Domain Expansion of MACs: Alternative Uses of the FIL-MAC[⋆]

Ueli Maurer and Johan Sjödin

Department of Computer Science,
Swiss Federal Institute of Technology (ETH), Zurich,
CH-8092 Zurich, Switzerland
{maurer, sjoedin}@inf.ethz.ch

Abstract. In this paper, a study of a paradigm for domain expansion of MACs is generalized. In particular, a tradeoff between the efficiency of a MAC and the tightness of its security reduction is investigated in detail. Our new on-line single-key AIL-MAC construction, the PDI-construction, transforms any FIL-MAC into an AIL-MAC and is superior to all previous AIL-MAC constructions given in the literature (taking the tradeoff into account). It appears obvious that this construction is essentially optimal.

Keywords: Message authentication code (MAC), arbitrary-input-length (AIL), variable-input-length (VIL), fixed-input-length (FIL).

1 Introduction

1.1 Motivation: Data Integrity

A message authentication code (MAC) is a function family

$$H := \{h_k : \mathcal{M} \to \mathcal{T}\}_{k \in \mathcal{K}},$$

where $\mathcal{M}$ is the message space, $\mathcal{T}$ the tag space, and $\mathcal{K}$ the key space. It is the most commonly used method for assuring the integrity of data communicated between two parties sharing a secret key k. A party authenticates a message m by computing a tag $\tau = h_k(m)$ which is sent along with m to the other party. A party receiving (m', τ') accepts the message m' if (m', τ') is valid, i.e., satisfies $\tau' = h_k(m')$. Of course, it should be infeasible for a party not in possession of k to be able to generate a valid message-tag pair (which is new), since this would contradict data integrity. The function h_k is referred to as an *instantiation* of the MAC H.

[⋆] This work was partially supported by the Zurich Information Security Center. It represents the views of the authors.

1.2 Domain Expansion of MACs

Cryptographic primitives can be classified according to their domain. We refer to a primitive with domain:

- $\{0,1\}^L$, i.e., the set of all bitstrings of length L, as a *fixed-input-length* (FIL) primitive.
- $\{0,1\}^*$, i.e., the set of all bitstrings of finite length, as a *arbitrary-input-length* (AIL) primitive.
- $\{0,1\}^{\leq N}$, i.e., the set of all bitstrings of length at most N, as a *variable-input-length* primitive.

VIL- and AIL-primitives are often constructed by iterating applications of some FIL-primitive.

In the context of constructing VIL- or AIL-MACs, a natural and weak assumption on the FIL-primitive is that of being a MAC. This was first studied by An and Bellare in [1], who proposed and proved the security of the NI-construction, the first VIL-MAC based on a FIL-MAC. Domain expansion of MACs was further studied in [8], where a general paradigm for constructing VIL- and AIL-MACs by iterating applications of a FIL-MAC was proposed. Several improvements on the NI-construction and two single-key AIL-MAC constructions, Chain-Shift (CS) and Chain-Rotate (CR), were presented. While the CS-construction transforms FIL-MACs with input-length/output-length ratio at least 2, the CR-construction transforms any FIL-MAC (irrespectively of its input-length/output-length ratio) at the cost of a less tight security reduction (by a factor of roughly 5). In this paper the paradigm is generalized and analyzed further.

Domain expansion is well studied for many cryptographic primitives such as collision resistant hash function [6,10], pseudo-random functions (PRFs) [2,3,11,7], universal one-way hash functions [4,12], and random oracles [5]. Since a PRF is (trivially) also a MAC, many VIL- and AIL-PRFs based on a FIL-PRF are widely used as VIL- and AIL-MACs, respectively. However, these MACs are only guaranteed to be secure under the (relative strong) assumption that the FIL-primitive is a PRF. For instance the CBC-MAC [3] is not secure under the assumption that the FIL-primitive is a secure MAC [1]. In cryptography a central goal is to prove the security of cryptographic schemes under as weak assumptions as possible. Demanding that the FIL-primitive is a MAC (rather than a PRF) is a more cautious cryptographic assumption.

1.3 The Construction Paradigm

Let us briefly recall the construction paradigm of [8] as a reference for our contributions. Throughout this paper, the function family

$$G := \{g_k : \{0,1\}^L \to \{0,1\}^\ell\}_{k\in\{0,1\}^\kappa}$$

(with $L > \ell$) denotes a FIL-MAC with *compression parameter*

$$b := L - \ell.$$

The paradigm considers a type of construction $C^{\cdot}$, which uses G to construct an AIL-MAC[1]

$$C^G := \{C^{g_k} : \{0,1\}^* \to \{0,1\}^\ell\}_{k \in \{0,1\}^\kappa}.^2$$

More precisely, the computation of the tag $\tau = \mathrm{C}^{g_k}(m)$ for an n-bit message m can be described as follows. In a pre-processing step m is encoded into a bit string m', for instance by padding m and appending information about its length. The processing step is best described with a buffer initialized with m', where each call to g_k fetches (and deletes) some L bits and writes back the ℓ-bit result to the buffer (for instance by concatenating it at the end of the buffer). This reduces the number of bits in the buffer by b with each call to g_k. The output of the last (possible) call to g_k is returned as the tag (instead of being written back to the buffer). The length of m' is appropriately chosen to be $t(n) \cdot b + \ell$ for some $t(n) \in \mathbb{N}$, to leave the buffer empty after the computation. We stress that $t(n)$ is exactly the number of calls to g_k before the tag is returned. The function $t(\cdot)$ is referred to as the *application* function of $\mathrm{C}^{\cdot}$. A particular construction can thus be described by the encoding function mapping m to m' and by the scheme by which the L-bit blocks are fetched. The computation process is illustrated in Fig. 1.

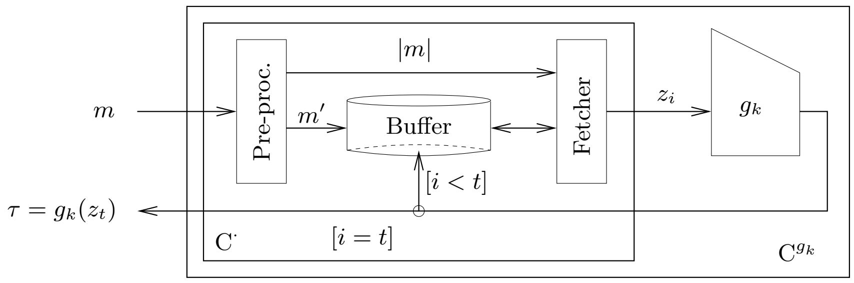

Fig. 1. The construction paradigm

The efficiency of a construction is measured in the number of applications $t(n)$ of the FIL-MAC, or, equivalently, in terms of the *waste*

$$w(n) := t(n) \cdot b + \ell - n,$$

i.e., the amount by which pre-processing expands the message.

[1] We consider single-key AIL-MAC constructions in this paper, i.e., AIL-MAC constructions which use one instantiation of the FIL-MAC.

[2] $C^{\cdot}$ denotes the construction, where the dot is the placeholder for G. Actually, $C^{\cdot}$ transforms any FIL-MAC with any compression parameter b and output length ℓ (if not stated otherwise). We describe $C^{\cdot}$ for arbitrary but fixed values of b and ℓ, and let these parameters be implicitly given (if not stated otherwise).

