CRYPTREC evaluation report on PC-MAC-AES

John P. Steinberger
Institute for Theoretical Computer Science
Beijing, China

February 7, 2011

Contents

1 Purpose and Scope

This document is a third-party security evaluation of the PC-MAC-AES algorithm, as defined by its CRYPTREC standard [1]. This evaluation was commissioned by CRYPTREC, to be completed on January 24, 2011.

PC-MAC-AES is a Message Authentication Code (MAC) algorithm proposed by Minematsu and Tsunoo [19], loosely based off of the Pelican MAC algorithm of Daemen and Rijmen [12,13] and also inspired by the 3-key constructions of Black and Rogaway [8] and the subsequent improvements of Iwata and Kurosawa [16]. PC-MAC-AES uses a 256-bit key and produces 128-bit digests. As its name indicates, PC-MAC-AES uses the Advanced Encryption Standard (AES) as a component (moreover, the round function of AES is a standalone component used in PC-MAC-AES). PC-MAC-AES was designed to be faster than CBC-like MACs, such as OMAC [17], while retaining provable security features. The speed of PC-MAC-AES may be as much as 1.75 times faster[1] than OMAC, for $d = 5$ (where d, $1 \leq d \leq 5$, is a parameter of PC-MAC-AES; the larger the value of d, the less the provable security and the higher the amount of preprocessing, but the faster the MAC). However, the cost of this speedup is a larger amount of preprocessing and a loss in the level of provable security (OMAC achieves provable security against adversaries of time complexity 2^{64}, PC-MAC-AES against adversaries of time complexity 2^{56}).

The purpose of this document is to evaluate the security of PC-MAC-AES from the point of view of (i) its provable security, (ii) practical cryptanalysis. We also give some suggestions for the key management of PC-MAC-AES. These points are summarized below.

Provable security. Tsunoo and Minematsu [19] give a result establishing the indistinguishability of PC-MAC-AES from a random function assuming the indistinguishability of AES from a family of random permutations. (As indistinguishability from a random function implies unforgeability, this result in particular implies the the security of PC-MAC-AES against chosen plaintext attacks.)

While we found the proof to be correct this result does not, unfortunately, establish the security of PC-MAC-AES against practical adversaries, even assuming the pseudorandomness of AES, since this theorem only guarantees security against adversaries of time complexity $\approx 2^{32}$. The claim that PC-MAC-AES achieves security against adversaries of time complexity 2^{56}, put forward in [19] and [2], is in fact only valid for adversaries that ask short messages (which is a nonstandard assumption to make on an adversary).

This shortcoming is pointed out in the original security proof [19] but is left unmentioned in the self-evaluation report [2]. Thankfully, we found that substantially stronger security can be proven for PC-MAC-AES. In Section 6 we show two different such improvements. Our first result, discussed in Section 6.1, involves some minor modifications of the original proof and proves security against adversaries of time complexity 2^{42}. Our second result, discussed in Section 6.2, requires changes that are more technical, but proves security against adversaries of time complexity 2^{56}. This is the security originally (but erroneously) claimed in the self-evaluation report.

Cryptanalysis. We review the known results of [20–23] on differential cryptanalysis attacks on ALPHA-MAC and PC-MAC-AES, and their potential security implications for PC-MAC-AES. We found that these attacks do not currently pose a security risk for PC-MAC-AES. We discuss the

[1] We note that [2] contains a typo at the beginning of Section 4.1: the speedup factor of 1.8 (really 1.75) is for $d = 5$, not $d = 3$.

side-channel attack of Biryukov et. al [10] on Alpha-MAC, which currently seems to be the only SCA for this family of constructions, and the possible generalization of such an attack to PC-MAC-AES. As far as we can tell, this attack does not either pose a security risk for PC-MAC-AES. See Section 8.

Possible Improvements. Besides the provable security improvements discussed above (which just involve changing proofs) we also point out that the key of PC-MAC-AES may be halved in length with practically no loss in efficiency. See Section 7.

The reader may note that we devote less attention to differential cryptanalysis and side channel attacks than to provable security. The reasons are dual: (i) provable security is our field of expertise; (ii) resistance to differential cryptanalysis and/or side-channel attacks of a scheme is never absolute; the best proof of security, in this area, is past (and failed) scrutiny by experts. Thankfully, PC-MAC-AES does have a documented record of some such scrutiny.

1.1 Notations; conventions

We write $\{0,1\}^*$ for the set of all finite *nonempty* bit strings. We write $(\{0,1\}^n)^+$ for the set of bit strings whose length is a positive multiple of n. We write $|x|$ for the length of a bit string $x \in \{0,1\}^*$. We write $x\|y$ for the concatenation of two bit strings x and y. The notation $K \xleftarrow{\$} \mathcal{K}$ means that K is chosen uniformly at random from set $\mathcal{K}$. Following [1], we let $[k]$ denote the 128-bit encoding of an integer k, $0 \leq k < 2^{128}$. The notation $A \to 1$ indicates that the adversary A outputs the bit 1.

