf{Adv}^{\mathsf{mac}}_{f^2}(t, \tilde{q}) + \mathbf{Adv}^{\mathsf{wcr}}_{f^1}\left(t, \left\lfloor \frac{\mu}{c} \right\rfloor + 2\tilde{q}\right).$$

Remark 1. $\left\lfloor \frac{\mu}{c} \right\rfloor + 2\tilde{q}$ is the maximum number of queries to f_1 required to compute $MD[f_1](x_i)$ for $x_1, \ldots, x_{\tilde{q}}$ such that $|x_1| + \ldots + |x_{\tilde{q}}| \leq \mu$.

Experiment $\mathbf{Exp}^{\mathsf{mac}}_{\mathcal{A}}$
$k \xleftarrow{\$} \{0,1\}^\kappa$
$(M, z) \leftarrow \mathcal{A}^{f_k(\cdot)}$
if M is new and $f_k(M) = z$ **then**
output 1
else
output 0

(a) Quantification of unforgeability

Experiment $\mathbf{Exp}^{\mathsf{wcr}}_{\mathcal{A}}$
$k \xleftarrow{\$} \{0,1\}^\kappa$
$(M, M') \leftarrow \mathcal{A}^{f_k(\cdot)}$
if $f_k(M) = f_k(M')$ **then**
output 1
else
output 0

(b) Quantification of weak collision resistance

Fig. 3. Experiments for quantification of unforgeability and weak collision resistance

Indifferentiability and Indistinguishability. In the indifferentiability framework, a distinguisher is given two systems $(F[\mathcal{P}], \mathcal{P})$ and $(\mathcal{H}, \mathcal{S}[\mathcal{H}])$. Here $\mathcal{P}$ is an ideal primitive used as a building block for the construction of $F[\mathcal{P}]$. An ideal primitive $\mathcal{H}$ and a probabilistic Turing machine $\mathcal{S}[\mathcal{H}]$ with oracle access to $\mathcal{H}$ have the same interfaces as $F[\mathcal{P}]$ and $\mathcal{P}$, respectively. The *simulator* $\mathcal{S}[\mathcal{H}]$ tries to emulate the ideal primitive $\mathcal{P}$ so that no distinguisher can tell apart the two systems $(\mathcal{H}, \mathcal{S}[\mathcal{H}])$ and $(F[\mathcal{P}], \mathcal{P})$ with non-negligible probability, based on their responses to queries that the distinguisher may send. We say that the construction $F[\mathcal{P}]$ is indifferentiable from $\mathcal{H}$ if the existence of such a simulator is proved. The indifferentiability implies the absence of a generic attack against $F[\mathcal{P}]$ that regards $\mathcal{P}$ merely as a black-box. Now we give a formal definition of indifferentiability in the information-theoretic model. For a more comprehensive introduction of the indifferentiability framework, we refer to [3, 18].

Definition 1. *A Turing machine F with oracle access to an ideal primitive $\mathcal{P}$ is said to be* (q, ϵ, t)-indifferentiable from an ideal primitive $\mathcal{H}$ *if there exists a simulator $\mathcal{S}$ of running time at most t with oracle access to $\mathcal{H}$ such that for any distinguisher $\mathcal{A}$ making at most q queries, it holds that*

$$\left|\mathbf{Pr}\left[\mathcal{A}^{F[\mathcal{P}],\mathcal{P}} = 1\right] - \mathbf{Pr}\left[\mathcal{A}^{\mathcal{H},\mathcal{S}[\mathcal{H}]} = 1\right]\right| < \epsilon.$$

If $\mathcal{H}$ is a public random function, then $F[\mathcal{P}]$ is called a (q, ϵ, t)-pseudorandom oracle (PRO).

If $\mathcal{A}$ is not allowed to make queries for the underlying primitive, we obtain the definition of indistinguishability.

Definition 2. *A Turing machine F with oracle access to an ideal primitive $\mathcal{P}$ is said to be* (q, ϵ)-indistinguishable from an ideal primitive $\mathcal{H}$ *if for any distinguisher $\mathcal{A}$ making at most q queries, it holds that*

$$\left|\mathbf{Pr}\left[\mathcal{A}^{F[\mathcal{P}]} = 1\right] - \mathbf{Pr}\left[\mathcal{A}^{\mathcal{H}} = 1\right]\right| < \epsilon.$$

If $\mathcal{H}$ is a public random function, then $F[\mathcal{P}]$ is called a (q, ϵ)-pseudorandom function (PRF).

Collision Resistance and Adaptive Preimage Resistance. First, we review the definition of collision resistance *in the information-theoretic model*. Given a function $F = F[\mathcal{P}]$ and an information-theoretic adversary $\mathcal{A}$ both with oracle access to an ideal primitive $\mathcal{P}$, the collision resistance of F against $\mathcal{A}$ is estimated by the experiment $\mathbf{Exp}^{\mathsf{coll}}_{\mathcal{A}}$ described in Figure 4(a). The experiment records every query-response pair that $\mathcal{A}$ obtains by oracle queries into a *query history* $\mathcal{Q}$. We write $z = F_{\mathcal{Q}}(u)$ if $\mathcal{Q}$ contains all the query-response pairs required to compute $z = F(u)$. At the end of the experiment, $\mathcal{A}$ would like to find two distinct evaluations yielding a collision. The *collision-finding advantage* of $\mathcal{A}$ is defined to be

$$\mathbf{Adv}^{\mathsf{coll}}_{F}(\mathcal{A}) = \mathbf{Pr}\left[\mathbf{Exp}^{\mathsf{coll}}_{\mathcal{A}} = 1\right]. \tag{3}$$

The probability is taken over the random choice of $\mathcal{P}$ and $\mathcal{A}$'s coins (if any). For $q > 0$, we define $\mathbf{Adv}^{\mathsf{coll}}_{F}(q)$ as the maximum of $\mathbf{Adv}^{\mathsf{coll}}_{F}(\mathcal{A})$ over all adversaries $\mathcal{A}$ making at most q queries.

In this section, we also present a new notion of security, called *adaptive preimage resistance*. This notion will be useful for the proof of preimage awareness. Given a function $F = F[\mathcal{P}]$ and an information-theoretic adversary $\mathcal{A}$ both with oracle access to an ideal primitive $\mathcal{P}$, the adaptive preimage resistance of F against $\mathcal{A}$ is estimated by the experiment $\mathbf{Exp}^{\mathsf{adpr}}_{\mathcal{A}}$ described in Figure 4(b). At any point during the experiment, the adversary $\mathcal{A}$ is allowed to choose a commitment element z in the range of F such that the query history $\mathcal{Q}$ has not determined any preimage of z. The experiment $\mathbf{Exp}^{\mathsf{adpr}}_{\mathcal{A}}$ records the element z into a *commitment list* $\mathcal{L}$. Queries and commitments are made in an arbitrarily interleaved order. At the end of the experiment, $\mathcal{A}$ would like to succeed in finding a preimage of some element in the commitment list. The *adaptive preimage-finding advantage* of $\mathcal{A}$ is defined to be

$$\mathbf{Adv}^{\mathsf{adpr}}_{F}(\mathcal{A}) = \mathbf{Pr}\left[\mathbf{Exp}^{\mathsf{adpr}}_{\mathcal{A}} = 1\right]. \tag{4}$$

For $q_p, q_e > 0$, we define $\mathbf{Adv}^{\mathsf{adpr}}_{F}(q_p, q_e)$ as the maximum of $\mathbf{Adv}^{\mathsf{adpr}}_{F}(\mathcal{A})$ over all adversaries $\mathcal{A}$ that make at most q_p queries and at most q_e commitments.

Experiment $\mathbf{Exp}^{\mathsf{coll}}_{\mathcal{A}}$
$\mathcal{A}^{\mathcal{P}}$ updates $\mathcal{Q}$
if $\exists\, u \neq u'$, z s.t. $z = F_{\mathcal{Q}}(u) = F_{\mathcal{Q}}(u')$ **then**
  output 1
else
  output 0

(a) Quantification of collision resistance

Experiment $\mathbf{Exp}^{\mathsf{adpr}}_{\mathcal{A}}$
$\mathcal{A}^{\mathcal{P}}$ updates $\mathcal{Q}$ and $\mathcal{L}$
if $\exists\, u$ s.t. $z = F_{\mathcal{Q}}(u)$ for some $z \in \mathcal{L}$ **then**
  output 1
else
  output 0

(b) Quantification of adaptive preimage resistance

Fig. 4. Experiments for quantification of collision resistance and adaptive preimage resistance

Preimage Awareness. The notion of preimage awareness was first introduced by Dodis, Ristenpart and Shrimpton [6]. This notion is useful for the proof of indifferentiability of "NMAC" type constructions. Let $F = F[\mathcal{P}]$ be a function with oracle access to an ideal primitive $\mathcal{P}$. In order to estimate the preimage awareness of F, we use the experiment described in Figure 5. Here an adversary $\mathcal{A}$ is provided two oracles P and Ex. The oracle P provides access to the ideal primitive $\mathcal{P}$ and records a query history $\mathcal{Q}$. Note that this oracle is implicitly used in the experiments for collision resistance and adaptive preimage resistance. The *extraction oracle* Ex provides an interface to an *extractor* $\mathcal{E}$, which is a deterministic algorithm that takes as input an element z in the range of F and the query history $\mathcal{Q}$, and returns either $\perp$ or an element in the domain of F. With respect to the extractor $\mathcal{E}$, we define the advantage of $\mathcal{A}$ to be

$$\mathbf{Adv}^{\mathsf{pra}}_{F}(\mathcal{A}, \mathcal{E}) = \mathbf{Pr}\left[\mathbf{Exp}^{\mathsf{pra}}_{\mathcal{A},\mathcal{E}} = 1\right]. \tag{5}$$

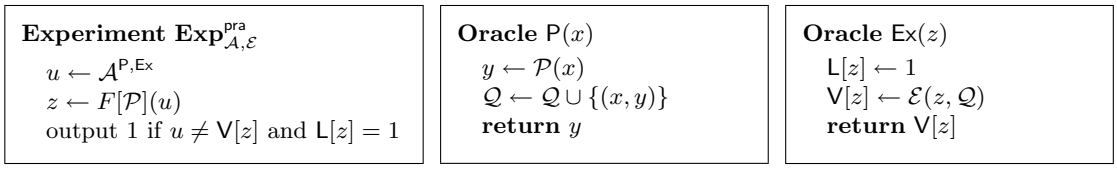


Fig. 5. Experiments for quantification of preimage awareness. Arrays L and V are global, and respectively initialized empty and $\perp$ everywhere

Let $\mathcal{E}^*$ be an algorithm that on input $(z, \mathcal{Q})$ returns an element u if there exists u such that $F_{\mathcal{Q}}(u) = z$ and $\perp$ otherwise. Let

$$\mathbf{Adv}^{\mathsf{pra}}_F(\mathcal{A}, \mathcal{E}^*) = \mathsf{P}_1 + \mathsf{P}_2,$$

where P_1 is the probability that $u \neq \mathsf{V}[z] \neq \perp$ and $\mathsf{L}[z] = 1$ at the end of the experiment and P_2 is the probability that $u \neq \mathsf{V}[z] = \perp$ and $\mathsf{L}[z] = 1$ at the end of the experiment. Then $\mathcal{A}$ can be regarded as a collision-finding adversary such that $\mathbf{Adv}^{\mathsf{coll}}_F(\mathcal{A}) = \mathsf{P}_1$. Furthermore, $\mathcal{A}$ can be transformed into an adaptive preimage-finding adversary $\mathcal{B}$ such that $\mathbf{Adv}^{\mathsf{adpr}}_F(\mathcal{B}) = \mathsf{P}_2$: $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine, asks the same primitive queries as $\mathcal{A}$, and makes commitments z if $\mathcal{A}$ makes a query for $\mathsf{Ex}(z)$ and $\mathcal{Q}$ has not determined any preimage of z. If $\mathcal{A}$ makes at most q_p primitive queries and q_e extraction queries, then it follows that $\mathsf{P}_1 \leq \mathbf{Adv}^{\mathsf{coll}}_F(q_p)$ and $\mathsf{P}_2 \leq \mathbf{Adv}^{\mathsf{adpr}}_F(q_p, q_e)$. We record this observation as the following lemma.

Lemma 2. *Let $F = F[\mathcal{P}]$ be a function with oracle access to an ideal primitive $\mathcal{P}$. Then there exists an extractor $\mathcal{E}^*$ such that for any adversary $\mathcal{A}$ it holds that*

$$\mathbf{Adv}^{\mathsf{pra}}_F(\mathcal{A}, \mathcal{E}^*) \leq \mathbf{Adv}^{\mathsf{coll}}_F(q_p) + \mathbf{Adv}^{\mathsf{adpr}}_F(q_p, q_e).$$

The main application of preimage awareness lies in the construction of pseudorandom oracles. In the following lemma which is a combination of Theorem 4.1 and Theorem 4.2 in [6], unpad is an algorithm such that $\mathsf{unpad}(y) = x$ if $y = \mathsf{pad}(x)$ is a valid output of pad and $\mathsf{unpad}(y) = \perp$ otherwise. For any algorithm $\mathcal{F}$, we write $\mathsf{Time}(\mathcal{F}, l)$ for the maximum time required to compute $\mathcal{F}(x)$ for any input x such that $|x| \leq l$. If $\mathcal{F}$ is an algorithm with oracle access to an ideal primitive $\mathcal{P}$, then $\mathsf{NQ}(F, l)$ is the maximum number of queries to $\mathcal{P}$ required to compute $F(x)$ for any input x such that $|x| \leq l$. Without any constraint on the input length, we just write $\mathsf{Time}(\mathcal{F})$ and $\mathsf{NQ}(F)$.