1.4 Our Contribution

In this paper, we generalize the construction paradigm of [8]. The main idea is to comprise constructions which first transform the FIL-MAC G into a FIL-MAC

$$[G]_f := \{[g_k]_f\}_{k\in\{0,1\}^\kappa}$$

defined by

$$[g_k]_f(x) := f(g_k(x)),$$

where $f : \{0,1\}^\ell \to \{0,1\}^{\ell-\delta}$ (with $\delta > 0$) is a key-less compression function, and then the FIL-MAC $[G]_f$ (rather than G) into an AIL-MAC $C^{[G]_f}$ (by some AIL-MAC construction $C^{\cdot}$). The new type of construction $C^{[\cdot]_f}$ is more efficient than $C^{\cdot}$, since $[G]_f$ compresses more than G. However, this improvement is at the cost of a worse security reduction. For example if the function f cuts away the δ most significant bits of its input, the security reduction is worsened by a factor of roughly 2^δ.

This tradeoff is investigated in detail. At first sight, a less tight security reduction by some constant factor (as for the CR-construction [8]) seems irrelevant. However, by allowing such a factor, the throughput of other constructions can be improved substantially and result in overall better constructions (see Sect. 4.3).

In this paper, we also propose a new on-line[3] AIL-MAC construction, the PDI-construction, which is superior to all AIL-MAC constructions given in the literature, taking the security/efficiency tradeoff into account.

2 Preliminaries

2.1 Notation

If M is a set, $\#M$ denotes its cardinality. For a sequence S of elements, $|S|$ denotes its length and S_i the sequence of its first $i \le |S|$ elements. For any $n \in \mathbb{N}_0$, let $[n] := \{1,\ldots,n\}$ (with $[0] := \emptyset$).

For x, $y \in \{0,1\}^*$, let $|x|$ denote the length of x (in bits), $x\|y$ the concatenation of x and y, $\langle n\rangle_b$ a b-bit encoding of a positive integer $n \le 2^b$, $x[i]$ the i^{th} bit of x, and

$$x[i,j] := x[i]\big\|x[i+1]\big\|\cdots\big\|x[j]$$

for $1 \le i < j \le |x|$. Furthermore, let $\mathrm{RR}(\cdot)$ denote the operator on bit strings that rotates the input by one position to the right, i.e.,

$$\mathrm{RR}(x) := x[L]\|x[1, L-1].$$

An encoding $\sigma : \{0,1\}^* \to \{0,1\}^*$ is called *prefix-free* if there are no three strings x, x', $y \in \{0,1\}^*$ such that $x \ne x'$ and $\sigma(x)\|y = \sigma(x')$. A *non-trivial* collision for a function f is a pair $x \ne x'$ of inputs for which $f(x) = f(x')$.

If $\mathcal{E}$ denotes an event, $\bar{\mathcal{E}}$ denotes the complementary event.

[3] i.e., the ability to process a message as the message bits arrive, without knowing the message length in advance.

2.2 Security Definition for MACs

A forger F for a MAC $H := \{h_k : \mathcal{M} \to \mathcal{T}\}_{k\in\mathcal{K}}$ has oracle access to $h_k(\cdot)$ (for which k is chosen uniformly at random from $\mathcal{K}$ and kept secret) and can thus learn the tag values for some adaptively chosen messages $m_1, \ldots, m_q$. It then returns a *forgery* (m, τ), i.e., a message m together with a tag τ. The forger F is considered successful if $h_k(m) = \tau$. The only constraint on m is that it must be *new*, i.e., different from all previous messages $m_1, \ldots, m_q$. A forger F is referred to as a (t, q, μ, ε)-forger, if t, q, and μ are upper bounds on the running time, the number of messages (or oracle queries), and the total length (in bits) of the oracle queries including the forgery message m, respectively, and ε is a lower bound on the success probability. Informally, a MAC is considered secure against *existential forgery* under an *adaptive chosen-message attack*, if there is no (t, q, μ, ε)-forger, even for very high values of t, q, and μ, and a very small value of ε.

Definition 1. *A MAC is* (t, q, μ, ε)-secure *if there exists no* (t, q, μ, ε)*-forger.*

A forger for a FIL-MAC will be denoted simply as a (t, q, ε)-forger, since the parameter μ is determined by q and the input-length L, i.e., $\mu = (q + 1) \cdot L$.

To prove the security of a MAC, based on a FIL-MAC, one shows that the existence of a (t, q, μ, ε)-forger F for the MAC implies the existence of a (t', q', ε')-forger F' for the FIL-MAC, where t', q', and ε' are functions of t, q, μ, and ε. In all our security proofs F is called only once by F'. Therefore, the running time of F' is essentially that of F, i.e., $t' \approx t$, with some small overhead that is obvious from the construction of F'. We will therefore not bother to explicitly compute the running time of forgers, as this complicates the analysis unnecessarily without providing more insight. Therefore we drop the time parameter t in the sequel.

2.3 Security Reductions

We make use of the proof technique of [8], which we recall for completeness. Let F be a (q, μ, ε)-forger for a MAC C^G and let

$$F \circ \mathrm{C}^{g_k}$$

denote the process in which F's queries to (its oracle) C^{g_k} are computed and returned to F, and where F's forgery (m, τ) is verified by computing $\mathrm{C}^{g_k}(m)$. Consider the random variables occurring at the interface to g_k (in the process $F \circ \mathrm{C}^{g_k}$), and let z_i denote the i^{th} input to g_k and $y_i := g_k(z_i)$ the corresponding output. The sequences

$$\mathbf{Z} := (z_1, z_2, \ldots) \quad \text{and} \quad \mathbf{Y} := (y_1, y_2, \ldots)$$

are thus naturally defined. Note that as soon as the key k and the random coins of F are fixed, all values in $\mathbf{Z}$ and $\mathbf{Y}$ are determined, and also whether F is successful or not. Let $\mathcal{E}$ denote the event that F is successful. Without loss of generality we assume that F's forgery message m is distinct from F's oracle queries. Thus $\mathcal{E}$ occurs if and only if $\mathrm{C}^{g_k}(m) = \tau$.

A forger F' for the FIL-MAC G simulates $F \circ \mathrm{C}^{g_k}$ with the help of F and its oracle access to g_k. At some query z_i to g_k it stops the simulation and returns a forgery (z', τ') for g_k (without making any other oracle queries to g_k). Such a forger is characterized by the time when it stops (i.e., i) and the way it produces its forgery. This is referred to as a *strategy* s of F' and F'_s denotes the corresponding forger.

The most simple strategy is the *naïve* strategy s_{na}. $F'_{s_{\mathrm{na}}}$ stops the simulation of $F \circ \mathrm{C}^{g_k}$ at the very last query $\mathbf{z}$ to g_k (i.e., $\mathbf{z}$ is the last entry in $\mathbf{Z}$). Then it returns $(\mathbf{z}, \tau)$ as a forgery, where τ is the forgery tag of F's forgery (m, τ) for C^{g_k}. $F'_{s_{\mathrm{na}}}$ is successful if the following two conditions hold. First, $\mathcal{E}$ occurs, i.e., $\mathrm{C}^{g_k}(m) = \tau$ (and thus $g_k(\mathbf{z}) = \tau$ by definition of $\mathrm{C}^{\cdot}$), and second $\mathbf{z}$ is new, i.e., $\mathbf{z}$ is only the last entry in $\mathbf{Z}$. Let $\mathcal{E}_{\mathrm{new}}$ denote the event that $\mathbf{z}$ is new. Thus $F'_{s_{\mathrm{na}}}$ is successful whenever $\mathcal{E} \wedge \mathcal{E}_{\mathrm{new}}$ occurs.