In this report, n everywhere denotes the block length of the scheme, which is equal to the plaintext/ciphertext length of the blockcipher used, as well as to the output length of the MAC unless truncation is performed. In the case of PC-MAC-AES the block length is $n = 128$. However, we often phrase and prove results in a more general setting, in which case the block length is an arbitrary value (also called the "security parameter").

2 Message Authentication Codes

A *message authentication code* (MAC) is a (stateless, deterministic) function $H : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^n$. Here $\mathcal{K} \subseteq \{0,1\}^*$ is the *key space* and $n \in \mathbb{N}$ is the security parameter. We write $H_K(x)$ instead of $H(K,x)$ for $K \in \mathcal{K}$ and $x \in \{0,1\}^*$. The purpose of a MAC is to assure the *data integrity* and *authenticity* of a message x transmitted between two parties sharing a secret key $K \in \mathcal{K}$:

Data integrity. Party B transmits the pair $(x, t = H_K(x))$ to party C; should (x,t) becomes accidentally corrupted to $(x',t') \neq (x,t)$, there should be low probability that $H_K(x') = t'$ (and thus a high probability that C detects the corruption).

Authenticity. A party A, without knowledge of the secret key K, should find it computationally infeasible to forge a pair (x,t) such that $H_K(x) = t$.

Authenticity is thus an adversarial version of integrity, and, as such, a stronger notion. When proving the security of a MAC, we focus on authenticity, since it implies integrity.

When quantifying authenticity, one typically assumes the (worst-case) scenario that the adversary A has oracle access to $H_K(\cdot)$; this assumption models (in A's favor) an eavesdropping capability on the communication channel between B and C. Given this oracle access to $H_K(\cdot)$, A must attempt to forge a pair (x, t) such that $H_K(x) = t$ but such that A has not yet queried x to its oracle. Most literature (e.g. [4, 6–8, 16, 17, 19]) assumes that the adversary is only allowed a single forgery attempt; however, a much more realistic assumption is to assume that the adversary has several forgery attempts [5], and wins if any of these is validated. The original security proof for PC-MAC-AES assumes a single forgery attempt [19]; thankfully, as pointed out in the self-evaluation report [2], the proof can easily be adapted to the case of multiple forgery attempts, using results of Bellare et al. [5].

3 Security Definitions

3.1 MAC security

We now give formal definitions for the security of a MAC $H : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^n$.

Let $K \in \mathcal{K}$. If H_K is given as an oracle to an adversary, we call H_K the *tagging oracle* and queries to H_K are called *tagging queries*.

A *verification oracle* V_{H_K} for H_K is a two-argument oracle of domain $\{0,1\}^* \times \{0,1\}^n$. On query $(x, t) \in \{0,1\}^* \times \{0,1\}^n$, V_{H_K} returns 1 if $H_K(x) = t$, and 0 otherwise. We call the query (x, t) a *verification query*.

The *block length* of a tagging query $x \in \{0,1\}^*$ is $\lceil x/n \rceil$. The *block length* of a verification query $(x, t) \in \{0,1\}^* \times \{0,1\}^n$ is also defined as $\lceil x/n \rceil$. The *total block length* σ of an adversary A with access to a tagging oracle and a verification oracle is the sum of the block lengths of all A's queries.

The *number of queries* q of an adversary A with access to a tagging oracle H_K and a verification oracle V_{H_K} is the sum of the number of queries A makes to H_K and the number of queries A makes to V_{H_K}. (Namely, q is the total number of queries A makes.)

We note that PC-MAC-AES pads inputs of length $|x|$ to inputs of length $n\lceil |x|/n \rceil$. Hence, in the case of PC-MAC-AES (and other related constructions such as CBC-MAC, XCBC and OMAC), the total block length σ is exactly the total number of blocks of (the first coordinates of) A's queries after padding, which is also equal to number of compression function evaluations necessary to answer A's queries. As such, $\sigma(A)$ is a lower bound on the time complexity of running A with oracles H_K, V_{H_K}, and is a more accurate reflection of the cost of A's attack than the number of queries q made by A. We note that $\sigma \geq q$. In general $\sigma > q$ unless A only makes queries that are one block long after padding.

We say an adversary A with access to a tagging oracle H_K and a verification oracle V_{H_K} (for the same key K) *wins* or *forges* or *makes a successful forgery* if A makes a query (x, t) to V_{H_K} such that $V_{H_K}(x, t) = 1$ and such that A had not previously queried x to H_K. We let

$$\mathsf{SuccessfulForgery}(A^{H_K, V_{H_K}})$$

denote this event, and we define the MAC security of H with respect to A as

$$\mathbf{Adv}^{\mathrm{mac}}_H(A) = \Pr[K \xleftarrow{\$} \mathcal{K}; \mathsf{SuccessfulForgery}(A^{H_K, V_{H_K}})]$$

where the probability is taken over the random choice of K (chosen uniformly in $\mathcal{K}$) as well as over