Lemma 3. *Let $F : \{0,1\}^{n+c} \to \{0,1\}^n$ be a function with oracle access to an ideal primitive $\mathcal{P}$ and let $g : \{0,1\}^n \to \{0,1\}^m$ and $\mathcal{H} : \{0,1\}^* \to \{0,1\}^m$ be public random functions for $m \leq n$. For an arbitrary extractor $\mathcal{E}$ with respect to F, there exists a simulator $\mathcal{S}$ such that for any distinguisher $\mathcal{A}$ making at most (q_0, q_1, q_2) queries to the three oracle interfaces associated with $(\mathcal{H}, \mathcal{P}, g)$, there exists an adversary $\mathcal{B}$ such that*

$$\left|\mathbf{Pr}\left[\mathcal{A}^{g \circ MD[F[\mathcal{P}]],(\mathcal{P},g)} = 1\right] - \mathbf{Pr}\left[\mathcal{A}^{\mathcal{H},\mathcal{S}[\mathcal{H}]} = 1\right]\right| \leq \mathbf{Adv}^{\mathsf{pra}}_F(\mathcal{B}, \mathcal{E}).$$

Let l_{max} be the length in bits of the longest query made by $\mathcal{A}$ to its first oracle, and let $L = \left\lceil \frac{l_{max}+1}{c} + 1 \right\rceil$. Then, simulator $\mathcal{S}$ runs in time $O\left(q_1 + Lq_2\mathsf{Time}(\mathcal{E}) + Lq_2\mathsf{Time}(\mathsf{unpad})\right)$. Adversary $\mathcal{B}$ runs in time $O\left(\mathsf{Time}(\mathcal{A}) + q_0\mathsf{Time}(MD[F], l_{max}) + q_1 + (L+1)q_2\right)$, makes at most $L\mathsf{NQ}(F)(q_0+1) + q_1$ primitive queries, and makes at most Lq_2 extraction queries.

4 Security of the Polynomial-based Mode of Operation

For this section and the rest of the paper, $\phi[f_1]$ and $H[f_1, f_2]$ refer to the compression function and hash function defined in Section 1. We use "log" to denote the logarithm base 2. For simplicity of notation, we assume that $\log q$ is an integer for the number of queries q.

4.1 Weak Collision Resistance and Unforgeability

We begin with the proof of weak collision resistance for $\phi[f_1]$ such that f_1 is randomly chosen from a function family f.

Theorem 1. *Let ϕ be a function family defined by $f : \{0,1\}^\kappa \times \{0,1\}^{2n+c} \to \{0,1\}^n$. Then,*

$$\mathbf{Adv}^{\mathsf{wcr}}_{\phi}(t,q) \leq 2q \max(d + \log q, d2^{d+2}) \mathbf{Adv}^{\mathsf{mac}}_{f}\left(t + O(d^2n^2q^{d+2}) + \mathsf{Time}_d, q\right),$$

where $d = \left\lceil \frac{2n+c}{n} \right\rceil - 1$ and Time_d is the time required to solve a univariate polynomial equation of degree d over $\mathbb{F}_{2^n}$.

Remark 2. For a univariate polynomial of degree d over $\mathbb{F}_{2^n}$, there is a deterministic algorithm to find zeros using $O(d^{3/2}n)$ field operations (ignoring log factors). See [8, 9].

In order to prove Theorem 1, we use a generalization of the multicollision-to-forgery (MTF) balls-in-bins game first introduced in [7].

MTF game. This game is played by two players $\mathcal{A}$ and $\mathcal{B}$ according to the following rules. Parameters are integers $q > 0$ and $m_2 > m_1 \geq 0$.

1. The game consists of q rounds.
2. At each round, $\mathcal{A}$ publicly places a set of balls into a set of bins such that
   (a) balls placed at the same round go into distinct bins,
   (b) the number of balls that are placed into bins already containing m_1 balls at the moment of placement is finite,
   (c) some bin eventually contains more than m_2 balls.
3. Before each round, $\mathcal{B}$ can secretly "pass" or "guess" a bin that will receive a ball in the next round. $\mathcal{B}$ makes exactly one guess throughout the game.
4. If $\mathcal{B}$ makes a correct guess, then $\mathcal{B}$ wins. Otherwise, $\mathcal{B}$ loses.

Note there is no constraint on the total number of balls or bins.

Lemma 4. *Irrespective of $\mathcal{A}$'s strategy, there exists a strategy for $\mathcal{B}$ to win the above game with probability at least $1/q \cdot 1/(c_{m_1})^{1/(m_2-m_1)}$, where c_{m_1} is the number of balls that are placed into bins already containing m_1 balls at the moment of placement.*

Proof. $\mathcal{B}$'s strategy is simple, as follows:

1. Choose a round $i \in \{1, \ldots, q\}$ and a level $j \in \{m_1 + 1, \ldots, m_2\}$ uniformly at random.
2. Before the i-th round of the game, guess a bin uniformly at random from all bins containing at least j balls already.

Let c_j be the total number of balls that are placed into bins that already have at least j balls in them right before the round when the ball is placed. For a given value of j, each ball placed into a bin with at least j balls in it already gives $\mathcal{B}$ chance at least $1/(qc_{j-1})$ of winning since c_{j-1} is an upper bound for the number of bins that have j balls in them at the end of the game. Therefore $\mathcal{B}$'s chance of winning is at least $c_j/(qc_{j-1})$ for a given value of j, and hence $\mathcal{B}$'s overall chance of winning is at least

$$\begin{aligned}
\frac{1}{m_2 - m_1} \sum_{j=m_1+1}^{m_2} \frac{c_j}{qc_{j-1}} &= \frac{1}{q} \mathsf{ArithmeticMean}\left(\frac{c_{m_1+1}}{c_{m_1}}, \ldots, \frac{c_{m_2}}{c_{m_2-1}}\right) \\
&\geq \frac{1}{q} \mathsf{GeometricMean}\left(\frac{c_{m_1+1}}{c_{m_1}}, \ldots, \frac{c_{m_2}}{c_{m_2-1}}\right) \\
&\geq \frac{1}{q} \left(\frac{1}{c_{m_1}}\right)^{1/(m_2-m_1)}.
\end{aligned}$$

□

Note that Lemma 4 asserts nothing about $\mathcal{B}$'s efficiency—in fact, if a huge number of balls are thrown, it may be difficult to keep track of all bins that have received at least j balls already (which is necessary for sampling uniformly among the bins). In our case, bins will often be curves defined by polynomials of degree $\leq d$ over $\mathbb{F}_{2^n}$ and balls points in $\mathbb{F}_{2^n} \times \mathbb{F}_{2^n}$, where a ball (x, y) goes into bin $\mathcal{C}$ if $(x, y) \in \mathcal{C}$ (a ball is thus "cloned" into many different bins). In this setting, it becomes easier to keep track of which bins have at least j balls in them when $j \geq d + 1$, as $d + 1$ points uniquely determine a polynomial of degree $\leq d$. Thus for such a game it may be helpful to set $m_1 = d$, in order to keep the complexity of sampling under control. (Dodis and Steinberger do not have games in which the number of bins containing balls is so large that sampling for small values of j is an issue, and always use MTF games with $m_1 = 0$.) The value of m_2 is then set large enough to make the term $(1/c_{m_1})^{1/(m_2-m_1)}$ small.

We typically upper bound c_{m_1} by qM where M is an upper bound on *the total number of bins with at least $m_1 + 1$ balls at the end of the game*. Indeed, because balls are thrown into distinct bins at each round, this definition of M implies that at each round at most M balls are thrown into bins with m_1 balls in them already. We thus have the following corollary:

Corollary 1. *If the number of bins that contain at least $m_1 + 1$ balls at the end of the MTF game is at most M, then $\mathcal{B}$ can win the MTF game with probability at least $1/q \cdot 1/(qM)^{1/(m_2-m_1)}$.*

We note that Corollary 1 is a bit wasteful, in the sense that it is possible to give a better bound for $\mathcal{B}$'s chance of success as a function of m_1, m_2 and M from the relationship $M = c_{m_1} - c_{m_1+1}$ and the fact that $\mathcal{B}$'s chance of success is also lower bounded by $\frac{\alpha}{q(m_2-m_1)}$ where $\alpha = \max\left(\frac{c_{m_1+1}}{c_{m_1}}, \ldots, \frac{c_{m_2}}{c_{m_2-1}}\right)$. However this gain leads to a more complicated statement and is minor enough for us to ignore [2].

Proof (Theorem 1). Let $\mathcal{A}$ be a weak collision-finding adversary such that

$$\mathbf{Adv}^{\mathsf{wcr}}_{\phi}(\mathcal{A}) = \mathbf{Adv}^{\mathsf{wcr}}_{\phi}(t, q) = \epsilon.$$

We write $u[i] = u_d[i]||\cdots||u_0[i]$ for the i-query that $\mathcal{A}$ makes to $f_1(\cdot)$ and $\phi(u[i]) = (x[i], y[i])$, where $x[i] = f_1(u[i])$ and $y[i] = u_d[i]x[i]^d + \cdots + u_1[i]x[i] + u_0[i]$. The i-th query is associated with a curve

$$\mathcal{C}_i = \{(x, y) \in \mathbb{F}^2_{2^n} : y = u_d[i]x^d + \cdots + u_1[i]x + u_0[i]\}.$$

Let $\Gamma_i = \{1 \leq j \leq i : (x[j], y[j]) \in \mathcal{C}_i\}$ and let $\gamma = \max_i |\Gamma_i|$. By assumption that $\mathcal{A}$ succeeds to find a collision with probability ϵ, one of the following two events occurs with probability at least $\epsilon/2$.

Case 1: $\mathcal{A}$ finds a collision and $\gamma \leq d + \log q$. For this case, we can construct a forger $\mathcal{B}_1$ for f_1 as follows.

1. $\mathcal{B}_1$ chooses $i \in \{1, \ldots, q\}$ and $s \in \{1, \ldots, d + \log q\}$ uniformly at random.
2. $\mathcal{B}_1$ runs $\mathcal{A}$ as a subroutine and faithfully answers the queries made by $\mathcal{A}$ until the $(i - 1)$-th query.
3. On the i-query $u[i]$, $\mathcal{B}_1$ presents a forgery $(u[i], x[j_s])$ without making a query to $f_1(\cdot)$, where j_s is the s-th element of Γ_i. (If $|\Gamma_i| < s$, then $\mathcal{B}_1$ presents a random value.)

Note that if there exists a collision $(x[j], y[j]) = (x[i], y[i])$ for $j < i$, then $(x[j], y[j]) \in \mathcal{C}_i$ or equivalently $j \in \Gamma_i$. With this observation, we obtain

$$\mathbf{Adv}^{\mathsf{mac}}_{f}(\mathcal{B}_1) \geq \frac{\epsilon}{2q(d + \log q)}. \tag{6}$$

[2] More precisely, we have $M = c_{m_1} - c_{m_1+1} \geq c_{m_1}(1-\alpha)$ or $c_{m_1} \leq M/(1-\alpha)$. Then $\mathcal{B}$'s chance of success is at least $\frac{1}{q} \max\left(\frac{\alpha}{m_2-m_1}, \left(\frac{1-\alpha}{M}\right)^{1/(m_2-m_1)}\right)$ where we know $0 < \alpha \leq 1$.

Case 2: $\mathcal{A}$ produces $\gamma > d + \log q$. This is the case where we construct a forger $\mathcal{B}_2$ for f_1 using the MTF game: The bins are $(d+1)$-tuples $(u_d, \ldots, u_0) \in \mathbb{F}_{2^n}^{d+1}$ (regarded as a curve in the plane $\mathbb{F}_{2^n}^2$) and the balls are points $(x, y) \in \mathbb{F}_{2^n}^2$. A query $f_1(u_d||\cdots||u_0)$ results in a new ball $(x, y) = (f_1(u_d||\cdots||u_0), u_d x^d + \cdots + u_1 x + u_0)$ being placed into all bins $(v_d, \ldots, v_0)$ such that $v_d x^d + \cdots + v_1 x + v_0 = y$, namely all bins giving the coefficients of a polynomial curve fitting the point (x, y), *except the bin* $(u_d, \ldots, u_0)$ *itself*. Thus one ball is replicated in $2^{dn} - 1$ different bins. We assume that the i-th query $u[i]$ is known to $\mathcal{B}_2$ before the i-th round of the game. Then, when $\mathcal{B}_2$ correctly guesses a bin $(v_d, \ldots, v_0)$ that will receive the new ball $(x[i], y[i])$, $\mathcal{B}_2$ has a chance to forge f_1 with probability $1/d$ since the intersection of the curves associated with $(u_d, \ldots, u_0)$ and $(v_d, \ldots, v_0)$ contains at most d elements. Here we assume the existence of an algorithm of running time Time_d to find zeros of a univariate polynomial of degree d. Let $m_1 = d$, $m_2 = d + \log q$ and $M = \binom{q}{d+1}$. Since $d + 1$ points determine a unique polynomial of degree d fitting the points, we can apply Corollary 1 to obtain a forger $\mathcal{B}_2$ of success probability

$$\mathbf{Adv}_f^{\mathsf{mac}}(\mathcal{B}_2) \geq \frac{\epsilon}{2} \cdot \frac{1}{d} \cdot \frac{1}{q} \left(\frac{1}{qM}\right)^{1/(m_2 - m_1)} \geq \frac{\epsilon}{2} \cdot \frac{1}{d} \cdot \frac{1}{q} \left(\frac{1}{q^{d+2}}\right)^{1/\log q} = \frac{\epsilon}{dq2^{d+3}}, \tag{7}$$

and of running time $O(d^2 n^2 q^{d+2}) + \mathsf{Time}_d$. From (6) and (7), it follows that

$$\mathbf{Adv}_\phi^{\mathsf{wcr}}(t, q) \leq 2q \max(d + \log q, d2^{d+2}) \mathbf{Adv}_f^{\mathsf{mac}}\left(t + O(d^2 n^2 q^{d+2}) + \mathsf{Time}_d, q\right).$$

□

The following theorem is immediate from Lemma 1 and Theorem 1.