Assume there is a set $\mathcal{S}$ of strategies for a construction with the property that, whenever $\bar{\mathcal{E}}_{\mathrm{new}}$ occurs, there exists at least one strategy $s \in \mathcal{S}$ for which F'_s is successful. Such a set is referred to as *complete* for the construction. Obviously, the set $\mathcal{S} \cup \{s_{\mathrm{na}}\}$ has the property that whenever $\mathcal{E}$ occurs, there is at least one strategy $s \in \mathcal{S} \cup \{s_{\mathrm{na}}\}$ for which F'_s is successful. Thus an overall strategy of F' is to pick its strategy uniformly at random from $\mathcal{S} \cup \{s_{\mathrm{na}}\}$. Its success probability is at least the probability that $\mathcal{E}$ occurs, divided by $\#\mathcal{S} + 1$. As F''s number of oracle queries is $|\mathbf{Z}|$, which is a random variable, it is convenient to introduce the following function.

Definition 2. [8] *The* expansion *function* $e : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *of a construction* $\mathrm{C}^{\cdot}$ *is defined as*

$$e(\tilde{q}, \tilde{\mu}) := \max \left\{ \sum_{i=1}^{\tilde{q}} t(n_i) : n_1, \ldots, n_{\tilde{q}} \in \mathbb{N}_0,\ n_1 + \cdots + n_{\tilde{q}} \leq \tilde{\mu} \right\},$$

where $t(\cdot)$ *is the application function of* $\mathrm{C}^{\cdot}$.

It follows that $|\mathbf{Z}| \leq e(q+1, \mu)$, since there are at most $q+1$ queries of total length at most μ to C^{g_k} in $F \circ \mathrm{C}^{g_k}$. In general, $\#\mathcal{S}$ is a function of $e(q+1, \mu)$.

Proposition 1. [8] *The existence of a complete set* $\mathcal{S}$ *for a construction* $\mathrm{C}^{\cdot}$ *and a* (q, μ, ε)*-forger* F *for* C^G *implies the existence of a* (q', ε')*-forger* F' *for* G*, where* $q' = e(q+1, \mu)$ *and* $\varepsilon' = \frac{\varepsilon}{\#\mathcal{S}+1}$.

An important class of strategies for F' are the deterministic strategies. A deterministic strategy s is characterized by a pair (i, f), where $i \in [e(q+1, \mu)]$ is an index and f a function mapping $(\mathbf{Z}_i, \mathbf{Y}_{i-1})$ to some value $\hat{y}_i \in \{0,1\}^{\ell}$ (which can be seen as a prediction of y_i). To be more precise, the corresponding forger F'_s stops (the simulation of $F \circ \mathrm{C}^{g_k}$) at query z_i and returns $(z_i, \hat{y}_i)$ as a forgery.[4] The forger is successful if $\hat{y}_i = y_i$ and if z_i is new, i.e., not contained in

[4] If $i > |\mathbf{Z}|$ the forger aborts.

the sequence $\mathbf{Z}_{i-1}$. In the sequel, we will make use of the following two sets of deterministic forgers (from [8]):

– Let $s_{i,y}$ (for $y \in \{0,1\}^\ell$) denote the strategy of stopping at query z_i and returning (z_i, y) as a forgery. Note that whenever the event occurs that an output of g_k is equal to y, i.e., y is an entry in $\mathbf{Y}$, then there exists a strategy $s \in \mathcal{S}_y := \{s_{i,y} | i \in [e(q+1,\mu)]\}$ for which F'_s is successful. We have

$$\#\mathcal{S}_y = e(q+1,\mu). \tag{1}$$

– Let $s_{\text{coll},i,j}$ (for $i > j$) denote the strategy of stopping at query z_i and returning (z_i, y_j) as a forgery. Note that whenever a non-trivial collision for g_k occurs, i.e., $\alpha, \beta \in [|\mathbf{Z}|]$ satisfying $z_\alpha \neq z_\beta$ and $y_\alpha = y_\beta$, then there is a strategy $s \in \mathcal{S}_{\text{coll}} := \{s_{\text{coll},i,j} | i,j \in [e(q+1,\mu)], i > j)\}$ for which F'_s is successful. The cardinality of $\mathcal{S}_{\text{coll}}$ is

$$\#\mathcal{S}_{\text{coll}} = e(q+1,\mu)^2/2 - e(q+1,\mu)/2. \tag{2}$$

3 Concrete AIL-MAC Constructions

In this section we present new on-line AIL-MAC constructions. First, we introduce the Double-Iterated (DI) construction which has constant waste (i.e., $w(n) \in \theta(1)$) and therefore is efficient for long messages. Then, we present the Prefix-Free Iterated (PI) construction which has linear waste (i.e., $w(n) \in \theta(n)$) but is more efficient than the DI-construction for short messages.

Finally, we propose the Prefix-Free Double Iterated (PDI) construction, which depends on some design parameter $r \in \mathbb{N}_0$ and is a hybrid constructions between the DI- and the PI-construction. For $r = 0$ the construction is equivalent to the DI-construction and for $r \to \infty$ to the PI-construction. For values of r between this range the advantages of both the DI- and the PI-construction are exploited. The idea is to simply apply the PI-construction for short messages and the DI-construction for long messages. What short and long means depends on the value of r.

3.1 The Iteration (I) Method

Before the AIL-MAC constructions are presented, we analyze the iteration $\mathrm{I}^h_{\mathrm{IV}}(\cdot)$ of a function $h : \{0,1\}^{b+\ell} \to \{0,1\}^\ell$, where IV denotes a fixed ℓ-bit initialization value. It is defined as follows and illustrated in Fig. 2 (see Sect. 9.3.1 of [9]).

The value $\tau = \mathrm{I}^h_{\mathrm{IV}}(m)$ for a string $m \in (\{0,1\}^b)^*$, i.e., $m_1 \| \cdots \| m_t = m$ for some $t \in \mathbb{N}_0$ and $|m_i| = b$ for $i \in [t]$, is computed as

$$y_0 = \mathrm{IV}; \quad y_i = h(y_{i-1} \| m_i)\ ,\ 1 \le i \le t; \quad \tau = y_t.$$

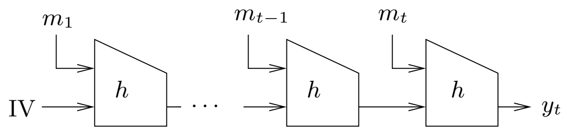

Fig. 2. The iteration (I) method

Lemma 1. *A non-trivial collision in* $\mathrm{I}^h_{\mathrm{IV}}(\cdot)$ *implies a non-trivial collision in* h *or that an output of* h *is equal to* IV*.*

Proof. Let $m \neq m'$ and $\mathrm{I}^h_{\mathrm{IV}}(m) = \mathrm{I}^h_{\mathrm{IV}}(m')$ denote a non-trivial collision in $\mathrm{I}^h_{\mathrm{IV}}(\cdot)$. Furthermore, let $(z_1, \ldots, z_t)$ and $(z'_1, \ldots, z'_{t'})$ denote the sequence of inputs to h in the computation of $\mathrm{I}^h_{\mathrm{IV}}(m)$ and $\mathrm{I}^h_{\mathrm{IV}}(m')$, respectively. Note that $h(z_t) = \mathrm{I}^h_{\mathrm{IV}}(m) = \mathrm{I}^h_{\mathrm{IV}}(m') = h(z'_{t'})$.