the coin tosses of A, if any. Overloading this notation, we let

$$\mathbf{Adv}^{\mathrm{mac}}_{H}(q, \sigma)$$

be the maximum of $\mathbf{Adv}^{\mathrm{mac}}_{H}(A)$ taken over all adversaries A that make at most q queries of total block length σ. We also let

$$\mathbf{Adv}^{\text{mac-max}}_{H}(q, \rho)$$

be the maximum of $\mathbf{Adv}^{\mathrm{mac}}_{H}(A)$ taken over all adversaries A that make at most q queries where each query has block length at most ρ. Thus

$$\mathbf{Adv}^{\text{mac-max}}_{H}(q, \rho) \leq \mathbf{Adv}^{\mathrm{mac}}_{H}(q, q\rho). \tag{1}$$

One also has, trivially (and often unsatisfactorily),

$$\mathbf{Adv}^{\mathrm{mac}}_{H}(q, \sigma) \leq \mathbf{Adv}^{\text{mac-max}}_{H}(q, \sigma). \tag{2}$$

The above notions presume an adversary which is allowed several different forgery attempts, since the adversary may win at any query. This is a stronger notion of security than what is classically considered. Classically, the adversary is only allowed *one* forgery attempt, namely only one query to its verification oracle. More precisely, we define

$$\mathbf{Adv}^{\text{mac-1}}_{H}(q, \sigma)$$

to be the maximum of $\mathbf{Adv}^{\mathrm{mac}}_{H}(A)$ taken over all adversaries A that make at most q queries of total block length σ, at most one of which is a verification query. Thus

$$\mathbf{Adv}^{\text{mac-1}}_{H}(q, \sigma) \leq \mathbf{Adv}^{\mathrm{mac}}_{H}(q, \sigma)$$

and $\mathbf{Adv}^{\text{mac-1}}_{H}(q, \sigma)$ corresponds to the "standard" (but somewhat weak) notion of chosen plaintext security.

We likewise define

$$\mathbf{Adv}^{\text{mac-1-max}}_{H}(q, \rho)$$

to be the maximum of $\mathbf{Adv}^{\text{mac-1}}_{H}(A)$ taken over all adversaries A that make at most q queries where each query has block length at most ρ. Similarly to (1) and (2) we have

$$\mathbf{Adv}^{\text{mac-1-max}}_{H}(q, \rho) \leq \mathbf{Adv}^{\text{mac-1}}_{H}(q, q\rho). \tag{3}$$

and

$$\mathbf{Adv}^{\text{mac-1}}_{H}(q, \sigma) \leq \mathbf{Adv}^{\text{mac-1-max}}_{H}(q, \sigma). \tag{4}$$

3.2 Indistinguishability and PRP security

Let $F : \mathcal{K} \times D \to \{0,1\}^n$, $F' : \mathcal{K}' \times D \to \{0,1\}^n$ be two keyed family functions of same domain D. We define the *distinguishing advantage* of an adversary A against F, F' as

$$\mathbf{Adv}^{\mathrm{cpa}}_{F,F'}(A) = \Pr[K \xleftarrow{\$} \mathcal{K}; A^{F_K} \to 1] - \Pr[K \xleftarrow{\$} \mathcal{K}'; A^{F'_K} \to 1].$$

Here 'cpa' stands for 'chosen plaintext attack'. We define $\mathbf{Adv}^{\mathrm{cpa}}_{F,F'}(q,\sigma)$ to be the maximum of $\mathbf{Adv}^{\mathrm{cpa}}_{F,F'}(A)$ over all adversaries A making at most q queries of total block length σ. (See Section 3.1.)

Let $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher of key space $\mathcal{K}$. We define the *pseudorandom permutation* security of E against an adversary A as

$$\mathbf{Adv}^{\mathrm{prp}}_{E}(A) = \Pr[K \xleftarrow{\$} \mathcal{K}; A^{E_K} \to 1] - \Pr[P \xleftarrow{\$} \mathsf{perm}(n); A^{P} \to 1].$$

Here "$P \xleftarrow{\$} \mathsf{perm}(n)$" denotes the sampling of a permutation P uniformly at random from all permutations from $\{0,1\}^n$ to $\{0,1\}^n$. We define

$$\mathbf{Adv}^{\mathrm{prp}}_{E}(q)$$

as the maximum of $\mathbf{Adv}^{\mathrm{prp}}_{E}(A)$ over all adversaries A making at most q queries. (Note the total block length of the queries is determined by the number of queries.)