Theorem 2. *Let $H = H[f_1, f_2]$ be a function family where f_1 and f_2 are independently chosen from two function families f^1 and f^2, respectively. Then for $q = \lfloor \mu/c \rfloor + 2\tilde{q}$,*

$$\mathbf{Adv}_H^{\mathsf{mac}}(t, \tilde{q}, \mu) \leq \epsilon,$$

where

$$\epsilon = \mathbf{Adv}_{f^2}^{\mathsf{mac}}(t, \tilde{q}) + 2q \max\left(d + \log q, d2^{d+2}\right) \mathbf{Adv}_{f^1}^{\mathsf{mac}}\left(t + O(d^2 n^2 q^{d+2}) + \mathsf{Time}_d, q\right),$$

and Time_d is the time required to solve a univariate polynomial equation of degree d over $\mathbb{F}_{2^n}$. If $f_1 = f_2$ are chosen from the same function family f, then

$$\mathbf{Adv}_{H^*}^{\mathsf{mac}}(t, \tilde{q}, \mu) \leq \epsilon + q2^{\max\{0, n-c\}} \mathbf{Adv}_f^{\mathsf{mac}}(t, q),$$

where ϵ is as above with f replacing f^1 and f^2.

In the single-key setting, we assume that $IV_1 \neq 0^n$ for two n-bit blocks IV_1 and IV_2 such that $IV = IV_1||IV_2$. Then we can use the techniques employed in the CS construction [19]. The term $q2^{\max\{0,n-c\}}\mathbf{Adv}_f^{\mathsf{mac}}(t, q)$ comes from the case where $f_1(M[i]) = 0^{\min\{n,c\}}||*$ for some message block $M[i]$ during the Merkle-Damgård iteration.

4.2 Collision Resistance and Indistinguishability

Let Φ_{2n+c}^n be the set of all functions from $\{0,1\}^{2n+c}$ to $\{0,1\}^n$. Then Φ_{2n+c}^n can be regarded as a function family $f^* : \{0,1\}^\kappa \times \{0,1\}^{2n+c} \to \{0,1\}^n$ by identifying Φ_{2n+c}^n and $\{0,1\}^\kappa$ for $\kappa = n2^{2n+c}$. The weak collision resistance of ϕ defined by f^* against a computationally unbounded adversary implies its collision resistance in the information-theoretic model (due to the equivalence of oracle access to either ϕ or f^*). Since

$$\mathbf{Adv}_{f^*}^{\mathsf{mac}}(t, q) = \frac{1}{2^n}$$

for any q and t, the following theorem is immediate from Theorem 1.

Theorem 3. *If $f_1 : \{0,1\}^{2n+c} \to \{0,1\}^n$ is a random function, then*

$$\mathbf{Adv}^{\mathsf{coll}}_{\phi[f_1]}(q) \leq \frac{\max\left(d + \log q, d2^{d+2}\right) q}{2^{n-1}}.$$

When we construct NMAC type pseudorandom oracles based on preimage aware functions, adaptive preimage resistance is only needed for the case where a distinguisher makes a query to the finalization function. If there is no interface to access the inner primitive, we do not need to worry about adaptive preimage finding. The following lemma shows that any collision resistant function can be combined with a random function producing a pseudorandom function.

Lemma 5. *Let $F : \{0,1\}^* \to \{0,1\}^n$ be a function with oracle access to an ideal primitive $\mathcal{P}$ and let $g : \{0,1\}^n \to \{0,1\}^m$ and $H : \{0,1\}^* \to \{0,1\}^m$ be random functions. Then the composite function $g \circ F$ is $(\tilde{q}, \epsilon)$-indistinguishable from H, where $\epsilon = \mathbf{Adv}^{\mathsf{coll}}_F(q)$, $q = \mathsf{NQ}(F, l_{max})\tilde{q}$ and l_{max} is the length in bits of the longest query made by a distinguisher.*

Proof. Let $\mathcal{G}_0$ and $\mathcal{G}_1$ be games with a single interface, as defined in Figure 6. Assume that a distinguisher $\mathcal{A}$ makes no redundant query. Then whenever $\mathcal{A}$ makes a query to $H(\cdot)$ in game $\mathcal{G}_0$, it will receive an independent random value in $\{0,1\}^m$. It means that the interface $H(\cdot)$ faithfully implements a random function $H : \{0,1\}^* \to \{0,1\}^m$ in game $\mathcal{G}_0$. On the other hand, the interface $H(\cdot)$ in game $\mathcal{G}_1$ faithfully implements $g \circ F[\mathcal{P}] : \{0,1\}^* \to \{0,1\}^m$ for a random function $g : \{0,1\}^n \to \{0,1\}^m$ since Sample-$g(v,w)$ only depends on the value v. Flag bad sets to true only when $\mathcal{A}$ makes a collision in $F[\mathcal{P}]$. Therefore, for any distinguisher $\mathcal{A}$ that makes at most $\tilde{q}$ queries to $H(\cdot)$, we have

$$\left|\mathbf{Pr}\left[\mathcal{A}^{g\circ F[\mathcal{P}]} = 1\right] - \mathbf{Pr}\left[\mathcal{A}^{H} = 1\right]\right| \leq \mathbf{Pr}\left(\mathcal{A}^{\mathcal{G}_1} \text{ sets } \mathsf{bad}\right) \leq \mathbf{Adv}^{\mathsf{coll}}_F(q),$$

where $q = \mathsf{NQ}(F, l_{max})\tilde{q}$ and l_{max} is the length in bits of the longest query made by a distinguisher. □

Games $\mathcal{G}_0$ | **Games $\mathcal{G}_1$**

Interface $H(v)$
  $w \leftarrow F[\text{Sample-}\mathcal{P}](v)$
  $z \leftarrow \text{Sample-}g(v,w)$
  return z

Subroutine Sample-$\mathcal{P}(x)$
  return $\mathcal{P}(x)$

Subroutine Sample-$g(v,w)$
  $g(v,w) \xleftarrow{\$} \{0,1\}^m$
  if $g(v',w)$ was previously queried for $v' \neq v$ **then**
    bad $\leftarrow$ **true**
    $g(v,w) \leftarrow g(v',w)$
  return $g(v,w)$

Fig. 6. Games $\mathcal{G}_0$ and $\mathcal{G}_1$. $\mathcal{G}_1$ includes the boxed statement

Since the strengthened Merkle-Damgård transform preserves collision resistance, we obtain the following theorem.

Theorem 4. *If $f_1, f_2 : \{0,1\}^{2n+c} \to \{0,1\}^n$ are random functions, then $H[f_1, f_2]$ is $(\tilde{q}, \epsilon)$-indistinguishable from a random function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$, where*

$$\epsilon = \frac{\max\left(d + \log q, d2^{d+2}\right) q}{2^{n-1}},$$

and

$$q = \mathsf{NQ}(MD[\phi[f_1]], l_{max})\tilde{q} = \left\lceil \frac{l_{max}+1}{c} + 1 \right\rceil \tilde{q},$$

for the length in bits l_{max} of the longest query made by a distinguisher. In the single-key setting, $H[f_1, f_1]$ is $(\tilde{q}, \epsilon + \frac{q}{2^c})$-indistinguishable from $\mathcal{H}$.

Lemma 5 holds with $\epsilon = \mathbf{Adv}^{\mathsf{wcr}}_F(t, \tilde{q}, l_{max}\tilde{q})$ when F is a keyed function family (in the complexity-theoretic model). This implies that $H[f_1, f_2]$ is pseudorandom up to $O(2^n/n)$ query complexity as long as f_1 is unforgeable and f_2 is pseudorandom.

4.3 Preimage Awareness and Indifferentiability

We begin with the proof of adaptive preimage resistance for $\phi[f_1]$ where f_1 is a public random function. Let $\mathcal{A}$ be an "optimal" adaptive preimage-finding adversary that makes at most q_p queries and at most q_e commitments. That is, $\mathbf{Adv}^{\mathsf{adpr}}_{\phi[f_1]}(\mathcal{A}) = \mathbf{Adv}^{\mathsf{adpr}}_{\phi[f_1]}(q_p, q_e)$. During the experiment $\mathbf{Exp}^{\mathsf{adpr}}_{\mathcal{A}}$, $\mathcal{A}$ makes queries and commitments in an arbitrarily interleaved order based on a deterministic strategy. Here we can assume that the strategy does not depend on the responses of oracle $f_1(\cdot)$ to queries that $\mathcal{A}$ sends since the probability distribution of the response to a certain query is independent of the previous query-response pairs (as long as $\mathcal{A}$ does not make a redundant query). Therefore, in order to estimate $\mathbf{Adv}^{\mathsf{adpr}}_{\phi[f_1]}(\mathcal{A})$, we can use the following game.

1. $\mathcal{A}$ makes q_p queries $\{\mathcal{C}_1, \ldots, \mathcal{C}_{q_p}\}$ and q_e commitments $\mathcal{L} = \{(x_1, y_1), \ldots, (x_{q_e}, y_{q_e})\}$ based on the optimal strategy. (Here each query is represented by a curve in $\mathbb{F}^2_{2^n}$ as in the analysis of unforgeability.)
2. One point (x^*_i, y^*_i) is chosen from each curve $\mathcal{C}_i$ uniformly at random.
3. If $(x^*_i, y^*_i) \in \mathcal{L}$ for some $i = 1, \ldots, q_e$, then $\mathcal{A}$ wins. Otherwise, $\mathcal{A}$ loses.

The winning probability of $\mathcal{A}$ for the above game is equal to the adaptive preimage-finding advantage of $\mathcal{A}$. Let $\mathit{\Gamma}_i = \mathcal{C}_i \cap \mathcal{L}$ for $i = 1, \ldots, q_p$, and let

$$\mathit{\Delta}_\theta = \{1 \le i \le q_p : |\mathit{\Gamma}_i| \ge \theta\},$$

for a parameter θ. Then for $\mathit{\Delta} \subset \mathit{\Delta}_\theta$ and $\delta = |\mathit{\Delta}|$, we have

$$q_e \ge \left| \bigcup_{i \in \mathit{\Delta}} \mathit{\Gamma}_i \right| \ge \sum_{i \in \mathit{\Delta}} |\mathit{\Gamma}_i| - \sum_{i \ne j \in \mathit{\Delta}} |\mathit{\Gamma}_i \cap \mathit{\Gamma}_j| \ge \delta\theta - \binom{\delta}{2} \cdot d.$$

Therefore we conclude that $|\mathit{\Delta}_{\theta^*}| < \delta^*$ for any (θ^*, δ^*) such that

$$\delta^*\theta^* - \binom{\delta^*}{2} \cdot d > q_e. \tag{8}$$

This implies that the number of curves that intersect with $\mathcal{L}$ at $\ge \theta^*$ points is less than δ^*. Thus the winning probability of $\mathcal{A}$ is upper-bounded by

$$\mathbf{Adv}^{\mathsf{adpr}}_{\phi[f_1]}(\mathcal{A}) \le \delta^* \frac{q_e}{2^n} + (q_p - \delta^*)\frac{\theta^*}{2^n}.$$

By Lemma 2 and Theorem 3, we obtain the following theorem.

Theorem 5. *Let (θ^*, δ^*) satisfy inequality (8). Then for a random function f_1, there exists an extractor $\mathcal{E}^*$ such that for any adversary $\mathcal{A}$ it holds that*

$$\mathbf{Adv}^{\mathsf{pra}}_{\phi[f_1]}(\mathcal{A}, \mathcal{E}^*) \le \frac{\max\left(d + \log q_p, d2^{d+2}\right) q_p}{2^{n-1}} + \delta^* \frac{q_e}{2^n} + (q_p - \delta^*)\frac{\theta^*}{2^n}.$$

We can use Lemma 3 and Theorem 5 with $(\theta^*, \delta^*) = (dq_e^{1/2}, q_e^{1/2})$ to obtain the following theorem.

Theorem 6. *Let $f_1, f_2 : \{0,1\}^{2n+c} \to \{0,1\}^n$ and $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$ be public random functions. Then there exists a simulator $\mathcal{S}$ such that for any distinguisher $\mathcal{A}$ making at most (q_0, q_1, q_2) queries to the three oracle interfaces associated with $(\mathcal{H}, f_1, f_2)$,*

$$\left|\mathbf{Pr}\left[\mathcal{A}^{H[f_1,f_2],(f_1,f_2)} = 1\right] - \mathbf{Pr}\left[\mathcal{A}^{\mathcal{H},\mathcal{S}[\mathcal{H}]} = 1\right]\right| \leq \epsilon,$$

where

$$\epsilon = \frac{\max\left(d + \log\left(Lq_0 + q_1 + L\right), d2^{d+2}\right)}{2^{n-1}}\left(Lq_0 + q_1 + L\right) + \frac{L^{1/2}q_2^{1/2}}{2^n}\left(dLq_0 + dq_1 + Lq_2 + dL\right),$$

l_{max} is the length in bits of the longest query made by $\mathcal{A}$ to its first oracle and $L = \left\lceil \frac{l_{max}+1}{c} + 1 \right\rceil$. Simulator $\mathcal{S}$ runs in time $O\left(q_1 + Lq_2\mathsf{Time}(\mathcal{E}^*) + Lq_2\mathsf{Time}(\mathsf{unpad})\right)$, where $\mathcal{E}^*$ is the obvious extractor used in Lemma 2. In the single-key setting, we have

$$\left|\mathbf{Pr}\left[\mathcal{A}^{H[f_1,f_1],f_1} = 1\right] - \mathbf{Pr}\left[\mathcal{A}^{\mathcal{H},\mathcal{S}'[\mathcal{H}]} = 1\right]\right| \leq \epsilon + \frac{Lq_0}{2^c},$$

where simulator $\mathcal{S}'$ is obtained by a slight modification of $\mathcal{S}$: On input x, $\mathcal{S}'$ returns $f_2(x)$ if $x = 0^c||*$ for some $* \in \{0,1\}^{2n}$ and returns $f_1(x)$ otherwise, by using the interfaces f_1 and f_2 of $\mathcal{S}$.