Let i denote the smallest index (if any) for which $z_{t-i} \neq z'_{t'-i}$ and $h(z_{t-i}) = h(z'_{t'-i})$. The existence of i directly implies a non-trivial collision in $h(\cdot)$. The non-existence of such an index i implies that one of the sequences $(z_1, \ldots, z_t)$ and $(z'_1, \ldots, z'_{t'})$ is a suffix of the other with $t \neq t'$ since $m \neq m'$. Assume without loss of generality that $t < t'$. In this case we have $\mathrm{IV}\|v = z_1 = z'_{t'-t+1} = h(z_{t'-t})\|v$ for some $v \in \{0,1\}^b$, which means that an output of h is equal to IV. □

Lemma 2. $\mathrm{I}^h_{\mathrm{IV}}(m) = \mathrm{I}^h_{\mathrm{IV}'}(m')$ *with* $m, m' \in (\{0,1\}^b)^*$ *and* $\mathrm{IV} \neq \mathrm{IV}'$ *imply a non-trivial collision in* h*, or that an output of* h *is equal to* IV *or* IV$'$*.*

Proof. Let $(z_1, \ldots, z_t)$ and $(z'_1, \ldots, z'_{t'})$ denote the sequence of inputs to h in the computation of $\mathrm{I}^h_{\mathrm{IV}}(m)$ and $\mathrm{I}^h_{\mathrm{IV}'}(m')$, respectively. Note that $h(z_t) = \mathrm{I}^h_{\mathrm{IV}}(m) = \mathrm{I}^h_{\mathrm{IV}'}(m') = h(z'_{t'})$.

Let i denote the smallest index (if any) for which $z_{t-i} \neq z'_{t'-i}$ and $h(z_{t-i}) = h(z'_{t'-i})$. The existence of i directly implies a non-trivial collision in $h(\cdot)$. The non-existence of such an index i implies that one of the sequences $(z_1, \ldots, z_t)$ and $(z'_1, \ldots, z'_{t'})$ is a suffix of the other with $t \neq t'$ since $\mathrm{IV} \neq \mathrm{IV}'$. If $t < t'$ we have $\mathrm{IV}\|v = z_1 = z'_{t'-t+1} = h(z_{t'-t})\|v$ for some $v \in \{0,1\}^b$, which means that an output of h is equal to IV. Analogously, one shows that if $t > t'$ an output of h is equal to IV$'$. □

Remark 1. The Merkle-Damgård (MD) iteration method [6,10] for collision-resistant hashing is a result of similar nature. The hash value $\mathrm{MD}^h_{\mathrm{IV}}(m)$, where $m \in \{0,1\}^{\leq 2^b}$ (and $\mathrm{IV} \in \{0,1\}^{\ell}$), is defined by first breaking m into sequence of b-bit blocks $m_1, \ldots, m_t$ (where m_t is padded with zeroes if necessary) and then returning the value $\mathrm{I}^h_{\mathrm{IV}}(m_1\| \cdots \|m_t\|\langle|m|\rangle_b)$. A non-trivial collision in $\mathrm{MD}^h_{\mathrm{IV}}(\cdot)$ implies a non-trivial collision in $h(\cdot)$.

3.2 The DI-Construction

The DI-construction is a generalization of the CS-construction [8], which transforms any FIL-MAC (irrespectively of its input-length/output-length ratio) to an AIL-MAC.[5] To be more precise, $\mathrm{DI}^{\cdot}$ uses any FIL-MAC G to construct an AIL-MAC $\mathrm{DI}^G := \{\mathrm{DI}^{g_k} : \{0,1\}^* \to \{0,1\}^\ell\}_{k\in\{0,1\}^\kappa}$ as follows.

Break the message $m \in \{0,1\}^*$ (of length n) into a sequence of b-bit blocks $m_1, \ldots, m_{t-1}$ (if $t > 1$) and a $(\lceil \ell/b \rceil b - \ell)$-bit block m_t, where a 1 followed by 0's is used as padding, i.e., $m_1\|\cdots\|m_t = m\|10^\nu$ for some $\nu \in \{0,\ldots,b-1\}$. Let

$$\mathrm{DI}^{g_k}(m) := \begin{cases} \mathrm{I}^{g_k}_{1^\ell}\left(\mathrm{I}^{g_k}_{0^\ell}(m_1\|\cdots\|m_{t-1})\,\|m_t\right) & \text{if } t > 1 \\ \mathrm{I}^{g_k}_{1^\ell}(0^\ell\|m_1) & \text{otherwise} \end{cases}.$$

The application function is $t(n) = \left\lceil \frac{n+1+\ell}{b} \right\rceil$ (resulting in the waste $w(n) \in \Theta(1)$).

Theorem 1. *A (q, μ, ε)-forger F for DI^G implies a (q', ε')-forger F' for G, where $q' = \frac{\mu}{b} + \frac{b+\ell}{b} \cdot (q+1)$ and $\varepsilon' = \frac{\varepsilon}{\frac{1}{2}q'^2 + \frac{3}{2}q' + 1}$.*

Proof. We show that $\mathcal{S} := \mathcal{S}_{\mathrm{coll}} \cup \mathcal{S}_{0^\ell} \cup \mathcal{S}_{1^\ell}$ is complete for $\mathrm{DI}^{\cdot}$ by proving that, whenever the last input $\mathbf{z}$ to g_k is not new, there is a non-trivial collision in g_k or an output of g_k that is equal to 0^ℓ or 1^ℓ.

Assume that $\mathbf{z}$ is not new. Furthermore, assume that there is no non-trivial collision in g_k and no output of g_k that is equal to 0^ℓ or 1^ℓ. We show that this leads to a contradiction. By Lemma 1, there can not be a non-trivial collision in $\mathrm{I}^{g_k}_{0^\ell}(\cdot)$. Furthermore, no output of $\mathrm{I}^{g_k}_{0^\ell}(\cdot)$ is equal to 0^ℓ, since this would directly imply a non-trivial collision in g_k. As a consequence, the last input $\tilde{m}$ to $\mathrm{I}^{g_k}_{1^\ell}(\cdot)$ is distinct from the other inputs to $\mathrm{I}^{g_k}_{1^\ell}(\cdot)$.[6] Since $\mathbf{z}$ is not new, $\mathbf{z}$ must have been an earlier query to g_k, resulting from some query $m' = m'_1\|\cdots\|m'_{t'}$ to $\mathrm{I}^{g_k}_{\mathrm{IV}}(\cdot)$ with $\mathrm{IV} \in \{0^\ell, 1^\ell\}$. Let $z'_1, \ldots, z'_{t'}$ denote the sequence of queries to g_k in the computation of $\mathrm{I}^{g_k}_{\mathrm{IV}}(m')$ and let s be the index for which $z'_s = \mathbf{z}$. Thus, we have $\mathrm{I}^{g_k}_{\mathrm{IV}}(m'_1\|\cdots\|m'_s) = \mathrm{I}^{g_k}_{1^\ell}(\tilde{m})$. We distinguish two cases:

- If $\mathrm{IV} = 0^\ell$, we arrive at a contradiction by Lemma 2.
- If $\mathrm{IV} = 1^\ell$, it follows from the construction that $|m'| = |\tilde{m}|$. Thus, we have $m'_1\|\cdots\|m'_s \neq \tilde{m}$, since $\tilde{m}$ is distinct (from the other queries to $\mathrm{I}^{g_k}_{1^\ell}(\cdot)$). As a consequence, we arrive at a contradiction by Lemma 1.