3.3 PRF and VIL-PRF security

Let $R : \{0,1\}^* \to \{0,1\}^n$ be a random function and let, as before, $H : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^n$ be a MAC. The distinguishing advantage of an adversary A with access to an oracle of domain $\{0,1\}^*$ and range $\{0,1\}^n$ is defined as

$$\mathbf{Adv}^{\mathrm{vilprf}}_{H}(A) = \Pr[K \xleftarrow{\$} \mathcal{K}; A^{H_K} \to 1] - \Pr[A^{R} \to 1].$$

The probabilities are understood to be taken over the random choice of K, the coins of A and the randomness of R. We then define

$$\mathbf{Adv}^{\mathrm{vilprf}}_{H}(q,\sigma)$$

as the maximum of $\mathbf{Adv}^{\mathrm{vilprf}}_{H}(A)$ taken over all adversaries A making at most q queries, of total block length σ. It is easy to see, as observed by Bellare et al. [5], that

$$\mathbf{Adv}^{\mathrm{mac}}_{H}(q,\sigma) \le \frac{q}{2^n} + \mathbf{Adv}^{\mathrm{vilprf}}_{H}(q,\sigma). \tag{5}$$

For completeness, and since a complete proof of (5) does not appear in [5], we take a moment to provide a proof here.

Proof of (5). Let B be an optimal (q,σ)-MAC adversary for H, namely an adversary such that $\mathbf{Adv}^{\mathrm{mac}}_{H}(B) = \mathbf{Adv}^{\mathrm{mac}}_{H}(q,\sigma)$ and such that B makes at most q queries of total block length at most σ. We construct a PRF adversary A for H as follows: A simulates B; if B successfully forges, then A outputs 1, otherwise A outputs 0. Clearly, A makes at most q queries to its oracle, of total block length at most σ. We have

$$\Pr[K \xleftarrow{\$} \mathcal{K}; A^{H_K} \to 1] = \mathbf{Adv}^{\mathrm{mac}}_{H}(B)$$

and

$$\Pr[A^{R} \to 1] \le \frac{q}{2^n}$$

so, by definition,

$$\mathbf{Adv}^{\mathrm{vilprf}}_{H}(A) \geq \mathbf{Adv}^{\mathrm{mac}}_{H}(B) - \frac{q}{2^n}$$

as desired. □

We note that if H^π denotes H truncated to $\pi \leq n$ bits (i.e., $H^\pi_K(x)$ is the first π bits of $H_K(x)$), then we have likewise that

$$\mathbf{Adv}^{\mathrm{mac}}_{H^\pi}(q,\sigma) \leq \frac{q}{2^\pi} + \mathbf{Adv}^{\mathrm{vilprf}}_{H^\pi}(q,\sigma) \tag{6}$$

which in turn implies that

$$\mathbf{Adv}^{\mathrm{mac}}_{H^\pi}(q,\sigma) \leq \frac{q}{2^\pi} + \mathbf{Adv}^{\mathrm{vilprf}}_{H}(q,\sigma) \tag{7}$$

since $\mathbf{Adv}^{\mathrm{vilprf}}_{H^\pi}(q,\sigma) \leq \mathbf{Adv}^{\mathrm{vilprf}}_{H}(q,\sigma)$ (indeed, truncating can only hurt the adversary's distinguishing advantage).

Analogously to $\mathbf{Adv}^{\text{mac-max}}_{H}(q,\rho)$, we define

$$\mathbf{Adv}^{\text{vilprf-max}}_{H}(q,\rho)$$

to be the maximum of $\mathbf{Adv}^{\mathrm{vilprf}}_{H}(A)$ taken over all adversaries A making q queries each of block length at most ρ. We have, similarly to (5), that

$$\mathbf{Adv}^{\text{mac-max}}_{H}(q,\rho) \leq \frac{q}{2^n} + \mathbf{Adv}^{\text{vilprf-max}}_{H}(q,\rho) \tag{8}$$

and, if H^π again denotes the truncation of H to $\pi \leq n$ bits, that

$$\mathbf{Adv}^{\text{mac-max}}_{H^\pi}(q,\rho) \leq \frac{q}{2^\pi} + \mathbf{Adv}^{\text{vilprf-max}}_{H}(q,\rho). \tag{9}$$

Let $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher of key space $\mathcal{K}$. We define the PRF security of E against an adversary A as

$$\mathbf{Adv}^{\mathrm{prf}}_{E}(A) = \Pr[K \xleftarrow{\$} \mathcal{K}; A^{E_K} \to 1] - \Pr[R \xleftarrow{\$} \mathsf{func}(n,n); A^{R} \to 1].$$

Here "$R \xleftarrow{\$} \mathsf{func}(n,n)$" denotes the sampling of a function R uniformly at random from all functions from $\{0,1\}^n$ to $\{0,1\}^n$. We define

$$\mathbf{Adv}^{\mathrm{prf}}_{E}(q)$$

as the maximum of $\mathbf{Adv}^{\mathrm{prf}}_{E}(A)$ over all adversaries A making at most q queries. (Note here too the total block length of the queries is determined by the number of queries.)

3.4 Computational securities

In the above security definitions, the computational power of the adversaries is not mentioned (and therefore not restricted). This can be meaningful in certain information-theoretic settings. However, when discussing, say, the PRP security of AES-128, one must consider computationally bounded adversaries in order to get meaningful statements. (Indeed, AES-128 can be distinguished from a PRP in $O(1)$ queries if we allow computationally unbounded adversaries.)