Tightness of Indifferentiability. The preservation of indifferentiability is guaranteed only up to $O(2^{2n/3})$ query complexity which is beyond the birthday bound but still far from optimal. This bound is dominated by the adaptive preimage resistance, depending on a configuration that consists of q curves in $\mathbb{F}_{2^n}^2$ and q points on the curves (assuming $q = q_p = q_e$). If there exists a subfield $\mathbb{F}'$ of $\mathbb{F}_{2^n}$ such that $|\mathbb{F}'| = \sqrt{q}$, then we have a configuration that provides tight adaptive preimage resistance: The set of points is $\mathbb{F}' \times \mathbb{F}' \subset \mathbb{F}_{2^n}^2$ and the set of curves consists of q polynomials of degree d with coefficients in $\mathbb{F}'$. However, for the case where $\mathbb{F}_{2^n}$ admits no proper subfield (e.g. with prime n), there remains a question whether a similar construction exists or the bound can be qualitatively improved. We pose this as an open problem.

5 The Quadratic Blockcipher-based Compression Function

In this section and the next, we discuss how to instantiate $\phi[f_1]$ for $c = n$ (the "quadratic" polynomial mode) by replacing the $3n$-bit to n-bit compression function f_1 with a smaller primitive. First, we discuss a concrete instantiation of the quadratic compression function using a blockcipher with $2n$-bit keys. Given $f_1 : \{0,1\}^{3n} \to \{0,1\}^n$, the compression function $\phi[f_1] : \{0,1\}^{3n} \to \{0,1\}^{2n}$ is defined by $\phi[f_1](u_2||u_1||u_0) = x||y$, where $x = f_1(u_2||u_1||u_0)$ and $y = u_2x^2 + u_1x + u_0 \in \mathbb{F}_{2^n}$ for u_2, u_1, $u_0 \in \{0,1\}^n$. In the *quadratic blockcipher-based compression function*, f_1 is implemented using a blockcipher $E : \{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^n$, $E(k,x) = E_k(x)$ by letting $f_1(u_2||u_1||u_0) = E_{u_2||u_1}(u_0) + u_0$ as described in Figure 2(a). We write $\psi[E]$ for the resulting compression function. Thus $\psi[E] : \{0,1\}^{3n} \to \{0,1\}^n$ and $\psi[E](u_2||u_1||u_0) = x||y$ where

$$x = E_{u_2||u_1}(u_0) + u_0,$$
$$y = u_2x^2 + u_1x + u_0.$$

We can prove that $\psi[E]$ provide similar security as the quadratic mode $\phi[f_1]$ when instantiated with an ideal cipher E, in terms of unforgeability, collision resistance and pseudorandomness. In fact, our results do not actually necessitate an ideal cipher E (which is a set of independent random permutations with one permutation per key) but only an "unpredictable" blockcipher E. For the latter, all that is assumed is that it is difficult for an adversary to fully predict the output of an unqueried value. We call this the *unpredictability* of the blockcipher (which is similar to

the unforgeability of a keyed function family, except no keys are involved) and we quantify it by the advantage $\mathbf{Adv}^{\mathsf{unp}}_{E}(t,q)$ which is the maximum over all adversaries A running in time t and making at most q queries to E of the probability that A can output a tuple $(u_2||u_1, v, w)$ such that $E_{u_2||u_1}(v) = w$ without making queries for $E_{u_2||u_1}(v)$ or $E^{-1}_{u_2||u_1}(w)$.

Implicitly $\mathbf{Adv}^{\mathsf{unp}}_{E}(t,q)$ depends on a sampling procedure for E. In the ideal cipher model, E is sampled uniformly at random among all n-bit blockciphers with $2n$-bit keys. Here we allow any sampling procedure for E. Note that $\mathbf{Adv}^{\mathsf{unp}}_{E}(t,q) \leq 1/(2^n - q)$ if E is an ideal cipher, so we can always revert to that bound by assuming an ideal cipher. Our use of unpredictable blockciphers is somewhat similar to that of [7], with the significant difference that the blockciphers of [7] use fixed keys, and that they are sampled by sampling the fixed keys. The unpredictability then corresponds to the unforgeability of a keyed function family (which happens to be a family of permutations).

We show that the collision resistance of $\psi[E]$ can be effectively bounded in terms of $\mathbf{Adv}^{\mathsf{unp}}_{E}(t,q)$. Let $\mathbf{Adv}^{\mathsf{coll}}_{\psi[E]}(t,q)$ be the maximum probability that an adversary A of running time t with oracle access to E and E^{-1} outputs a collision (M, M') for $\psi[E]$ which it has verified (i.e. has made the queries necessary to compute $\psi[E](M)$ and $\psi[E](M')$).

Theorem 7. *Let $\psi[E]$ be the quadratic blockcipher-based compression function, where E is sampled from the set of all n-bit blockciphers with $2n$-bit keys according to an arbitrary fixed distribution. Then,*

$$\mathbf{Adv}^{\mathsf{coll}}_{\psi[E]}(t,q) \leq 2q(\log q + 3)\mathbf{Adv}^{\mathsf{unp}}_{E}(t + O(n^2q^4) + \mathsf{Time}_2, q).$$

Furthermore, let $G = G[E, f_2] = f_2 \circ MD\,[\psi[E]]$ be a function family where f_2 is chosen from a function family $f : \{0,1\}^{\kappa} \times \{0,1\}^{2n} \to \{0,1\}^n$. Then for $q = \lfloor \mu/c \rfloor + 2\tilde{q}$,

$$\mathbf{Adv}^{\mathsf{mac}}_{G}(t,\tilde{q},\mu) \leq \mathbf{Adv}^{\mathsf{mac}}_{f}(t,\tilde{q}) + 2q(\log q + 3)\mathbf{Adv}^{\mathsf{unp}}_{E}(t + O(n^2q^4) + \mathsf{Time}_2, q).$$

Remark 3. The term "Time_2", which represents the time necessary to select a root of a quadratic polynomial over $\mathbb{F}_{2^n}$, is mainly kept to facilitate comparison with Theorem 1.

Proof. Let $\mathcal{A}$ be a collision-finding adversary for $\psi[E]$ of running time t, that makes q queries and that achieves advantage $\epsilon = \mathbf{Adv}^{\mathsf{coll}}_{\psi[E]}(t,q)$. We make the standard assumption that $\mathcal{A}$ never asks E a query to which it knows the answer. We say $\mathcal{A}$ "completes a query $E_{u_2||u_1}(u_0) = c$" to mean either that $\mathcal{A}$ made a forward query $E_{u_2||u_1}(u_0)$ resulting in the answer c or that $\mathcal{A}$ made an inverse query $E^{-1}_{u_2||u_1}(c)$ resulting in the answer u_0. If $\mathcal{A}$ completes two queries $E_{u_2||u_1}(u_0) = c$ and $E_{u_2'||u_1'}(u_0') = c'$ such that $\psi[E](u_2, u_1, u_0) = \psi[E](u_2', u_1', u_0')$ we say these queries are "colliding". A single query is "colliding" if it collides with some earlier query. We note that if $E_{u_2||u_1}(u_0) = c$ then

$$\begin{aligned}\psi[E](u_2, u_1, u_0) &= u_0 + c||u_2(u_0 + c)^2 + u_1(u_0 + c) + u_0\\ &= u_0 + c||u_2(u_0 + c)^2 + (u_1 + 1)(u_0 + c) + c.\end{aligned}$$

Thus we may view the second half of output either as polynomial in $(u_0 + c)$ with coefficients determined by u_2, u_1, u_0 or as a polynomial in $(u_0 + c)$ with coefficients determined by u_2, u_1, c.

For every triplet of values (u_2, u_1, u_0), we define a curve

$$\mathcal{C}(u_2, u_1, u_0) = \{(x, y) \in \mathbb{F}_{2^n} \times \mathbb{F}_{2^n} : u_2(u_0 + x)^2 + u_1(u_0 + x) + u_0 = y\},$$

and for every triplet of values (u_2, u_1, c), we define a curve

$$\mathcal{D}(u_2, u_1, c) = \{(x, y) \in \mathbb{F}_{2^n} \times \mathbb{F}_{2^n} : u_2(x + c)^2 + (u_1 + 1)(x + c) + c = y\}.$$

We consider each curve $\mathcal{C}(u_2, u_1, u_0)$ and $\mathcal{D}(u_2, u_1, c)$ as a distinct "bin", into which we will place balls as described below. (We emphasize that every $\mathcal{C}$-curve is considered a distinct bin from every $\mathcal{D}$-curve, even though they may consist of the same set of points in $\mathbb{F}_{2^n} \times \mathbb{F}_{2^n}$.)

Say the adversary completes a query $E_{u_2'||u_1'}(u_0') = c'$ and let $t' = u_2'(u_0' + c')^2 + u_1'(u_0' + c') + u_0'$. Then for every tuple $(u_2, u_1, u_0, c) \neq (u_2', u_1', u_0', c')$ such that $u_0 + c = u_0' + c'$, we place a ball (c, t')

in the bin $\mathcal{C}(u_2, u_1, u_0)$ if (c, t') is actually a point on the curve $\mathcal{C}(u_2, u_1, u_0)$, and we place a ball (u_0, t') in the bin $\mathcal{D}(u_2, u_1, c)$ if (u_0, t') is a point on $\mathcal{D}(u_2, u_1, c)$. (Thus, the placement of balls in bins may be viewed as "highlighting" or selecting points on the curves.) We never "duplicate" (i.e., add or "highlight" twice) a ball inside a bin once it has already been added. We note that a query adds at most one ball to any bin, as c cannot be modified without changing u_0 and vice-versa.

To make matters a little more complicated, we finally define certain balls to be *phantom balls*. For a bin $\mathcal{C}(u_2, u_1, u_0)$, the ball added by the inverse query $E^{-1}_{u_2||u_1+1}(u_0)$ is called a phantom ball, and for a bin $\mathcal{D}(u_2, u_1, c)$, the ball added by the forward query $E_{u_2||u_1+1}(c)$ is called a phantom ball. Therefore, at most one ball per bin is a phantom ball. Intuitively, phantom balls pose a problem because forecasting the appearance of a phantom ball in a bin does not imply being able to forecast the answer to a query. Thankfully, there are few phantom balls per bin.

We let γ be the maximum number of non-phantom balls in a bin at the end of the attack (which differs at most by 1 from the maximum total number of balls in a bin). By assumption that $\mathcal{A}$ finds a collision with probability ϵ, one of the following two events occurs with probability at least $\epsilon/2$.

Case 1: $\mathcal{A}$ finds a collision and $\gamma \leq \log q + 2$. For this case, we can construct a forger $\mathcal{B}$ for E as follows.

1. $\mathcal{B}$ chooses $i \in \{1, \ldots, q\}$ uniformly at random.
2. $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine and faithfully answers the queries made by $\mathcal{A}$ until the $(i-1)$-th query.
3. If the i-th query of $\mathcal{A}$ is a forward query $E_{u_2||u_1}(u_0)$, $\mathcal{B}$ chooses a random (potentially phantom) ball (c, z) from the bin $\mathcal{C}(u_2, u_1, u_0)$, and guesses $E_{u_2||u_1}(u_0) = c$. (If the bin $\mathcal{C}(u_2, u_1, u_0)$ is empty, then $\mathcal{B}$ gives up.)
4. If the i-th query of $\mathcal{A}$ is an inverse query $E^{-1}_{u_2||u_1}(c)$, $\mathcal{B}$ chooses a random (potentially phantom) ball (u_0, z) from the bin $\mathcal{D}(u_2, u_1, c)$, and guesses $E^{-1}_{u_2||u_1}(c) = u_0$. (If the bin $\mathcal{D}(u_2, u_1, c)$ is empty, then $\mathcal{B}$ gives up.)

To analyze the chance of success of this strategy, assume that $\mathcal{A}$ obtains a collision with an inverse query $E^{-1}_{u_2||u_1}(c) = u_0$. Then $\mathcal{B}$ has chance $1/q$ of correctly guessing the index i of this query. Moreover, because $\mathcal{A}$ obtains a collision, $\mathcal{A}$ previously completed a query $E_{u_2'||u_1'}(u_0') = c'$ such that $u_0 + c = u_0' + c'$ and

$$u_2(u_0 + c)^2 + (u_1 + 1)(u_0 + c) + c = u_2'(u_0' + c')^2 + u_1'(u_0' + c') + u_0' = t',$$

which is to say that the ball (u_0, t') appears in the bin $\mathcal{D}(u_2, u_1, c)$. Since at most $\gamma + 1 \leq \log q + 3$ balls are in the bin $\mathcal{D}(u_2, u_1, c)$, $\mathcal{B}$ has advantage

$$\mathbf{Adv}^{\mathsf{unp}}_E(\mathcal{B}) \geq \frac{\epsilon}{2} \cdot \frac{1}{q} \cdot \frac{1}{\log q + 3} \tag{9}$$

A similar argument treats forward queries. Note that it takes $\mathcal{B}$ time $O(n^2 q)$ to enumerate the balls in a given bin $\mathcal{D}(u_2, u_1, c)$ or $\mathcal{C}(u_2, u_1, u_0)$.

Case 2: $\mathcal{A}$ produces $\gamma > \log q + 2$. For this case, $\mathcal{B}$ attempts to forge the value of E by using the MTF game played only with the *non-phantom balls*. We set $m_1 = 2$ and $m_2 = \log q + 2$; we are guaranteed that some bin contains more than m_2 balls at the end since $\gamma > \log q + 2$ is the number of non-phantom balls. To apply Corollary 1, we need to upper bound M, the number of bins that have at least three balls at the end of the game. For this purpose, note that if a completed query $E_{u_2'||u_1'}(u_0') = c'$ adds a ball (c, t') where $t' = u_2'(u_0' + c')^2 + u_1'(u_0' + c') + u_0'$ to a bin $C(u_2, u_1, u_0)$, then $(u_0' + c', t')$ is a point on the curve

$$\mathcal{E} = \{(x, y) \in \mathbb{F}_{2^n} \times \mathbb{F}_{2^n} : u_2 x^2 + u_1 x + u_0 = y\}.$$

Thus three non-colliding queries that produce balls in the same bin $C(u_2, u_1, u_0)$ must each determine distinct values $u'_0 + c'$ and, thus, uniquely determine the curve $\mathcal{E}$ and the bin $C(u_2, u_1, u_0)$. Therefore the number of C-bins with three or more balls is at most $\binom{q}{3}$. As a similar argument applies for $\mathcal{D}$-bins, we have $M \leq 2\binom{q}{3}$.