By definition of $e(q+1, \mu)$, there exist $n_1, \ldots, n_{q+1} \in \mathbb{N}_0$ such that:

$$e(q+1,\mu) = \sum_{i=1}^{q+1} t(n_i) = \sum_{i=1}^{q+1} \left\lceil \frac{n_i + 1 + \ell}{b} \right\rceil \leq \frac{\mu + (b+\ell)(q+1)}{b} =: q'.$$

Thus $\#\mathcal{S} + 1 \leq q'^2/2 + 3q'/2 + 1$ by (1) and (2). Proposition 1 concludes the proof. □

[5] The constructions coincide for $b \geq \ell$.

[6] Recall that, with out loss of generality, we assume that the forgery message m of F is distinct from its oracle queries.

Remark 2. The method, used (in [8]) for improving the efficiency of the CS-construction for short messages, can directly be applied for the DI-construction as well. This results in a more efficient construction, which is unfortunately not (completely) on-line.

The DI-construction can also be parallelized in the same way as the CS-construction (see [8]).

3.3 The PI-Construction

The PI-construction uses a prefix-free encoding $\sigma : \{0,1\}^* \to (\{0,1\}^b)^*$, to be defined later, for transforming G into the AIL-MAC $\mathrm{PI}^G := \{\mathrm{PI}^{g_k} : \{0,1\}^* \to \{0,1\}^\ell\}_{k\in\{0,1\}^\kappa}$. It is defined as follows.

For a message $m \in \{0,1\}^*$, let

$$\mathrm{PI}^{g_k}(m) := \mathrm{I}_{0^\ell}^{g_k}(\sigma(m)).$$

Theorem 2. *A (q,μ,ε)-forger for PI^G (with a prefix-free encoding σ) implies a (q',ε')-forger for G, where $q' = e(q+1,\mu)$ and $\varepsilon' = \frac{\varepsilon}{\frac{1}{2}q'^2+\frac{1}{2}q'+1}$. The expansion function e depends on the concrete choice of σ.*

Proof. We apply Proposition 1 and show that $\mathcal{S} := \mathcal{S}_{\mathrm{coll}} \cup \mathcal{S}_{0^\ell}$ is complete for $\mathrm{PI}^\cdot$ by showing that if $\mathbf{z}$ is not new, then there is a non-trivial collision in g_k or a 0^ℓ-output of g_k. This follows directly from Lemma 1 and the fact that an old $\mathbf{z}$ implies a non-trivial collision in $\mathrm{I}_{0^\ell}^{g_k}(\cdot)$ (due to the prefix-free encoding). □

The on-line property and the efficiency of the construction (hence also the expansion function e) depend on which prefix-free encoding σ is used. It seems obvious that there is no prefix-free encoding for which the construction is on-line and has waste $w(n) \in O(\log(n))$.[7] However, allowing linear waste, i.e., $w(n) \in \theta(n)$, there are prefix-free encodings for which the construction has the on-line property. Throughout this paper, we define σ as follows.

Let

$$\sigma(m) := 0\|m_1\|0\|m_2\|\cdots\|0\|m_{t-1}\|1\|m_t,$$

where $m \in \{0,1\}^*$ and $m_1,\ldots,m_t$ are $(b-1)$-bit blocks such that $m_1\|\cdots\|m_t = m\|10^\nu$ with $\nu \in \{0,\ldots,b-2\}$.

The application function of the PI-construction, with prefix-free encoding σ (as just defined), is $t(n) = \lceil (n+1)/(b-1) \rceil$. This results in waste $w(n) \in \theta(n)$. However, note that the PI-construction is more efficient than the DI-construction if (and only if) the message length is shorter than $\ell(b-1)$.

The following Corollary follows.

[7] The prefix-free encoding, described next, has logarithmic waste but is not on-line. Let $\sigma : \{0,1\}^* \to (\{0,1\}^b)^*$ be defined by $r = |\langle|m|\rangle| - 1$ and $\rho(m) := 0^r1\|\langle|m|\rangle\|m\|0^\nu$, where $\nu \in \{0,\ldots,b-1\}$ is chosen such that the length is a multiple of b.

Corollary 1. *A (q,μ,ε)-forger for* PI^G *(with σ defined as above) implies a (q',ε')-forger for G, where $q' = \frac{\mu}{b-1} + (q+1)$ and $\varepsilon' = \frac{\varepsilon}{\frac{1}{2}q'^2+\frac{1}{2}q'+1}$.*

Proof. The proof follows directly from Theorem 2 and the fact that there exist $n_1,\ldots,n_{q+1} \in \mathbb{N}_0$ such that

$$\begin{aligned} e(q+1,\mu) &= \sum_{i=1}^{q+1} t(n_i) = \sum_{i=1}^{q+1} \left\lceil \frac{n_i+1}{b-1} \right\rceil \leq \sum_{i=1}^{q+1} \frac{n_i+b-1}{b-1} \\ &\leq \frac{\mu}{b-1} + (q+1) =: q'. \end{aligned}$$

As a consequence $\#\mathcal{S} + 1 \leq q'^2/2 + q'/2 + 1$ by (1) and (2). □

3.4 The PDI-Construction

The PDI-construction is an AIL-MAC construction, which is a hybrid construction between the PI- and DI-construction. It exploits the advantages of both constructions as follows.

Let $r \in \mathbb{N}_0$ be a design parameter. The construction $\mathrm{PDI}_r^{\cdot}$ transforms any FIL-MAC G into the AIL-MAC

$$\mathrm{PDI}_r^G := \{\mathrm{PDI}_r^{g_k} : \{0,1\}^* \rightarrow \{0,1\}^\ell\}_{k\in\{0,1\}^\kappa},$$

where $\mathrm{PDI}_r^{g_k}(\cdot)$ is defined as follows.

For a message $m \in \{0,1\}^*$ (of length n), let

$$\mathrm{PDI}_r^{g_k}(m) := \begin{cases} \mathrm{PI}^{g_k}(m) & \text{if } n < r(b-1) \\ \mathrm{DI}^{g_k}(0\|m_1\|0\|m_2\|\cdots\|0\|m_r\|m_{r+1}) & \text{otherwise} \end{cases},$$

where $m_1,\ldots,m_r$ is a sequence of $(b-1)$-bit blocks and m_{r+1} a bitstring such that $m_1\|\cdots\|m_r\|m_{r+1} = m$.