Each of the security notions $\mathbf{Adv}^{\mathrm{mac}}_{H}$, $\mathbf{Adv}^{\mathrm{prp}}_{E}$, $\mathbf{Adv}^{\mathrm{vilprf}}_{H}$, $\mathbf{Adv}^{\mathrm{cpa}}_{F,F'}$, etc, defined so far can be augmented with a *time bound* t on the adversary. For example, we write

$$\mathbf{Adv}^{\mathrm{mac}}_{H}(q, \sigma, t)$$

for the maximum of $\mathbf{Adv}^{\mathrm{mac}}_{H}(A)$ over all adversaries A of running time at most t, making at most q queries of total length at most σ. Here the meaning of "running time" is supposed fixed with respect to some reasonable model of computation. What matters is establishing meaningful reductions from one type of security to another: for example, showing that if $\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{PC-MAC-AES}}(q, \sigma, t)$ is large (meaning bad security), where H uses a blockcipher E, then it must be that (say) $\mathbf{Adv}^{\mathrm{prp}}_{AES}(\sigma, t')$ is large for a value of t' not much larger than t. Eventually, all security bounds must reduce to the PRP security of the underlying blockcipher.

3.5 Collision security, MEDP and MESDP

Let $H : \mathcal{K} \times (\{0,1\}^n)^+ \to \{0,1\}^n$ be a keyed function whose domain $(\{0,1\}^n)^+$ is the set of all strings whose length is a positive multiple of n. Then for positive integers m, m' we define

$$\mathbf{Adv}^{\mathrm{coll}}_{H}(m, m') = \max_{x \in \{0,1\}^{nm}, x' \in \{0,1\}^{nm'}, x \neq x'} \Pr_K[H_K(x) = H_K(x')].$$

We note the probability is only taken over the key K, and that this definition involves no adversary; however we keep the **Adv**-type notation for the sake of uniformity.

Let $P : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a keyed permutation of $\{0,1\}^n$; that is $P_K(\cdot) = P(K, \cdot)$ is a permutation of $\{0,1\}^n$ for all $K \in \mathcal{K}$. Then we define the *maximum expected differential probability*, or MEDP of P as

$$\mathrm{MEDP}(P) = \max_{a,b \in \{0,1\}^n, a \neq 0} \Pr_{K \xleftarrow{\$} \mathcal{K}, X \xleftarrow{\$} \{0,1\}^n} [P_K(X) \oplus P_K(X \oplus a) = b]$$

(The notation is meant to indicate the probability is taken over the random choice of $X \in \{0,1\}^n$ and $K \in \mathcal{K}$.)

We finally define the *maximum expected self-differential probability* of P as

$$\mathrm{MESDP}(P) = \max_{a \{0,1\}^n} \Pr_{K \xleftarrow{\$} \mathcal{K}, X \xleftarrow{\$} \{0,1\}^n} [X \oplus P_K(X) = a].$$

The MEDP of blockciphers such as AES has been studied in connection with differential cryptanalysis [14]. The MESDP was introduced by Minematsu and Tsunoo [19].

4 The PC-MAC-AES Specification

In this section we reproduce the PC-MAC-AES specfication [2]. We do so to establish notation as well as for the reader's convenience. Some minor differences in notation exist with [2]; we mostly introduced these in order to better harmonize with [19].

We note that 384 (which appears many places in the specification) is $3 \cdot 128$.

4.1 The simplified 4-round AES function

Let SubBytes : $\{0,1\}^{128} \to \{0,1\}^{128}$, ShiftRows : $\{0,1\}^{128} \to \{0,1\}^{128}$, MixColumns : $\{0,1\}^{128} \to \{0,1\}^{128}$ be be the three components of the AES round function, as specified in [3]. Let Rnd : $\{0,1\}^{128} \to \{0,1\}^{128}$ be the composition

$$Rnd(x) = \text{MixColumns}(\text{ShiftRows}(\text{SubBytes}(x))).$$

(We note that Rnd is an unkeyed permutation.) Thus a round of AES, applied to a 128-bit value x, consists of xoring x with a 128-bit "subkey" followed by an application of Rnd. (AES-128 consists of 10 such rounds, where the subkeys for each round are derived from a single 128-bit key using AES's key scheduling mechanism; the key scheduling mechanism of AES is of no interest for what follows.)

Let $U = U^{(1)}\|U^{(2)}\|U^{(3)}$ be a 384-bit value where each $U^{(i)}$ is a 128-bit value. The $U^{(i)}$'s are called *subkeys*. The 4-round AES function $4\text{r}_U : \{0,1\}^{128} \to \{0,1\}^{128}$ with key U is defined by

$$4\text{r}_U(x) = Rnd(U^{(3)} \oplus Rnd(U^{(2)} \oplus Rnd(U^{(1)} \oplus Rnd(x)))). \tag{10}$$

We note that 4r_U involves 4 applications of the AES round function Rnd. However, an xor with a subkey is missing in the first round, and hence we refer to 4r_U as the "simplified 4-round AES function".