We also need to check that if $\mathcal{B}$ correctly guesses a bin for an oncoming ball then $\mathcal{B}$ can actually use this guess to forge the value of E or E^{-1}. Assume that the adversary has made a forward query $E_{u'_2||u'_1}(u'_0)$ and that $\mathcal{B}$ has correctly guessed this query will result in a new ball (u_0, t') being placed in the bin $\mathcal{D}(u_2, u_1, c)$. The values which are unknown to $\mathcal{B}$ are u_0 and $c' = E_{u'_2||u'_1}(u'_0)$. The constraints are $u'_0 + c' = u_0 + c$ and

$$u_2(u_0 + c)^2 + (u_1 + 1)(u_0 + c) + c = u'_2(u'_0 + c')^2 + u'_1(u'_0 + c') + u'_0,$$

which is

$$(u_2 + u'_2)z^2 + (u_1 + 1 + u'_1)z + c + u'_0 = 0,$$

where $z = u'_0 + c' = u_0 + c$. As long as this polynomial in z is nonzero, $\mathcal{B}$ can guess the right root z with probability $\frac{1}{2}$ and solve for c' using $c' = z + u'_0$. (If the polynomial is a nonzero constant, it simply means that the original system is not solvable, and that $\mathcal{B}$ guessed the wrong bin.) But if $u_2 + u'_2 = u_1 + 1 + u'_1 = c + u'_0 = 0$, we precisely obtain (the second case of) a phantom ball, so $\mathcal{B}$ could not guess this ball anyway. Similarly in the case where $\mathcal{A}$ makes an inverse query $E^{-1}_{u'_2||u'_1}(c')$ and $\mathcal{B}$ correctly guesses the query will result in a new ball (c, t') being added to a bin $\mathcal{C}(u_2, u_1, c)$, $\mathcal{B}$ can always forge with probability $\frac{1}{2}$ because of the exclusion of (the first type of) phantom balls. Finally, in the remaining two combinations (forward query and $\mathcal{C}$-bin guess or inverse query and $\mathcal{D}$-bin guess) $\mathcal{B}$ can also forge with probability $\frac{1}{2}$ using the stipulation $(u_2, u_1, u_0, c) \neq (u'_2, u'_1, u'_0, c')$, as is not hard to see.

By Corollary 1, $\mathcal{B}$ has forgery advantage

$$\mathbf{Adv}^{\mathsf{unp}}_E(\mathcal{B}) \geq \frac{\epsilon}{2} \cdot \frac{1}{2} \cdot \frac{1}{q} \left(\frac{1}{qM}\right)^{1/(m_2 - m_1)} \geq \frac{\epsilon}{4} \cdot \frac{1}{q} \left(\frac{1}{q^4}\right)^{1/\log q} = \frac{\epsilon}{64q}. \tag{10}$$

From (9) and (10), it follows that

$$\begin{aligned} \mathbf{Adv}^{\mathsf{coll}}_{\psi[E]}(t, q) &\leq 2q \max(\log q + 3, 32)\, \mathbf{Adv}^{\mathsf{unp}}_E\left(t + O(n^2q^4) + \mathsf{Time}_2, q\right) \\ &= 2q(\log q + 3)\, \mathbf{Adv}^{\mathsf{unp}}_E\left(t + O(n^2q^4) + \mathsf{Time}_2, q\right), \end{aligned}$$

where $O(n^2q^4) + \mathsf{Time}_2$ is the overhead of the forger for the second case. Now the second part of the theorem is immediate from Lemma 1. □

If E is an ideal cipher, then

$$\mathbf{Adv}^{\mathsf{unp}}_E(t, q) \leq \frac{1}{2^n - q},$$

for any q and t. Therefore, by Theorem 7 and Lemma 5, we obtain the following theorem.

Theorem 8. *Let $E : \{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^n$ be an ideal blockcipher and let $f_2 : \{0,1\}^{2n} \to \{0,1\}^n$ be a random functions. Then, $\mathbf{Adv}^{\mathsf{coll}}_{\psi[E]}(q) \leq \epsilon(q)$ for*

$$\epsilon(q) = \frac{2q(\log q + 3)}{2^n - q},$$

and $G[E, f_2] = f_2 \circ MD[\psi[E]]$ is $(\tilde{q}, \epsilon(\bar{q}))$-indistinguishable from a random function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$, where

$$\bar{q} = \mathsf{NQ}(MD[\psi[E]], l_{max})\tilde{q} = \left\lceil \frac{l_{max} + 1}{n} + 1 \right\rceil \tilde{q},$$

for the length in bits l_{max} of the longest query made by a distinguisher.

6 The Quadratic Cascade Compression Function

In this section, we discuss a concrete instantiation of the quadratic compression function (polynomial-based compression function of degree $d = 2$) using the cascade of two $2n \to n$ functions. In the *quadratic cascade compression function*, the compression function f_1 is implemented by the cascade of two compression functions $h_1, h_2 : \{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^n$ by letting $f_1(x||y||z) = h_2(h_1(x||y)||z)$ as described in Figure 7. We write $\tau[h_1, h_2]$ for the resulting compression function. Thus $\tau[h_1, h_2] : \{0,1\}^{3n} \to \{0,1\}^n$ and $\tau[h_1, h_2](x||y||z) = w||xw^2 + yw + z$, where $v = h_1(x||y)$ and $w = h_2(v||z)$.

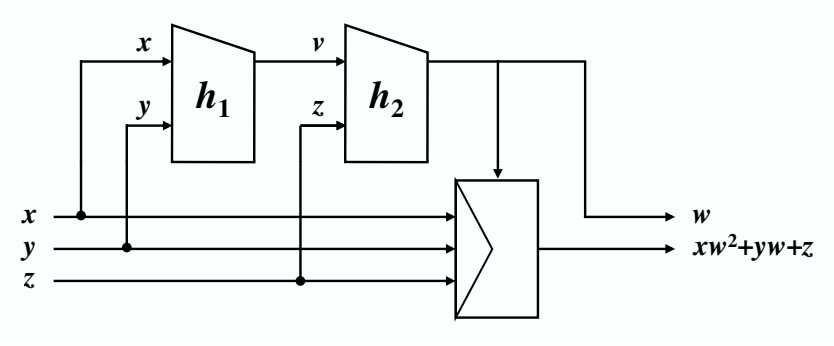


Fig. 7. The quadratic cascade compression function.

Theorem 9. *Let $\tau = \tau[h_1, h_2]$ be a function family where h_1 and h_2 are independently chosen from a function family h, respectively. Then,*

$$\mathbf{Adv}_{\tau}^{\mathsf{wcr}}(t, q) \leq 26q \log q \left(\log q + 1\right)^2 \mathbf{Adv}_{h}^{\mathsf{mac}}\left(t + O(n^2 q^4 \log^4 q) + \mathsf{Time}_2, q\right).$$

Furthermore, let $G = G[h_1, h_2, f_2] = f_2 \circ MD\left[\tau[h_1, h_2]\right]$ be a function family where f_2 is chosen from a function family f. Then for $q = \lfloor \mu/c \rfloor + 2\tilde{q}$,

$$\mathbf{Adv}_{G}^{\mathsf{mac}}(t, \tilde{q}, \mu) \leq \mathbf{Adv}_{f}^{\mathsf{mac}}(t, \tilde{q}) + 26q \log q \left(\log q + 1\right)^2 \mathbf{Adv}_{h}^{\mathsf{mac}}\left(t + O(n^2 q^4 \log^4 q) + \mathsf{Time}_2, q\right).$$

In order to prove Theorem 9, we slightly modify the MTF game described in Section 4 by allowing a bin to receive a multiple number of balls at each round. By "collapsing" all balls introduced into a bin at a given round into a single ball, we can prove the following lemma.

Lemma 6. *If (i) each bin receives at most R balls at each round, (ii) the number of bins that contain at least $m_1 + 1$ balls at the end of the MTF game is at most M, and (iii) some bin eventually contains more than Rm_2 balls, then $\mathcal{B}$ can win the MTF game with probability at least $1/q \cdot (1/qM)^{1/(m_2 - m_1)}$.*

Proof (Theorem 9). Let $\mathcal{A}$ be a weak-collision finding adversary for $\tau = \tau[h_1, h_2]$ such that

$$\mathbf{Adv}_{\tau}^{\mathsf{wcr}}(\mathcal{A}) = \mathbf{Adv}_{\tau}^{\mathsf{wcr}}(t, q) = \epsilon.$$

Without loss of generality, we give $\mathcal{A}$ direct oracle access to h_1 and h_2, allowing $\mathcal{A}$ at most q queries to h_1 and q queries to h_2. Due to this assumption, the collision resistance of $\tau[h_1, h_2]$ in the ideal primitive model is derived from this theorem. We also assume that $\mathcal{A}$ makes the queries necessary to verify its collision.

We construct a forging adversary $\mathcal{B}$ for the function family h from $\mathcal{A}$. The adversary $\mathcal{B}$ has oracle access to a single member h_0 of h. First, $\mathcal{B}$ flips a coin and samples a random key $k \in \{0,1\}^{\kappa}$.

If the coin is heads, $\mathcal{B}$ simulates $\mathcal{A}$ by answering $\mathcal{A}$'s queries with $h_1 = h_0$ and $h_2 = h_k$, otherwise $\mathcal{B}$ simulates $\mathcal{A}$ by answering $\mathcal{A}$'s queries with $h_1 = h_k$ and $h_2 = h_0$. Then $\mathcal{B}$ "forgets" which of the two worlds it is in, treating both h_1 and h_2 as oracles and attempting to forge either one of them—note that since the two worlds are indistinguishable once $\mathcal{B}$ "forgets" its choice, a successful forgery in the new two-oracle model gives a successful forgery of h_0 with probability $1/2$.

We now define 13 events with respect to the final contents of $\mathcal{A}$'s query history as follows.

$\mathsf{Win}_1 \Leftrightarrow \mathcal{A}$ finds a $(\log q + 1)$-collision for h_1.

$\mathsf{Win}_2 \Leftrightarrow \mathcal{A}$ finds a $(\log q + 1)$-collision for h_2.

$\mathsf{Win}_3 \Leftrightarrow 0^n$ is ever returned as a the answer to an h_2 query.

$\mathsf{Win}_4 \Leftrightarrow$ there exist a tuple $(\overline{x}, \overline{y}, z, v) \in \{0,1\}^{4n}$ such that $(\overline{x}, \overline{y}) \neq (0,0)$, and $k = \log q + 3$ distinct queries $h_2(v, z'_1), \ldots, h_2(v, z'_k)$ such that $\overline{x}w_i^2 + \overline{y}w_i + z = z'_i$ for $i = 1, \ldots, k$, where $w_i = h_2(v, z'_i)$.

$\mathsf{Win}_5 \Leftrightarrow$ there exist a tuple $(\overline{x}, \overline{y}, z) \in \{0,1\}^{3n}$ such that $(\overline{x}, \overline{y}) \neq (0,0)$, and a set of $k = \log q + 3$ distinct pairs $\{(w_1, z'_1), \ldots, (w_k, z'_k)\}$ such that $w_1 = h_2(v'_1, z'_1), \ldots, w_k = h_2(v'_k, z'_k)$ for some queries $h_2(v'_1, z'_1), \ldots, h_2(v'_k, z'_k)$ and $\overline{x}w_i^2 + \overline{y}w_i + z = z'_i$ for $i = 1, \ldots, k$.

$\mathsf{Win}_6 \Leftrightarrow$ there exist a tuple $(x, y, z) \in \{0,1\}^{3n}$ and a set of $k = (\log q + 2)^2 + 1$ distinct values $w_1, \ldots, w_k$ such that for each w_i there exists a pair of queries $(h_1(x'_i, y'_i), h_2(v'_i, z'_i))$ such that $(x'_i, y'_i) \neq (x, y)$ and $xw_i^2 + yw_i + z = x'_i w_i^2 + y'_i w_i + z'_i$, where $v'_i = h_1(x'_i, y'_i)$ and $w_i = h_2(v'_i, z'_i)$.

$\mathsf{Win}_7 \Leftrightarrow$ there exist a pair $(\overline{x}, \overline{y}) \in \{0,1\}^{2n}$ such that $(\overline{x}, \overline{y}) \neq (0,0)$, and $k = \log q + 2$ distinct values $w_1, \ldots, w_k$ such that for each w_i there exists a pair of distinct queries $(h_2(v_i, z_i), h_2(v'_i, z'_i))$ such that $h_2(v_i, z_i) = h_2(v'_i, z'_i) = w_i$ and $\overline{x}w_i^2 + \overline{y}w_i = z_i + z'_i$.

$\mathsf{Win}_8 \Leftrightarrow$ there exist a pair $(x, y) \in \{0,1\}^{2n}$ and $k = (\log q + 1)^2 + 1$ distinct values $w_1, \ldots, w_k$ such that for each $i = 1, \ldots, k$ there exists a triple of queries $(h_2(v_i, z_i), h_1(x'_i, y'_i), h_2(v'_i, z'_i))$ such that $(x'_i, y'_i) \neq (x, y)$, $v'_i = h_1(x'_i, y'_i)$, $(v_i, z_i) \neq (v'_i, z'_i)$, $h_2(v_i, z_i) = h_2(v'_i, z'_i) = w_i$ and $xw_i^2 + yw_i + z_i = x'_i w_i^2 + y'_i w_i + z'_i$.