The application function is $t(n) = \begin{cases} \lceil \frac{n+1}{b-1} \rceil & \text{if } n < r(b-1) \\ \lceil \frac{n+1+\ell+r}{b} \rceil & \text{otherwise} \end{cases}.$

Although not directly clear from the definition above, this construction is on-line (no matter whether $|m| < r(b-1)$ or not, the processing of m starts out in the same way).

We stress that the PDI-construction is equivalent to the DI-construction for $r = 0$ and to the PI-construction for $r \rightarrow \infty$. As is obvious from the definition of $\mathrm{PDI}_r^{\cdot}$, the construction is as efficient as $\mathrm{PI}^{\cdot}$ for messages of shorter length than $r(b-1)$ and slightly less efficient than $\mathrm{DI}^{\cdot}$ for longer messages.

Theorem 3. *A (q, μ, ε)-forger for* PDI_r^G *implies a (q', ε')-forger for G, where $q' = \frac{\mu}{b-1} + (q+1) + \frac{\ell+r}{b} \cdot \Lambda - \frac{1}{b\cdot(b-1)} \cdot \Pi$ and $\varepsilon' = \frac{\varepsilon}{\frac{1}{2}\cdot q'^2 + (\frac{1}{2}+\gamma)\cdot q' + 1}$, where*

$$(\Lambda, \Pi) := \begin{cases} (q+1, \mu) & \textit{if } r = 0 \\ \left(\left\lfloor \frac{\mu}{r(b-1)} \right\rfloor, 0\right) & \textit{if } \frac{\mu}{q+1} \le r(b-1) - 1 \\ \left(\min\left(q+1, \left\lfloor \frac{\mu}{r(b-1)} \right\rfloor\right), \mu - q(r(b-1)-1)\right) & \textit{otherwise} \end{cases},$$

and γ takes the value 1 if $\mu \ge r \cdot (b-1)$ and 0 otherwise.

Proof (Sketch). Let γ be an indicator variable that takes the value 1 if $\mu \ge r\cdot(b-1)$ and 0 otherwise. We omit the proof that $\mathcal{S}_{\mathrm{coll}} \cup \mathcal{S}_{0^\ell}$ is complete for the construction if $\gamma = 0$ and that $\mathcal{S}_{\mathrm{coll}} \cup \mathcal{S}_{0^\ell} \cup \mathcal{S}_{1^\ell}$ is complete for the PDI-construction otherwise, since it is similar to the proof of the DI- and PI-construction.[8] Applying Proposition 1 and the following fact concludes the proof.

By definition of $e(q+1, \mu)$, there exist $n_1, \ldots, n_{q+1} \in \mathbb{N}_0$ such that $e(q+1, \mu) = \sum_{i=1}^{q+1} t(n_i)$. Let ζ_i be an indicator variable that takes value 1 if $n_i \ge r(b-1)$ and 0 otherwise. We have that

$$\begin{aligned} \sum_{i=1}^{q+1} t(n_i) &\le \sum_{i=1}^{q+1} \zeta_i \cdot \left\lceil \frac{n_i + 1 + \ell + r}{b} \right\rceil + (1 - \zeta_i) \cdot \left\lceil \frac{n_i + 1}{b-1} \right\rceil \\ &\le \sum_{i=1}^{q+1} \zeta_i \cdot \frac{n_i + b + \ell + r}{b} + (1 - \zeta_i) \cdot \frac{n_i + b - 1}{b-1} \\ &\le \frac{\mu}{b-1} + (q+1) + \frac{\ell + r}{b} \cdot \sum_{i=1}^{q+1} \zeta_i - \frac{1}{b\cdot(b-1)} \sum_{i=1}^{q+1} \zeta_i \cdot n_i. \end{aligned}$$

Furthermore, it is straightforward to verify that the following two inequalities hold

$$\sum_{i=1}^{q+1} \zeta_i \le \begin{cases} q+1 & \text{if } r = 0 \\ \min\left(q+1, \left\lfloor \frac{\mu}{r(b-1)} \right\rfloor\right) & \text{otherwise} \end{cases} =: \Lambda$$

$$\sum_{i=1}^{q+1} \zeta_i \cdot n_i \ge \begin{cases} \mu & \text{if } r = 0 \\ 0 & \text{if } \frac{\mu}{q+1} \le r(b-1) - 1 \\ \mu - q(r(b-1) - 1) & \text{otherwise} \end{cases} =: \Pi.$$

As a consequence $\#\mathcal{S} + 1 \le q'^2/2 + (1/2 + \gamma) \cdot q' + 1$ by (1) and (2). □

[8] Note that if $\mu < r(b-1)$ all queries issued by the forger for $\mathrm{PDI}_r^{g_k}(\cdot)$ (including the forgery message) are shorter than $r(b-1)$ and hence $\mathrm{DI}^{g_k}(\cdot)$ is never invoked.

4 The Generalized Construction Paradigm

In this section, we generalize the construction paradigm to comprise a greater class of constructions. Furthermore, we investigate a tradeoff between the efficiency (of a construction) and the tightness (of its security reduction) in detail.

4.1 An Efficiency/Security Tradeoff

A general design goal of AIL-MAC constructions is to minimize the number of applications $t(n)$ of the FIL-MAC (where n denotes the message length). A natural approach to decrease the number of applications (which is not implied by the type of construction $\mathrm{C}^{\cdot}$) is to increase the compression parameter of the FIL-MAC before it is transformed by some construction $\mathrm{C}^{\cdot}$. However, as we will see, this is at the cost of a less tight security reduction.

To be more precise, let $f : \{0,1\}^{\ell} \to \{0,1\}^{\ell-\delta}$ be a compression function with compression parameter $\delta > 0$ and let $f^{-1}(y)$ denote the set of all preimages[9] of $y \in \{0,1\}^{\ell-\delta}$. Let $[\cdot]_f$ denote the construction, which transforms G into a FIL-MAC

$$[G]_f := \{[g_k]_f : \{0,1\}^L \to \{0,1\}^{\ell-\delta}\}_{k\in\{0,1\}^\kappa},$$

defined by

$$[g_k]_f(x) := f(g_k(x)).$$

Lemma 3. *A (q,ε)-forger F for $[G]_f$ implies a $(q,\varepsilon/s)$-forger F' for G, where $s = \max\{\#f^{-1}(y) : y \in \{0,1\}^{\ell-\delta}\}$.*

Proof. The forger F' runs F, answering all its oracle queries with the help of its own oracle. When F returns a forgery (m,τ), F' chooses an element $\hat{\tau}$ uniformly at random from $f^{-1}(\tau)$ and outputs $(m,\hat{\tau})$ as its own forgery. If F' is successful it follows that $\tau = [g_k]_f(m) = f(g_k(m))$. Thus, there is an element $\tau' \in f^{-1}(\tau)$ for which $\tau' = g_k(m)$. The probability that $\hat{\tau} = \tau'$ is

$$1/\#f^{-1}(\tau) \geq 1/s, \quad \text{where} \quad s = \max\{\#f^{-1}(y) : y \in \{0,1\}^{\ell-\delta}\}.$$

Let $\mathcal{E}'$ denote the event that F' is successful and $\mathcal{E}$ the event that F is successful. It follows that

$$\Pr[\mathcal{E}'] \geq \Pr[\mathcal{E}' \,|\, \mathcal{E}] \cdot \Pr[\mathcal{E}] \geq \underbrace{\Pr[\hat{\tau} = \tau']}_{\geq 1/s} \cdot \underbrace{\Pr[\mathcal{E}]}_{=\varepsilon}.$$

□

[9] We assume that, for all $y \in \{0,1\}^{\ell-\delta}$, one can efficiently sample an element uniformly at random from $f^{-1}(y)$.