For the remainder of the report we let $4\text{r} : \{0,1\}^{384} \times \{0,1\}^{128} \to \{0,1\}^{128}$ be the keyed function defined by $4\text{r}(U,x) = 4\text{r}_U(x)$, where $4\text{r}_U(x)$ is defined by (10).

4.2 Glossary of notations and basic functions

- K : 128-bit key of AES
- L : secondary 128-bit key
- d : order (a parameter of PC-MAC-AES, positive integer)
- π : bit length of final truncated tag (a parameter of PC-MAC-AES, a positive integer at most 128)
- $\text{AES}_K(\cdot)$: the AES-128 encryption function on key $K \in \{0,1\}^{128}$
- 4r_U : simplified 4-round AES function with 384-bit key, U
- $\text{mul2}(x)$: multiply-by-two operation defined over the finite field $\text{GF}(2^{128})$, defined as

$$\text{mul2}(x) = \begin{cases} x \ll 1 & \text{if msb}(x) = 0 \\ (x \ll 1) \oplus (0^{120}\|10000111) & \text{otherwise} \end{cases}$$

where msb(x) is the most significant bit of x.
- $\text{pad}(x)$: a padding function for x with $|x| \leq 128$ defined as

$$\text{pad}(x) = \begin{cases} x & \text{if } |x| = 128 \\ x\|1\|0^{128-|x|-1} & \text{if } |x| < 128 \end{cases}$$

4.3 Parameters

PC-MAC-AES takes two parameters: the *order* d, which is an integer between 1 and 5 (one could, syntactically speaking, use larger values than 5, but this is not recommended by the specification)

as well as the *output length* π, an integer between 64 and 128 (one could likewise use smaller values than 64, but this is likewise not recommended). We note that π does not affect PC-MAC-AES until the 128-bit output is obtained, at which point the final output is simply obtained by truncating the 128-bit output to its first π bits. We discuss the role of d below.

4.4 Key and key schedule

PC-MAC-AES uses a 256-bit key $K\|L$ where $K, L \in \{0,1\}^{128}$. From K and L, an additional $d \cdot 384 + (d-1) \cdot 128$ bits of key material are constructed. More precisely, d 384-bit values $U_1, \ldots, U_d$ are constructed by putting $U_i = U_i^{(1)} \| U_i^{(2)} \| U_i^{(3)}$ where

$$U_i^{(j)} = \mathrm{AES}_K(L \oplus [3(i-1) + (j-1)])$$

(we recall that $[k]$ stands for the 128-bit binary encoding of the integer k); moreover, $(d-1)$ 128-bit values $K_1^{\mathrm{xor}}, \ldots, K_{d-1}^{\mathrm{xor}}$ are built by letting

$$K_j^{\mathrm{xor}} = \mathrm{AES}_K(L \oplus [3d + j - 1]).$$

Thus the values $U_1^{(1)}, \ldots, U_d^{(3)}, K_1^{\mathrm{xor}}, \ldots, K_{d-1}^{\mathrm{xor}}$ are generated from $\mathrm{AES}_K(L \oplus \cdot)$ in counter mode.

We let $K_{\mathrm{aux}} = U_1 \| \cdots \| U_d \| K_1^{\mathrm{xor}} \| \cdots \| K_{d-1}^{\mathrm{xor}}$ denote the $d \cdot 384 + (d-1) \cdot 128$ additional bits of "auxiliary" key material.

4.5 Tag generation

Assume fixed values of d, π, K, L and let $U_1, \ldots, U_d, K_1^{\mathrm{xor}}, \ldots, K_{d-1}^{\mathrm{xor}}$ be generated as described above. The core components of PC-MAC-AES are the (fixed-key) AES encryption function AES_K as well as the simplified 4-round AES functions $4\mathrm{r}_{U_1}, \ldots, 4\mathrm{r}_{U_d}$.

Let $x \in \{0,1\}^*$ be a message whose MAC is to be computed; the π-bit MAC of x under PC-MAC-AES is given by the following algorithm (in which the generation of the auxiliary key is assumed):

function PC-MAC-AES(d, π, K, L, x)
let $x = x_1 \| \cdots \| x_m$ where $|x_i| = 128$ for $i < m$ and $1 \leq |x_m| \leq 128$
$s \leftarrow 0^{128}$
for $i \leftarrow 1$ to $m-1$ **do**
  $w \leftarrow (i-1) \bmod (d+1)$
  if $w = 0$ **then** $s \leftarrow \mathrm{AES}_K(s \oplus x_i)$
  else if $w = 1$ **then** $s \leftarrow 4\mathrm{r}_{U_1}(s \oplus x_i)$
  else $s \leftarrow 4\mathrm{r}_{U_w}(s \oplus K_{w-1}^{\mathrm{xor}} \oplus x_i)$
$h \leftarrow s \oplus \mathrm{pad}(x_m)$
if $|x_m| = 128$ **then** $h \leftarrow \mathrm{mul2}(L) \oplus h$
else $h \leftarrow \mathrm{mul2}(\mathrm{mul2}(L)) \oplus h$
return the first π bits of $\mathrm{AES}_K(h)$

We refer the reader to Figures 2 and 3 of [1] and Figure 1 of [19]. (We note that in these figures, G_U is our $4\mathrm{r}_U$.)