$\mathsf{Win}_9 \Leftrightarrow \mathcal{A}$ makes a query $h_1(x, y) = v$ such that the pre-existing query history contains a triple of queries $(h_2(v, z), h_1(x', y'), h_2(v', z'))$ such that $(v, z) \neq (v', z')$, $h_1(x', y') = v'$, $h_2(v, z) = h_2(v', z')$ and $xw^2 + yw + z = x'w^2 + y'w + z'$ where $w = h_2(v', z') = h_2(v, z)$.

$\mathsf{Win}_{10} \Leftrightarrow \mathcal{A}$ makes a query $h_2(v, z) = w$ such that the pre-existing query history contains a triple of queries $(h_1(x, y), h_1(x', y'), h_2(v', z'))$ such that $(x, y) \neq (x', y')$, $h_1(x, y) = v$, $h_1(x', y') = v'$, $h_2(v', z') = w$ and $xw^2 + yw + z = x'w^2 + y'w + z'$ where $w = h_2(v', z')$.

$\mathsf{Win}_{11} \Leftrightarrow \mathcal{A}$ makes a query $h_2(v, z) = w$ such that the pre-existing query history contains a pair of distinct queries $(h_1(x, y), h_1(x', y'))$ such that $h_1(x, y) = h_1(x', y') = v$ and $xw + y = x'w + y'$.

$\mathsf{Win}_{12} \Leftrightarrow$ there exists a pair $(x, y) \in \{0,1\}^{2n}$ and $k = \log q + 2$ distinct values $w_1, \ldots, w_k$ such that for each w_i there exists a pair of queries $(h_1(x'_i, y'_i), h_2(v'_i, z'_i))$ such that $h_2(v'_i, z'_i) = w_i$, $h_1(x'_i, y'_i) = v'_i$, $(x'_i, y'_i) \neq (x, y)$ and $xw_i + y = x'_i w_i + y'_i$ for $i = 1, \ldots, k$.

$\mathsf{Win}_{13} \Leftrightarrow \mathcal{A}$ makes a query $h_1(x, y) = v'$ such that the pre-existing query history contains a pair of queries $(h_1(x', y'), h_2(v', z'))$ such that $(x, y) \neq (x', y')$, $v' = h_1(x', y')$ and $xw + y = x'w + y'$ where $w = h_2(v', z')$.

Let $\mathsf{Ev}_i = \mathsf{Win}_i \bigwedge \neg \left(\bigvee_{j<i} \mathsf{Win}_i \right)$ for $i = 1, \ldots, 13$. Since two distinct colliding inputs (x, y, z) and (x', y', z') must have $(x, y) \neq (x', y')$, a collision either uses four distinct queries $h_1(x, y)$, $h_2(v, z)$, $h_1(x', y')$, $h_2(v', z')$ or three distinct queries $h_1(x, y)$, $h_2(v, z)$, $h_1(x', y')$. It is thus easy to check that $\mathsf{Coll} \Rightarrow \mathsf{Win}_3 \vee \mathsf{Win}_9 \vee \mathsf{Win}_{10} \vee \mathsf{Win}_{11} \vee \mathsf{Win}_{13}$, and therefore

$$\mathsf{Coll} \Rightarrow \mathsf{Win}_1 \vee \cdots \vee \mathsf{Win}_{13} \Rightarrow \mathsf{Ev}_1 \vee \ldots \vee \mathsf{Ev}_{13},$$

where Coll is the event that $\mathcal{A}$ finds a collision. By assumption that $\mathcal{A}$ succeeds to find a collision with probability ϵ, one of the events $\mathsf{Ev}_1, \ldots, \mathsf{Ev}_{13}$ occurs with probability at least $\epsilon/13$. For each event, we will construct a forger $\mathcal{B}$ of either h_1 or h_2, and analyze its success probability under the condition of the event. Especially, eight of these events use MTF games. For these games, each ball comes with a "label" which may be a query, a tuple of queries, or any other string. By convention, *balls with the same label are never added twice to a bin*: replicates do not contribute. We also stress that $\mathcal{B}$ does not need to or guess the labels of balls, but only to guess a bin that is going to receive a ball as the result of the answer to $\mathcal{A}$'s last query.

Ev_1: $\mathcal{B}$ plays a balls-in-bins game where bins are values $v \in \{0,1\}^n$ and balls are queries to h_1, where a query $h_1(x, y)$ goes in bin v if $h_1(x, y) = v$. The parameters for the game are $m_1 = 0$, $m_2 = \log q$, $M = q$ and $R = 1$. Here forgery follows obviously from the game: guessing a bin for a ball is guessing h_1. By Lemma 6, the success probability of $\mathcal{B}$ is at least $1/q \cdot (1/qM)^{\frac{1}{m_2 - m_1}} = 1/q \cdot \left(1/q^2\right)^{\frac{1}{\log q}} = 1/(4q)$.

Ev_2: Similar to the first case, we can construct a forger $\mathcal{B}$ of h_2 with success probability at least $1/(4q)$.

Ev_3: (Not a balls-in-bins game.) $\mathcal{B}$ uniformly selects an index i between 1 and q and at the i-th query to h_2, $\mathcal{B}$ guesses that the answer will be 0^n. Then $\mathcal{B}$ has chance at least $1/q$ of forging h_2.

Ev_4: $\mathcal{B}$ plays a balls-in-bins game where bins are tuples $(\overline{x}, \overline{y}, z, v) \in \{0,1\}^{4n}$ with $(\overline{x}, \overline{y}) \neq (0, 0)$ and balls are queries $h_2(v', z')$. A ball $h_2(v', z')$ goes into bin $(\overline{x}, \overline{y}, z, v)$ if $v = v'$ and if $\overline{x}w^2 + \overline{y}w + z = z'$ where $w = h_2(v', z')$. Obviously, the number of balls that enters a bin at any round is at most $R = 1$.

Let $m_1 = 2$ and $m_2 = \log q + 2$. If $\{h_2(v, z_1'), h_2(v, z_2'), h_2(v, z_3')\}$ are any three balls in a bin $(\overline{x}, \overline{y}, z, v)$, then z_1', z_2', z_3' are distinct, and so $\{h_2(v, z_1'), h_2(v, z_2'), h_2(v, z_3')\}$ *uniquely* determine the bin $(\overline{x}, \overline{y}, z, v)$ (from the equations $\overline{x}w_i^2 + \overline{y}w_i + z = z_i'$ for $i = 1, 2, 3$ where $w_i = h_2(v, z_i')$). Therefore the number of bins with more than m_1 balls is at most $\binom{q}{3} \leq q^3 = M$. Since some bin eventually contains more than Rm_2 balls, the success probability of $\mathcal{B}$ for the balls-in-bins game is at least

$$\frac{1}{q} \left(\frac{1}{qM} \right)^{\frac{1}{\log q}} \geq \frac{1}{16q}.$$

Once $\mathcal{B}$ guesses that a query $h_2(v', z')$ will go into bin $(\overline{x}, \overline{y}, z, v)$, $\mathcal{B}$ selects at random one of the two roots w of the quadratic equation $\overline{x}w^2 + \overline{y}w + z = z'$ (using $(\overline{x}, \overline{y}) \neq (0, 0)$), and guesses that this will be the output of $h_2(v', z')$. Therefore, the overall probability of success is $1/(32q)$, since $\mathcal{B}$ has chance $1/2$ of choosing the right root w if it guesses the right bin.

Ev_5: $\mathcal{B}$ plays a balls-in-bins game where bins are tuples $(\overline{x}, \overline{y}, z) \in \{0,1\}^{3n}$ with $(\overline{x}, \overline{y}) \neq (0, 0)$ and balls are pairs (w, z') such that $w = h_2(v', z')$ for each h_2-query $h_2(v', z')$. A ball (w, z') goes into bin $(\overline{x}, \overline{y}, z)$ if $\overline{x}w^2 + \overline{y}w + z = z'$. Since a query $w = h_2(v', z')$ can only add at most one pair (w, z') to any bin, we have $R = 1$.

Let $m_1 = 2$ and $m_2 = \log q + 2$. Since three distinct balls (w_1, z_1'), (w_2, z_2'), (w_3, z_3') can simultaneously appear in at most one bin, the number of bins with more than m_1 balls at the end of the game is at most $M = q^3$. Since some bin eventually contains more than Rm_2 balls, the success probability of $\mathcal{B}$ for the balls-in-bins game is at least $1/(16q)$.

Once $\mathcal{B}$ guesses that a query $h_2(v', z')$ will go into bin $(\overline{x}, \overline{y}, z)$, $\mathcal{B}$ selects at random one of the two roots w of the quadratic equation $\overline{x}w^2 + \overline{y}w + z = z'$ (using $(\overline{x}, \overline{y}) \neq (0,0)$), and guesses that this will be the output of $h_2(v', z')$. Therefore, the overall success probability of $\mathcal{B}$ is at least $1/(32q)$.

Ev_6: $\mathcal{B}$ plays a balls-in-bins game where bins are tuples $(x, y, z) \in \{0,1\}^{3n}$ and balls are values $w \in \{0,1\}^n$. A ball w goes into bin (x, y, z) if there exists a pair of queries $(h_1(x', y') = v', h_2(v', z') = w)$ such that $(x, y) \neq (x', y')$ and $xw^2 + yw + z = x'w^2 + y'w + z'$.

When $\mathcal{A}$ makes an h_1-query $h_1(x', y') = v'$, the number of balls that can be placed in a bin $(x, y, z) \in \{0,1\}^{3n}$, $(x, y) \neq (x', y')$, is at most the number of pre-existing queries $h_2(v', z') = w$ such that $xw^2 + yw + z = x'w^2 + y'w + z'$, namely such that $(x - x')w^2 + (y - y')w + z = z'$. Since $\mathsf{Ev}_6 \Rightarrow \neg\mathsf{Win}_4$, and since $(x - x', y - y') \neq (0,0)$, there are at most $\log q + 2$ such queries $h_2(v', z')$. Since when $\mathcal{A}$ makes a query $h_2(v', z')$ at most one new ball can be placed in any bin, the maximum number of balls that can be placed in a bin at any round is $R = \log q + 2$.

Let $m_1 = 2$ and $m_2 = \log q + 2$. Say a bin (x, y, z) is *associated* to a query pair $(h_1(x', y') = v', h_2(v', z') = w)$ if $(x', y') \neq (x, y)$ and $xw^2 + yw + z = x'w^2 + y'w + z'$ (in particular this implies that w is a ball in (x, y, z), but w may be associated to other query pairs as well). Thus for every ball in a bin there is at least one associated query pair for the ball. Distinct balls always correspond to distinct query pairs. For any three query pairs $\{(h_1(x'_i, y'_i) = v'_i, h_2(v'_i, z'_i) = w_i),\ i = 1,2,3\}$ such that w_1, w_2, w_3 are distinct, there is at most one bin (x, y, z) containing the balls w_1, w_2, w_3 associated to these query pairs, as (x, y, z) must solve the three equations $xw_i^2 + yw_i + z = x'_i w_i^2 + y'_i w_i + z'_i$, $i = 1,2,3$. Since there are at most $q \log q$ query pairs $(h_1(x', y'), h_2(v', z'))$ such that $h_1(x', y') = v'$ (using $\neg\mathsf{Win}_1$), there are at most $q^3 \log^3 q$ triples of query pairs, so at most $M = q^3 \log^3 q$ bins that receive at least $m_1 + 1 = 3$ balls. Since some bin eventually contains more than Rm_2 balls, the success probability of $\mathcal{B}$ for the balls-in-bins game is at least

$$\frac{1}{q}\left(\frac{1}{q^4 \log^3 q}\right)^{\frac{1}{\log q}} = \frac{1}{16q\,(\log q)^{3/\log q}}.$$

If $\mathcal{B}$ correctly guesses that an h_2-query $h_2(v', z')$ is going to result in a new ball appearing in a bin (x, y, z), then because $\neg\mathsf{Win}_1$ there are at most $\log q$ candidates for the query $h_1(x', y')$ such that $h_1(x', y') = v'$ and $(x', y') \neq (x, y)$, and for each of these chance $1/2$ of guessing the correct root w of $(x - x')w^2 + (y - y')w + (z - z') = 0$.

On the other hand if $\mathcal{B}$ correctly guesses that an h_1-query $h_1(x', y')$ is going to result in a new ball appearing in a bin (x, y, z), then because $\neg\mathsf{Win}_5$ there are at most $\log q + 2$ points (w_i, z'_i) such that $xw_i^2 + yw_i + z = x'w_i^2 + y'w_i + z'_i$ and such that $w_i = h_2(v'_i, z'_i)$ for some query $h_2(v'_i, z'_i)$. For each pair (w_i, z'_i), there are at most $\log q$ values v'_i such that $w_i = h_2(v'_i, z'_i)$ for some query $h_2(v'_i, z'_i)$ (using $\neg\mathsf{Win}_2$). Therefore, each guess makes a total of $\log q(\log q + 2)$ possibilities for the value v'. Thus in this case $\mathcal{B}$ has chance at least $1/(2\log q(\log q + 2))$ of forging. Overall, $\mathcal{B}$'s chance of forging is at least

$$\frac{1}{32q \log q\,(\log q + 2)\,(\log q)^{3/\log q}}.$$

Ev_7: $\mathcal{B}$ plays a balls-in-bins game where bins are pairs $(\overline{x}, \overline{y}) \in \{0,1\}^{2n}$ with $(\overline{x}, \overline{y}) \neq (0,0)$ and balls are values $w \in \{0,1\}^n$. A ball w goes into bin $(\overline{x}, \overline{y})$ if there exists a pair of distinct queries $(h_2(v, z), h_2(v', z'))$ with $h_2(v, z) = h_2(v', z') = w$ and $\overline{x}w^2 + \overline{y}w = z + z'$. Since balls correspond to outputs w of h_2, at most $R = 1$ ball can go into a bin at any given round.