To get as tight a security reduction in Lemma 3 as possible the largest preimage set of the key-less compression function must be as small as possible. A function achieving this is

$$\varDelta_\delta : \{0,1\}^\ell \to \{0,1\}^{\ell-\delta} \ , \quad \text{defined by} \quad x \mapsto x[1, \ell-\delta],$$

which simply cuts off the δ least significant bits of the input. As a consequence $\varDelta_\delta$ can always be chosen as the compression function without loss of generality. To simplify the notation, we write $[\cdot]_\delta$ to denote the construction $[\cdot]_{\varDelta_\delta}$.

Corollary 2. *A (q, ε)-forger for $[G]_\delta$ implies a $(q, \varepsilon/2^\delta)$ forger for G.*

Proof. Since each image of $\varDelta_\delta(\cdot)$ has equally many preimages, namely 2^δ, the largest preimage set is as small as possible. Apply Lemma 3. □

4.2 The Generalized Paradigm

The AIL-MAC $\mathrm{C}^{[G]_\delta}$ is defined by simply letting the construction $\mathrm{C}^\cdot$ transform the FIL-MAC $[G]_\delta$, which has compression parameter $b' = b + \delta$ and output-length $\ell' = \ell - \delta$. This is illustrated in Fig. 3.

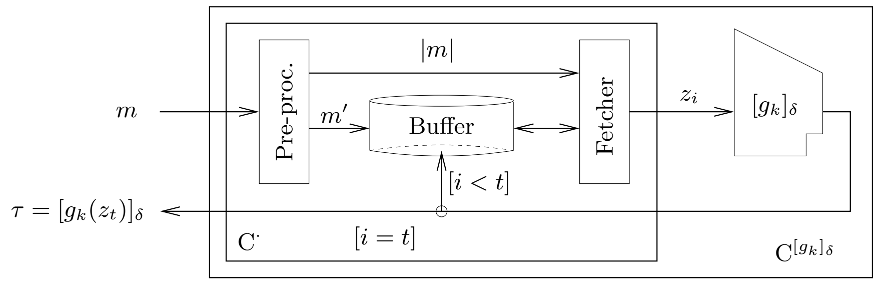

Fig. 3. The generalized construction paradigm

Since $[G]_\delta$ compresses more than G, the number of applications of the FIL-MAC G is in general smaller for $\mathrm{C}^{[\cdot]_\delta}$ than for $\mathrm{C}^\cdot$. However, this is at the cost of having a less tight security reduction for $\mathrm{C}^{[G]_\delta}$ by a factor of roughly 2^δ. The tradeoff between the efficiency and the tightness should be taken into account when comparing AIL-MAC constructions with each other (see next section).

Corollary 3. *Let b denote the compression parameter and ℓ the output-length of a FIL-MAC G.*[10] *If $t_{b,\ell}(n)$ is the application function of $C^\cdot$, then $t_{b+\delta,\ell-\delta}(n)$ is the application function of $C^{[\cdot]_\delta}$. If a (q, μ, ε)-forger for C^G implies a (q', ε')-forger for G, where*

$$q' = q'_{b,\ell}(q, \mu, \varepsilon) \quad \textit{and} \quad \varepsilon' = \varepsilon'_{b,\ell}(q, \mu, \varepsilon),$$

[10] Here we make the parameters b and ℓ explicit.

then a (q, μ, ε)-forger for $C^{[G]_\delta}$ implies a $(q'', \varepsilon''/2^\delta)$-forger for G, where

$$q'' = q'_{b+\delta,\ell-\delta}(q, \mu, \varepsilon) \quad \textit{and} \quad \varepsilon'' = \varepsilon'_{b+\delta,\ell-\delta}(q, \mu, \varepsilon).$$

Proof. The FIL-MAC $[G]_\delta$ has compression parameter $b' = b + \delta$ and output-length $\ell' = \ell - \delta$. Apply Corollary 2. □

To prove the security of a construction $C^{[\cdot]_\delta}$, one simply applies Corollary 3 (with the proof technique of Sect. 2.3).

4.3 An Illustrative Example

To illustrate the generalization and the security/efficiency tradeoff, let us first briefly recall the Chain-Rotate construction of [8] (as a reference). The CR-construction transforms any FIL-MAC G into the AIL-MAC $\mathrm{CR}^G := \{\mathrm{CR}^{g_k} : \{0,1\}^* \to \{0,1\}^\ell\}_{k \in \{0,1\}^\kappa}$, as follows.

Parse the message $m \in \{0,1\}^*$ into a sequence $\{m_i\}_{i=1}^t$ of b-bit blocks such that $m_1 \| \cdots \| m_t = m \| 10^\nu$ for a $\nu \in \{0, \ldots, b-1\}$, and let

$$\mathrm{CR}^{g_k}(m) := g_k\left(\mathrm{RR}\left(y \| m_t\right)\right) \ , \text{ where } \ y := \begin{cases} \mathrm{I}_{0^\ell}^{g_k}\left(m_1 \| \cdots \| m_{t-1}\right) & \text{if } t > 1 \\ 0^\ell & \text{otherwise} \end{cases} .$$

The application function is $t(n) = \left\lceil \frac{n+1}{b} \right\rceil$ (resulting in the waste $w(n) \in \Theta(1)$).

Theorem 4. [8] *A (q, μ, ε)-forger for CR^G implies a (q', ε')-forger for G, where $q' = \frac{\mu}{b} + (q+1)$ and $\varepsilon' = \frac{\varepsilon}{\frac{5}{2}q'^2 + \frac{3}{2}q' + 1}$.*

The efficiency of $\mathrm{CR}^{\cdot}$ is better than for $\mathrm{DI}^{\cdot}$ and $\mathrm{PI}^{\cdot}$ (just compare the application functions). However, note that the tightness of the security reduction is roughly a factor 5 worse. At first sight one might be tempted to neglect the factor 5 and consider the CR-construction as the better construction. However, as we show next, the construction $\mathrm{PI}^{[\cdot]_{\delta+1}}$ is as efficient and more secure than $\mathrm{CR}^{[\cdot]_\delta}$ for all δ. This illustrates the importance of taking the security/efficiency tradeoff into account when comparing AIL-MAC constructions.

The following two corollaries follow directly from Corollary 3, using Theorem 4 and Corollary 1, respectively.

Corollary 4. *A (q', ε')-secure FIL-MAC G implies a (q, μ, ε)-secure AIL-MAC $\mathrm{CR}^{[G]_\delta}$, for*

$$\varepsilon \geq 2^\delta \cdot \left(\frac{5}{2} \cdot q'^2 + \frac{3}{2} \cdot q' + 1\right) \cdot \varepsilon' \quad \textit{and} \quad \frac{\mu}{b+\delta} + (q+1) \leq q'.$$

The application function is $t(n) = \left\lceil \frac{n+1}{b+\delta} \right\rceil$.