We let

$$\text{PC-MAC-AES}^{\pi,d}_{K,L}(x)$$

be the output of the pseudocode given. We note this defines, in particular, a keyed function $\text{PC-MAC-AES}^{\pi,d}$ of 256 bit key (the key being the pair (K, L)).

4.6 Additional notations for PC-MAC-AES

In this section we give some notations for generalizations of PC-MAC-AES, and for some components of these generalizations. These generalizations are used as hybrids in the security proofs. There are three main hybrids, with attendant notations, that we use. In all hybrids n stands for the generalized block length. (Originally $n = 128$ and one may keep thinking of n as having value 128, if this is helpful.) In addition, we define a sub-component of the last hybrid in Section 4.6.4.

4.6.1 PC-MAC$^{\circ}_d[E_K, L, G]$

Let $G : \{0,1\}^{3n} \to \{0,1\}^n$ be any keyed permutation of $\{0,1\}^n$ of key length $3n$ and output length n. G is an abstract version of the simplified 4-round AES function 4r. We write $G_U(x)$ for $G(U, x)$, as usual, where $U = U^{(1)} \| U^{(2)} \| U^{(3)} \in \{0,1\}^{3n}$ and $x \in \{0,1\}^n$.

Let $E : \{0,1\}^n \to \{0,1\}^n$ be any blockcipher, writing $E_K(x)$ for $E(K, x)$, as usual; E_K generalizes AES_K.

We now write $\text{PC-MAC}^{\circ}_d[E_K, L, G]$ for the PC-MAC-AES algorithm applied with E_K substituted for AES_K and with G substituted for 4r. More precisely, $\text{PC-MAC}^{\circ}_d[E_K, L, G]$ uses an auxiliary key $K_{\text{aux}} = U_1 \| \cdots \| U_d \| K_1^{\text{xor}} \| \cdots \| K_{d-1}^{\text{xor}}$ which is generated by the function $E_K(L \oplus \cdot)$ in counter mode. (I.e., $U_1^{(1)} = E_K(L \oplus [0])$, $U_1^{(2)} = E_K(L \oplus [1])$, etc.) Then the value $\text{PC-MAC}^{\circ}_d[E_K, L, G](x)$ is computed by the following algorithm (in which once again we assume the existence of the auxiliary key without mention of its precomputation):

function $\text{PC-MAC}^{\circ}_d[E_K, L, G](x)$
let $x = x_1 \| \cdots \| x_m$ where $|x_i| = n$ for $i < m$ and $1 \le |x_m| \le n$
$s \leftarrow 0^n$
for $i \leftarrow 1$ to $m - 1$ **do**
  $w \leftarrow (i - 1) \bmod (d + 1)$
  if $w = 0$ **then** $s \leftarrow E_K(s \oplus x_i)$
  else if $w = 1$ **then** $s \leftarrow G_{U_1}(s \oplus x_i)$
  else $s \leftarrow G_{U_w}(s \oplus K_{w-1}^{\text{xor}} \oplus x_i)$
$h \leftarrow s \oplus \text{pad}(x_m)$
if $|x_m| = n$ **then** $h \leftarrow E_K(\text{mul2}(L) \oplus h)$
else $h \leftarrow E_K(\text{mul2}(\text{mul2}(L)) \oplus h)$
return h

We note that we have dropped all mention of π, the truncation length; thus $\text{PC-MAC}_d[R, L, G]$ returns an output of length n. Now $\text{mul2}(\cdot)$ stands for multiplication by 2 over $\text{GF}(2^n)$, for any representation of $\text{GF}(2^n)$, and $\text{pad}(\cdot)$ stands for padding to length n instead of to length 128 (generalized in the natural way).

We view $\text{PC-MAC}^{\circ}_d[E_K, L, G]$ as a keyed function, with key (K, L).

4.6.2 PC-MAC$^*_d[R, L, G]$

This hybrid is an abstraction (generalization) of PC-MAC$^\circ_d[E_K, L, G]$. We keep G as in the previous section. Let R be an arbitrary function from $\{0,1\}^n$ to $\{0,1\}^n$.