Let $m_1 = 1$ and $m_2 = \log q + 1$. Because $\mathsf{Ev}_7 \Rightarrow \neg\mathsf{Win}_2$, there are at most $q \log q$ distinct pairs $(h_2(v, z), h_2(v', z'))$ in the query history such that $h_2(v, z) = h_2(v', z')$. Thus there are at most $q^2 \log^2 q$ 4-tuples of queries $(h_2(v_1, z_1), h_2(v'_1, z'_1), h_2(v_2, z_2), h_2(v'_2, z'_2))$ such that $h_2(v_1, z_2) = h_2(v'_1, z'_2) = w_1 \neq w_2 = h_2(v_2, z_2) = h_2(v'_2, z'_2)$. Since $\neg\mathsf{Win}_3$, each such 4-tuple

uniquely determines a bin $(\overline{x}, \overline{y})$ such that

$$\overline{x}w_1^2 + \overline{y}w_1 = z_1 + z_1',$$
$$\overline{x}w_2^2 + \overline{y}w_2 = z_2 + z_2'.$$

(Note the determinant $w_1^2 w_2 + w_1 w_2^2$ for this system is zero if and only if $w_1 = w_2$, as $w_1 w_2 \neq 0$.) Conversely, there must exist such a 4-tuple for any bin $(\overline{x}, \overline{y})$ with at least $m_1 + 1 = 2$ balls in it, so the total number of bins with more than m_1 balls is at most $M = q^2 \log^2 q$. Since some bin eventually contains more than Rm_2 balls, the success probability of $\mathcal{B}$ for the balls-in-bins game is at least

$$\frac{1}{q}\left(\frac{1}{qM}\right)^{\frac{1}{\log q}} = \frac{1}{q}\left(\frac{1}{q^3 \log^2 q}\right)^{\frac{1}{\log q}} = \frac{1}{8q\left(\log q\right)^{\frac{2}{\log q}}}.$$

If $\mathcal{B}$ correctly guesses that an h_2-query $h_2(v, z)$ is going to add a ball to bin $(\overline{x}, \overline{y})$, then because $\mathsf{Ev}_7 \Rightarrow \neg\mathsf{Win}_5$, there are at most $\log q + 2$ pairs (w, z) such that $w = f(v', z')$ for some pre-existing query $h_2(v', z')$ and such that $\overline{x}w^2 + \overline{y}w + z = z'$. Thus in this case $\mathcal{B}$ has chance at least $1/(\log q + 2)$ of forging. Overall, $\mathcal{B}$'s chance of forging is at least

$$\frac{1}{8q\left(\log q + 2\right)\left(\log q\right)^{\frac{2}{\log q}}}.$$

Ev_8: $\mathcal{B}$ plays a balls-in-bins game where bins are pairs $(x, y) \in \{0,1\}^{2n}$ and balls are values $w \in \{0,1\}^n$. A ball w goes into bin (x, y) if there exists a triple of queries $(h_2(v, z), h_1(x', y'), h_2(v', z'))$ such that $(x', y') \neq (x, y)$, $v' = h_1(x', y')$, $(v, z) \neq (v', z')$, $h_2(v, z) = h_2(v', z') = w$ and $xw^2 + yw + z = x'w^2 + y'w + z'$.

Because balls correspond to output values of h_2, at most one ball can be added to any bin when $\mathcal{A}$ makes an h_2 query. On the other hand when $\mathcal{A}$ makes an h_1-query $h_1(x', y') = v'$, the number of balls that can be added to a bin (x, y) is at most $\log q + 1$ because $\mathsf{Ev}_8 \Rightarrow \neg\mathsf{Win}_7$. Thus the maximum number of balls added to a bin at any round is $R = \log q + 1$.

Let $m_1 = 1$ and $m_2 = \log q + 1$. For any two distinct balls w_1 and w_2, we can upper bound the number of bins that can contain both w_1 and w_2: for $i = 1, 2$ there are at most $\log^2 q$ pairs of queries $(h_2(v_i, z_i), h_2(v_i', z_i'))$ such that $h_2(v_i, z_i) = h_2(v_i', z_i') = w_i$, and for each value of v_i' there are at most $\log q$ queries $h_1(x_i', y_i')$ such that $h_1(x_i', y_i') = v_i'$. Moreover given values w_1, z_1, x_1', y_1', z_1' and w_2, z_2, x_2', y_2', z_2', there is exactly one solution (x, y) to the system

$$xw_1^2 + yw_1 + z_1 = x_1'w_1^2 + y_1'w_1 + z_1'$$
$$xw_2^2 + yw_2 + z_2 = x_2'w_2^2 + y_2'w_2 + z_2'$$

since $w_1 w_2 \neq 0$ (as $\mathsf{Ev}_7 \Rightarrow \neg\mathsf{Win}_3$). Since there are at most q^2 choices for w_1, w_2, there are at most $M = q^2 \log^6 q$ bins containing at least $m_1 + 1 = 2$ balls at the end of the game. Since some bin eventually contains more than Rm_2 balls, the success probability of $\mathcal{B}$ for the balls-in-bins game is at least

$$\frac{1}{q}\left(\frac{1}{qM}\right)^{\frac{1}{\log q}} = \frac{1}{q}\left(\frac{1}{q^3 \log^6 q}\right)^{\frac{1}{\log q}} = \frac{1}{8q\left(\log q\right)^{\frac{6}{\log q}}}.$$

If $\mathcal{B}$ guesses that an h_2-query is going to add a ball to a bin (x, y), then $\mathcal{B}$ flips a coin. If the coin is heads, then $\mathcal{B}$ regards the h_2-query as $h_2(v, z)$. Because $\neg\mathsf{Win}_6$ there exist at most $(\log q + 2)^2$ values w for which there exists a pair of queries $(h_1(x', y') = v', h_2(v', z') = w)$, $(x', y') \neq (x, y)$, such that $xw^2 + yw + z = x'w^2 + y'w + z'$. If the coin is tails, $\mathcal{B}$ regards the query as $h_2(v', z')$. Then there are at most $\log q$ queries $h_1(x', y')$ such that $h_1(x', y') = v'$, and for fixed values x, y, x', y', z' there are at most $\log q + 2$ pairs (w, z) such that $h_2(v, z) = w$ for some query $h_2(v, z)$ and such that $xw^2 + yw + z = x'w^2 + y'w + z'$, using $\neg\mathsf{Win}_5$. Thus for an h_2-query, $\mathcal{B}$ has chance of forging at least

$$\frac{1}{2}\min\left(\frac{1}{\left(\log q + 2\right)^2}, \frac{1}{\log q\left(\log q + 2\right)}\right) = \frac{1}{2\left(\log q + 2\right)^2},$$

if it wins the balls-in-bins game.

If $\mathcal{B}$ correctly guesses that an h_1-query $h_1(x', y')$ is going to add a ball to a bin (x, y) (where necessarily $(x, y) \neq (x', y')$), then because $\neg\mathsf{Win}_7$ there are at most $\log q + 1$ choices for a value w such that $xw^2 + yw + z = x'w^2 + y'w + z'$ where $w = h_2(v, z) = h_2(v', z')$ and $(v, z) \neq (v', z')$, and hence at most $\log q$ choices for v'. Thus in this case $\mathcal{B}$'s probability of forging is at least $1/(\log q(\log q + 1))$, greater than for h_2 queries. Thus $\mathcal{B}$'s overall chance of forging is at least

$$\frac{1}{16q(\log q + 2)^2 (\log q)^{\frac{6}{\log q}}}.$$

Ev_9: (Not a balls-in-bins game.) $\mathcal{B}$ randomly selects an index i between 1 and q. When $\mathcal{A}$ makes its i-th query $h_1(x, y)$, $\mathcal{B}$ selects uniformly at random among all the values w for which there exists a triple of queries $(h_2(v, z), h_1(x', y'), h_2(v', z'))$ such that $(v, z) \neq (v', z')$, $h_1(x', y') = v'$, $h_2(v, z) = h_2(v', z') = w$ and $xw^2 + yw + z = x'w^2 + y'w + z'$, selects uniformly at random among all values v for which there exists a query $h_2(v, z) = w$, and guesses $h_1(x, y) = v$.

Suppose that $\mathcal{A}$ makes a query $h_1(x, y)$ that triggers Win_9, and that $\mathcal{B}$ happens to guess the index i of this query right (with probability $1/q$). Since $\mathsf{Ev}_9 \Rightarrow \neg\mathsf{Win}_8$, there are at most $(\log q + 1)^2$ values w for which there exists a triple of queries $(h_2(v, z), h_1(x', y'), h_2(v', z'))$ such that $(v, z) \neq (v', z')$, $h_1(x', y') = v'$, $h_2(v, z) = h_2(v', z')$ and $xw^2 + yw + z = x'w^2 + y'w + z'$ where $w = h_2(v, z) = h_2(v', z')$. For each of these values w, there are at most $\log q$ possible queries $h_2(v, z)$ such that $h_2(v, z) = w$. Thus, $\mathcal{B}$ has chance of forging at least $1/\log q(\log q + 1)^2$ when it guesses i correctly, hence an overall chance of success of $1/\left(q \log q(\log q + 1)^2\right)$.

Ev_{10}: (Not a balls-in-bins game.) $\mathcal{B}$ randomly selects an index i between 1 and q. When $\mathcal{A}$ makes its i-th query $h_2(v, z)$, $\mathcal{B}$ selects uniformly at random among all the values w for which there exists a triple of queries $(h_1(x, y), h_1(x', y'), h_2(v', z'))$ such that $(x, y) \neq (x', y')$, $h_1(x', y') = v'$, $h_2(v', z') = w$ and $xw^2 + yw + z = x'w^2 + y'w + z'$, and guesses $h_2(v, z) = w = h_2(v', z')$.

Suppose that $\mathcal{A}$ makes a query $h_2(v, z)$ that triggers Win_{10}. Then there are at most $\log q$ queries $h_1(x, y)$ such that $h_1(x, y) = v$. Moreover, since $\neg\mathsf{Win}_6$, for given values of (x, y, z) there are at most $(\log q + 1)^2$ values w for which there exists a pair of queries $(h_1(x', y') = v', h_2(v', z') = w)$ such that $(x', y') \neq (x, y)$, and $xw^2 + yw + z = x'w^2 + y'w + z'$. Thus $\mathcal{B}$'s chance of forging is $1/\left(q \log q(\log q + 1)^2\right)$ since it has probability at least $1/q$ of choosing the right index i to makes its query.

Ev_{11}: (Not a balls-in-bins game.) $\mathcal{B}$ randomly selects an index i between 1 and q. When $\mathcal{A}$ makes its i-th query $h_2(v, z)$, $\mathcal{B}$ enumerates all pairs of distinct queries $(h_1(x', y'), h_1(x, y))$ such that $h_1(x, y) = h_1(x', y') = v$, selects one such pair uniformly at random, and guesses $h_2(v, z) = w$ where w is the unique solution of $xw + y = x'w + y'$.

For any query $h_2(v, z)$, there are at most $\log^2 q$ pairs of queries $(h_1(x, y), h_1(x', y'))$ such that $(x, y) \neq (x', y')$ and $h_1(x, y) = h_1(x', y') = v$ (using $\neg\mathsf{Win}_1$) and for each such pair there is a unique solution w to $xw + y = x'w + y'$. Thus $\mathcal{B}$ has chance at least $1/q \log^2 q$ of choosing both the right index i and the right value w.

Ev_{12}: $\mathcal{B}$ plays a balls-in-bins game where bins are pairs $(x, y) \in \{0,1\}^{2n}$ and balls are values $w \in \{0,1\}^n$. A ball w goes into bin (x, y) if there exists a pair of queries $(h_1(x', y'), h_2(v', z'))$ with $h_2(v', z') = w$, $h_1(x', y') = v'$, $(x', y') \neq (x, y)$ and $xw + y = x'w + y'$.

When $\mathcal{A}$ makes a query $h_1(x', y') = v'$, the number of new balls that are added to a bin $(x, y) \neq (x', y')$ is at most the number of values w such that $xw + y = x'w + y'$, namely at most one. When $\mathcal{A}$ makes a query $h_2(v', z') = w$, the only new ball that can enter *any* bin is w. Thus in either case the maximum number of new balls that can enter a bin at any round is $R = 1$.

Let $m_1 = 1$ and $m_2 = \log q + 1$. Since $\neg\mathsf{Win}_3$, it is easy to see that any two distinct pairs of queries $(h_1(x'_1, y'_1), h_2(v'_1, z'_1))$, $(h_1(x'_2, y'_2), h_2(v'_2, z'_2))$ where $h_1(x'_i, y'_i) = v'_i$ and $h_2(v'_1, z'_1) =$

$w_1 \neq w_2 = h_2(v_2', z_2')$ uniquely determine one bin (x, y) such that $xw_i + y = x_i'w_i + y_i'$, $(x, y) \neq (x_i', y_i')$ for $i = 1, 2$, and conversely there exists such a pair of queries for any bin (x, y) with at least 2 distinct balls in it. Since there are at most $q^2 \log^2 q$ such pairs of queries (as for every query $h_2(v', z')$ there are most $\log q$ queries $h_1(x', y')$ such that $h_1(x', y') = v'$), there are at most $M = q^2 \log^2 q$ bins with at least $m_1 + 1 = 2$ balls in them. Thus the chance of winning the balls-in-bins game is at least

$$\frac{1}{q}\left(\frac{1}{qM}\right)^{\frac{1}{\log q}} = \frac{1}{q}\left(\frac{1}{q^3 \log^2 q}\right)^{\frac{1}{\log q}} = \frac{1}{8q(\log q)^{\frac{2}{\log q}}}.$$

Suppose that $\mathcal{B}$ correctly guesses the answer to an h_1-query $h_1(x', y')$ is going to result in a new ball being placed in a bin (x, y). Then there is at most one solution w to $xw + y = x'w + y'$ (by definition of the game $(x, y) \neq (x', y')$) and there are at most $\log q$ queries $h_2(v', z')$ such that $h_2(v', z') = w$, so $\mathcal{B}$ has chance at least $1/\log q$ of guessing the right value of v'.