Corollary 5. *A (q', ε')-secure FIL-MAC G implies a (q, μ, ε)-secure AIL-MAC* $\mathrm{PI}^{[G]_\delta}$*, for*

$$\varepsilon \geq 2^\delta \cdot \left(\frac{1}{2} \cdot q'^2 + \frac{1}{2} \cdot q' + 1\right) \cdot \varepsilon' \quad \textit{and} \quad \frac{\mu}{b + \delta - 1} + (q+1) \leq q'.$$

The application function is $t(n) = \left\lceil \frac{n+1}{b+\delta-1} \right\rceil$.

It is straightforward to verify that the application function is equivalent for $\mathrm{PI}^{[\cdot]_{\delta+1}}$ and $\mathrm{CR}^{[\cdot]_\delta}$ (and hence the efficiency is the same). Furthermore, the bound for q and μ are also equivalent for the constructions. Since the lower bound for ϵ is smaller for $\mathrm{PI}^{[\cdot]_{\delta+1}}$, by a factor of roughly 2.5, it follows that $\mathrm{PI}^{[\cdot]_{\delta+1}}$ has a tighter security reduction.

5 Comparisons of AIL-MACs

It is clear from the above that we do not need to consider $\mathrm{CS}^{[\cdot]_\delta}$ and $\mathrm{CR}^{[\cdot]_\delta}$ (for any δ) in our comparison of AIL-MAC constructions, since $\mathrm{DI}^{[\cdot]_\delta}$ is a generalization of the former and $\mathrm{PI}^{[\cdot]_{\delta+1}}$ is more efficient and has a tighter security reduction than the latter. Furthermore, recall that $\mathrm{PDI}_r^{[G]_\delta}$ is equivalent to $\mathrm{DI}^{[G]_\delta}$ for $r = 0$ and to $\mathrm{PI}^{[G]_\delta}$ for $r \to \infty$.

As a consequence, for all AIL-MAC constructions in the literature there is a choice for r and δ for which $\mathrm{PDI}_r^{[\cdot]_\delta}$ is as efficient and secure. The concrete choice for δ and the design parameter r is application dependent. Combining Corollary 3 and Theorem 3, we get:

Corollary 6. *A (q', ε')-secure FIL-MAC G implies a (q, μ, ε)-secure AIL-MAC* $\mathrm{PDI}_r^{[G]_\delta}$*, for*

$$\varepsilon \geq 2^\delta \cdot \left(\frac{q'^2}{2} + \left(\frac{1}{2} + \gamma\right) \cdot q' + 1\right) \cdot \varepsilon', \ \textit{and}$$

$$\frac{\mu}{b+\delta-1} + (q+1) + \frac{\ell - \delta + r}{b + \delta} \cdot \varLambda - \frac{1}{(b+\delta)(b+\delta-1)} \cdot \varPi \leq q',$$

where

$$(\varLambda, \varPi) := \begin{cases} (q+1, \mu) & \textit{if } r = 0 \\ \left(\left\lfloor \frac{\mu}{r \cdot (b+\delta-1)} \right\rfloor, 0\right) & \textit{if } \frac{\mu}{q+1} \leq r \cdot (b+\delta-1) - 1 \\ \left(\min\left(q+1, \left\lfloor \frac{\mu}{r\cdot(b+\delta-1)} \right\rfloor\right), \mu - q \cdot (r \cdot (b+\delta-1) - 1)\right) & \textit{otherwise} \end{cases}$$

and γ equals 1 if $\mu \geq r \cdot (b+\delta-1)$ and 0 otherwise. The application function is

$$t(n) = \begin{cases} \left\lceil \frac{n+1}{b+\delta-1} \right\rceil & \textit{if } n < r(b+\delta-1) \\ \left\lceil \frac{n+1+\ell-\delta+r}{b+\delta} \right\rceil & \textit{otherwise} \end{cases}.$$

Note that Corollary 6 is equivalent to Corollary 5 for $r \to \infty$ and to the following corollary for $r = 0$.

Corollary 7. *A (q', ε')-secure FIL-MAC G implies a (q, μ, ε)-secure AIL-MAC $\mathrm{DI}^{[G]_\delta} (\equiv \mathrm{PDI}_0^{[G]_\delta})$, for*

$$\varepsilon \geq 2^\delta \cdot \left(\frac{1}{2} \cdot q'^2 + \frac{3}{2} \cdot q' + 1\right) \cdot \varepsilon' \quad \text{and} \quad \frac{\mu}{b+\delta} + \frac{b+\ell}{b+\delta} \cdot (q+1) \leq q'.$$

The application function is $t(n) = \left\lceil \frac{n+1+\ell-\delta}{b+\delta} \right\rceil$.

6 Conclusion

In this paper, a study of a paradigm for constructing AIL-MACs by iterating applications of a FIL-MAC was continued. The paradigm was generalized in a natural way and an efficiency/security tradeoff was investigated in detail.

Our new on-line single-key AIL-MAC construction, the PDI-construction, transforms any FIL-MAC into an AIL-MAC with constant waste. It is superior to all constructions given in the literature (taking the tradeoff into account) and it appears obvious that it is essentially optimal.

An open question is whether there exists a prefix-free encoding σ' such that the PI-construction (with encoding σ') is on-line and has logarithmic waste. Our conjecture is that there is no such encoding.

References

1. J. H. An and M. Bellare. Constructing VIL-MACs from FIL-MACs: message authentication under weakened assumptions. In *Advances of Cryptology — CRYPTO 1999*, volume 1666 of *Lecture Notes in Computer Science*, pages 252–269. Springer-Verlag, 1999.
2. M. Bellare, J. Guérin, and P. Rogaway. XOR MACs: new methods for message authentication using finite pseudorandom functions. In *Advances of Cryptology — CRYPTO 1995*, volume 963 of *Lecture Notes in Computer Science*, pages 15–28. Springer-Verlag, 1995.
3. M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. In *Journal of Computer and System Sciences*, 61(3):362–399, 2000.
4. M. Bellare and P. Rogaway. Collision-resistant hashing: towards making UOWHFs practical. In *Advances in Cryptology — CRYPTO 1997*, volume 1294 of *Lecture Notes in Computer Science*, pages 470–484. Springer-Verlag 1997.
5. J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-Damgård revisited: how to construct a hash function. In *Advances of Cryptology — CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448. Springer-Verlag 2005.
6. I. Damgård. A design principle for hash functions. In *Advances in Cryptology — CRYPTO 1989*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427. Springer-Verlag, 1990.

7. U. Maurer. Indistinguishability of random systems. In *Advances of Cryptology — EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 110–132. Springer-Verlag, 2002.
8. U. Maurer and J. Sjödin. Single-key AIL-MACs from any FIL-MAC. In *Proceedings of ICALP 2005*, volume 3580 of *Lecture Notes in Computer Science*, pages 472–484. Springer-Verlag, 2005.
9. A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of applied cryptography.* CRC Press, 1997. Available on line at http://www.cacr.math.uwaterloo.ca/hac/.
10. R. Merkle. A certified digital signature. In *Advances in Cryptology — CRYPTO 1989*, volume 435 of *Lecture Notes in Computer Science*, pages 218–232. Springer-Verlag, 1990.
11. E. Petrank and C. Rackoff. CBC MAC for real-time data sources. In *Journal of Cryptology*, 13(3):315–338, 2000.
12. V. Shoup. A composition theorem for universal one-way hash functions. In *Advances of Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 445–452. Springer-Verlag 2000.