If $\mathcal{B}$ correctly guesses that the answer to an h_2-query $h_2(v', z')$ is going to result in a new ball being placed in a bin (x, y), then there are most $\log q$ queries $h_1(x', y')$ such that $h_1(x', y') = v'$, and for each value of $(x', y') \neq (x, y)$ there is a unique solution w to $xw + y = x'w + y'$, so here again $\mathcal{B}$ has chance at least $1/\log q$ of guessing the correct value w. Thus $\mathcal{B}$'s overall chance of forging is at least

$$\frac{1}{8q \log q (\log q)^{\frac{2}{\log q}}}.$$

Ev_{13}: (Not a balls-in-bins game.) $\mathcal{B}$ randomly selects an index i between 1 and q. When $\mathcal{A}$ makes its i-th query $h_1(x, y)$, $\mathcal{B}$ selects uniformly at random among all the values w for which there exists a pair of distinct queries $(h_1(x', y') = v', h_2(v', z') = w)$ such that $xw + y = x'w + y$, selects uniformly at random among all values v' for which there exists a query $h_2(v', z') = w$, and guesses $h_1(x, y) = v'$.

Suppose that $\mathcal{A}$ makes a query $h_1(x, y)$ that triggers Win_{13}. Then because $\neg\mathsf{Win}_{12}$, there are at most $\log q + 1$ values w such that there exists a pair of queries $(h_1(x', y') = v', h_2(v', z') = w)$ for which $(x, y) \neq (x', y')$ and $xw + y = x'w + y'$. Moreover for each such w, there are at most $\log q$ queries $h_2(v', z')$ such that $h_2(v', z') = w$. Thus in total there are at most $\log q(\log q + 1)$ possibilities for v', so that $\mathcal{B}$ has chance at least $1/(\log q(\log q + 1))$ of forging if it guesses the right index i and overall chance of forging at least $1/(q \log q(\log q + 1))$.

To summarize, we have

$$\mathbf{Adv}_{\tau}^{\mathsf{wcr}}(t, q) \leq 26q \log q (\log q + 1)^2 \mathbf{Adv}_{h}^{\mathsf{mac}}\left(t + O(n^2 q^4 \log^4 q) + \mathsf{Time}_2, q\right),$$

where the factor $q \log q (\log q + 1)^2$ comes from events Ev_9 and Ev_{10}, and $t + O(n^2 q^4 \log^4 q) + \mathsf{Time}_2$ is the running time of the forger for the case Ev_6. The second part of the theorem is immediate from Lemma 1. □

By Theorem 9 and Lemma 5, we obtain the following theorem.

Theorem 10. *If $h_1, h_2, f_2 : \{0,1\}^{2n} \to \{0,1\}^n$ are random functions, then $\mathbf{Adv}_{\tau[h_1,h_2]}^{\mathsf{coll}}(q) \leq \epsilon(q)$ for*

$$\epsilon(q) = \frac{26q \log q (\log q + 1)^2}{2^n},$$

and $G[h_1, h_2, f_2] = f_2 \circ MD[\tau[h_1, h_2]]$ is $(\tilde{q}, \epsilon(\bar{q}))$-indistinguishable from a random function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$, where

$$\bar{q} = \mathsf{NQ}(MD[\tau[h_1, h_2]], l_{max})\tilde{q} = 2\left\lceil \frac{l_{max} + 1}{n} + 1 \right\rceil \tilde{q},$$

for the length in bits l_{max} of the longest query made by a distinguisher.

7 Improved Analysis of Lucks' Double-piped Mode of Operation

We begin with the definition of Lucks' double-piped mode of operation (with a slight modification). First, two $2n+c \to n$ bit functions f_1 and f_2 are concatenated, yielding the following compression function.

$$F : \{0,1\}^{2n+c} \longrightarrow \{0,1\}^{2n}$$
$$u \longmapsto f_1(u)||f_2(u).$$

The pictorial description of F for $c = n$ is shown in Figure 8. Given $F = F[f_1, f_2]$ and an independent compression function $f_3 : \{0,1\}^{2n+c} \to \{0,1\}^n$, the Lucks' mode of operation defines a hash function

$$G[f_1, f_2, f_3] : \{0,1\}^* \longrightarrow \{0,1\}^n$$
$$M \longmapsto f_3(0^c||v),$$

where $v = MD[F](M)$.

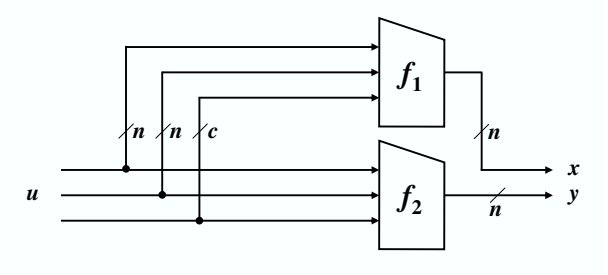


Fig. 8. Lucks' double-piped compression function

Now we prove the weak collision resistance of $F = F[f_1, f_2]$ where f_1 and f_2 are independently chosen from a function family f.

Theorem 11. *Let F be a function family defined by $f : \{0,1\}^\kappa \times \{0,1\}^{2n+c} \to \{0,1\}^n$ as described above. Then,*

$$\mathbf{Adv}^{\mathsf{wcr}}_F(t,q) \le \sqrt{2}q\mathbf{Adv}^{\mathsf{mac}}_f(t,q).$$

Proof. Let $\mathcal{A}$ be a weak collision-finding adversary such that

$$\mathbf{Adv}^{\mathsf{wcr}}_F(\mathcal{A}) = \mathbf{Adv}^{\mathsf{wcr}}_F(t,q) = \epsilon.$$

We write $u[i]$ for the i-query that $\mathcal{A}$ makes to F and $F(u[i]) = (x[i], y[i])$, where $x[i] = f_1(u[i])$ and $y[i] = f_2(u[i])$. Let

$$\Gamma = \left\{(j,i) \in \{1,\ldots,q\}^2 : j < i \text{ and } x[j] = x[i]\right\},$$

and let $\gamma = |\Gamma|$ be the number of "upper half-collisions". Here we fix an ordering in Γ. By assumption that $\mathcal{A}$ succeeds to find a collision with probability ϵ, at least one of the following two events happens with probability $\ge \epsilon/2$, where θ is a parameter to be optimized later.

Case 1: $\mathcal{A}$ finds a collision and $\gamma > \theta$. For this case, we can construct a forger $\mathcal{B}_1$ for f_1 as follows.

1. $\mathcal{B}_1$ chooses $(j,i) \in \{1,\ldots,q\}^2$ such that $j < i$ uniformly at random.
2. $\mathcal{B}_1$ runs $\mathcal{A}$ as a subroutine and faithfully answers the queries made by $\mathcal{A}$ until the $(i-1)$-th query. Here $\mathcal{B}_1$ simulates f_2 by choosing a key for f_2 uniformly at random.

3. On the i-query $u[i]$, $\mathcal{B}_1$ presents a forgery $(u[i], y[j])$ without making a query to $f_1(\cdot)$.

Since the probability that $(j,i) \in \Gamma$ is $\theta/\binom{q}{2}$, we have

$$\mathbf{Adv}_f^{\mathsf{mac}}(\mathcal{B}_1) \geq \frac{\epsilon\theta}{2\binom{q}{2}} \geq \frac{\epsilon\theta}{q^2}. \tag{11}$$

Case 2: $\mathcal{A}$ finds a collision and $\gamma \leq \theta$. For this case, we can construct a forger $\mathcal{B}_2$ for f_2 as follows.

1. $\mathcal{B}_2$ chooses $s \in \{1, \ldots, \theta\}$ uniformly at random.
2. $\mathcal{B}_2$ runs $\mathcal{A}$ as a subroutine: To each query made by $\mathcal{A}$, $\mathcal{B}_2$ faithfully respond by simulating f_1 and making queries to $f_2(\cdot)$.
3. $\mathcal{B}_2$ counts the number of collisions in f_1. At the s-th collision $(j,i) \in \Gamma$, $\mathcal{B}_2$ stops without making a query to $f_2(\cdot)$ and presents a forgery $(u[i], y[j])$.

If there exists a collision $(x[j], y[j]) = (x[i], y[i])$ for $j < i$, then obviously $(j,i) \in \Gamma$. Therefore we have

$$\mathbf{Adv}_f^{\mathsf{mac}}(\mathcal{B}_2) \geq \frac{\epsilon}{2\theta}. \tag{12}$$

From (11) and (12), it follows that

$$\mathbf{Adv}_F^{\mathsf{wcr}}(t,q) \leq \max\left\{\frac{q^2}{\theta}, 2\theta\right\} \mathbf{Adv}_f^{\mathsf{mac}}(t,q).$$

By setting $q^2/\theta = 2\theta$ or $\theta = q/\sqrt{2}$, we obtain

$$\mathbf{Adv}_F^{\mathsf{wcr}}(t,q) \leq \sqrt{2}q\mathbf{Adv}_f^{\mathsf{mac}}(t,q).$$

□

By Lemma 1 and Theorem 11, we obtain the following theorem.

Theorem 12. *Let $G = G[f_1, f_2, f_3]$ be a function family such that f_1, f_2 and f_3 are independently chosen from a function family f. Then for $q = \lfloor \mu/c \rfloor + 2\tilde{q}$,*

$$\mathbf{Adv}_G^{\mathsf{mac}}(t, \tilde{q}, \mu) \leq \mathbf{Adv}_f^{\mathsf{mac}}(t, \tilde{q}) + \sqrt{2}q\mathbf{Adv}_f^{\mathsf{mac}}(t,q).$$

With a slight modification of the above argument, we can prove that the Lucks' mode of operation *using a single key* also preserves unforgeability up to $O(2^n)$ query complexity, improving the bound $O(2^{5n/6})$ proved by Yasuda [24].

References

1. J. H. An and M. Bellare. Constructing VIL-MACs from FIL-MACs: Message authentication under weakened assumptions. Crypto 1999, LNCS 1666, pp. 252–269, Springer-Verlag, 1999.
2. M. Bellare and T. Ristenpart. Multi-property-preserving Hash Domain Extension and the EMD Transform. Asiacrypt 2006, LNCS 4284, pp. 299–314, Springer-Verlag, 2006.
3. J. Coron, Y. Dodis, C. Malinaud and P. Puniya. Merkle-Damgård revisited: How to construct a hash function. Crypto 2005, LNCS 3621, pp. 430–448, Springer-Verlag, 2005.
4. I. Damgård. A design principle for hash functions. Crypto 1989, LNCS 435, pp. 416–427, Springer-Verlag, 1990.
5. Y. Dodis, K. Pietrzak and P. Puniya. A new mode of operation for block ciphers and length-preserving MACs. Eurocrypt 2008, LNCS 4965, pp. 198–219, Springer-Verlag, 2008.
6. Y. Dodis, T. Ristenpart and T. Shrimpton. Salvaging Merkle-Damgård for practical applications. Eurocrypt 2009, LNCS 5479, pp. 371–388, Springer-Verlag, 2009.
7. Y. Dodis and J. Steinberger. Message authentication codes from unpredictable block ciphers. Crypto 2009, LNCS 5677, pp. 267–285, Springer-Verlag, 2009.

8. J. von zur Gathen and D. Panario. Factoring polynomials over finite fields: A survey. J. Symbolic computation, Vol. 31, pp. 3–17, 2001.
9. J. von zur Gathen and V. Shoup. Computing Frobenius maps and factoring polynomials. Computational complexity, Vol. 2, pp. 187–224, 1992.
10. S. Hirose. Some plausible constructions of double length hash functions. FSE 2006, LNCS 4047, pp. 210–225, Springer-Verlag, 2006.
11. A. Joux. Multicollisions in iterated hash functions. Application to cascaded constructions. Crypto 2004, LNCS 3152, pp. 306–316, Springer-Verlag, 2004.
12. J. Kelsey and T. Kohno. Herding hash functions and the Nostradmus attack. Eurocrypt 2006 LNCS 4004, pp. 183–200, Springer-Verlag, 2006.
13. J. Kelsey and B. Schneier. Second preimages on n-bit hash functions for much less than 2^n work. Eurocrypt 2005 LNCS 3494, pp. 474–490, Springer-Verlag, 2005.
14. X. Lai and J. Massey. Hash function based on block ciphers. Eurocrypt 1992, LNCS vol. 658, pp. 55–70, Springer-Verlag, 1992.
15. S. Lucks. A failure-freindly design principle for hash functions. Asiacrypt 2005, LNCS 3788, pp. 474–494, Springer-Verlag, 2005.
16. S. Lucks. A collision-resistant rate-1 double-block-length hash function. Symmetric Cryptography, Dagstuhl Seminar Proceedings 07021, 2007.
17. R. Merkle. One way hash functions and DES. Crypto 1989, LNCS 435, pp. 428–446, Springer-Verlag, 1990.
18. U. Maurer, R. Renner and R. Holenstein. Indifferentiability, impossibility results on reductions, and apllications to the random oracle methodology. TCC 2004, LNCS 2951, pp. 21–39, Springer-Verlag, 2008.
19. U. Maurer and J. Sjödin. Single-key AIL-MACs from any FIL-MAC. ICALP 2005, LNCS 3580, pp. 472–484, Springer-Verlag, 2005.
20. O. Özen and M. Stam. Another glance at double length hashing, preprint, 2009.
21. P. Rogaway and J. Steinberger. Constructing cryptographic hash functions from fixed-key blockciphers. Crypto 2008, LNCS 5157, pp. 433–450, Springer-Verlag, 2008.
22. M. Stam. Beyond uniformity: Security/efficiency tradeoffs for compression functions. Crypto 2008, LNCS 5157, pp. 397–412, Springer-Verlag, 2008.
23. M. Stam. Blockcipher based hashing revisited. FSE 2009, LNCS, Springer-Verlag, 2009. To appear.
24. K. Yasuda. A double-piped mode of operation for MACs, PRFs and PROs: Security beyond the birthday barrier. Eurocrypt 2009, LNCS 5479, pp. 242–259, Springer-Verlag, 2009.