We now write PC-MAC$^*_d[R, L, G]$ for the PC-MAC$^\circ_d[E_K, L, G]$ with E_K substituted by R. More precisely, PC-MAC$^*_d[R, L, G]$ uses an auxiliary key $K_{\text{aux}} = U_1\|\cdots\|U_d\|K_1^{\text{xor}}\|\cdots\|K_{d-1}^{\text{xor}}$ which is generated by the function $R(L \oplus \cdot)$ in counter mode. Then the value PC-MAC$^*_d[R, L, G](x)$ is computed by the following algorithm:

function PC-MAC$^*_d[R, L, G](x)$
let $x = x_1\|\cdots\|x_m$ where $|x_i| = n$ for $i < m$ and $1 \le |x_m| \le n$
$s \leftarrow 0^n$
for $i \leftarrow 1$ to $m-1$ **do**
  $w \leftarrow (i-1) \bmod (d+1)$
  if $w = 0$ **then** $s \leftarrow R(s \oplus x_i)$
  else if $w = 1$ **then** $s \leftarrow G_{U_1}(s \oplus x_i)$
  else $s \leftarrow G_{U_w}(s \oplus K_{w-1}^{\text{xor}} \oplus x_i)$
$h \leftarrow s \oplus \text{pad}(x_m)$
if $|x_m| = n$ **then** $h \leftarrow R(\text{mul2}(L) \oplus h)$
else $h \leftarrow R(\text{mul2}(\text{mul2}(L)) \oplus h)$
return h

We still view PC-MAC$^*_d[R, L, G]$ as a keyed function, whose key consists of the n-bit value L and of a $n2^n$-bit description of R. (Put differently, sampling the key space of PC-MAC$^*_d$ amounts to sampling an n-bit value L uniformly at random in $\{0,1\}^n$ as well as a function R uniformly at random in $\mathsf{func}(n, n)$, the set of all functions from $\{0,1\}^n \times \{0,1\}^n$.)

4.6.3 PC-MAC$^\star_d[\mathbf{R}, G]$

This hybrid is an abstraction (generalization) of PC-MAC$^*_d[R, L, G]$. We keep G as in the previous section.

Let R, R', R'' and R''' be any functions from $\{0,1\}^n$ to $\{0,1\}^n$. These will be abstract versions of, respectively, the functions $R(\cdot)$, $R(\text{mul2}(L) \oplus \cdot)$, $R(\text{mul2}(\text{mul2}(L)) \oplus \cdot)$ and $R(L \oplus \cdot)$ used in PC-MAC$^*_d[R, L, G]$. We write $\mathbf{R}$ for the 4-tuple (R, R', R'', R''').

We now write PC-MAC$^\star_d[\mathbf{R}, G]$ for the PC-MAC$^*_d[R, L, G]$ algorithm applied with R, R', R'', R''' substituted for the functions $R(\cdot)$, $R(\text{mul2}(L) \oplus \cdot)$, $R(\text{mul2}(\text{mul2}(L)) \oplus \cdot)$ and $R(L \oplus \cdot)$.

More precisely, PC-MAC$^\star_d[\mathbf{R}, G]$ involves an auxiliary key $K_{\text{aux}} = U_1\|\cdots\|U_d\|K_1^{\text{xor}}\|\cdots\|K_{d-1}^{\text{xor}}$ which is generated by the function R''' in counter mode: $U_1^{(1)} = R'''([0])$, $U_1^{(2)} = R'''([1])$, etc. Then the value PC-MAC$_d[\mathbf{R}, G](x)$ is computed by the following algorithm:

function PC-MAC$^\star_d[\mathbf{R}, G](x)$
let $x = x_1\|\cdots\|x_m$ where $|x_i| = n$ for $i < m$ and $1 \le |x_m| \le n$
$s \leftarrow 0^n$
for $i \leftarrow 1$ to $m-1$ **do**
  $w \leftarrow (i-1) \bmod (d+1)$
  if $w = 0$ **then** $s \leftarrow R(s \oplus x_i)$

else if $w = 1$ **then** $s \leftarrow G_{U_1}(s \oplus x_i)$
else $s \leftarrow G_{U_w}(s \oplus K^{\text{xor}}_{w-1} \oplus x_i)$
$h \leftarrow s \oplus \text{pad}(x_m)$
if $|x_m| = n$ **then** $h \leftarrow R'(h)$
else $h \leftarrow R''(h)$
return h

We note that R''' does not appear in the pseudocode above; however, R''' is used to generate the auxiliary key K_{aux}. We also note that by definition, then, if R''' is sampled at random from all functions from n bits to n bits, K_{aux} is sampled independently and at random from all bit strings of length $(4d-1)n$.

4.6.4 PC-MAC$^{\bullet}_d[\mathbf{R}, G]$

Let $\mathbf{R}$ and G be as in the last section, and also let the auxiliary key K_{aux} be generated from R''' as in the last section. PC-MAC$^{\bullet}_d[\mathbf{R}, G]$ is an algorithm that takes an input x whose length is a multiple of n, and returns a value h as follows:

function PC-MAC$^{\bullet}_d[\mathbf{R}, G](x)$
let $x = x_1 \| \cdots \| x_m$ where $|x_i| = n$ for $1 \le i \le m$, and where $m \ge 1$
$s \leftarrow 0^n$
for $i \leftarrow 1$ to $m-1$ **do**
$w \leftarrow (i-1) \bmod (d+1)$
if $w = 0$ **then** $s \leftarrow R(s \oplus x_i)$
else if $w = 1$ **then** $s \leftarrow G_{U_1}(s \oplus x_i)$
else $s \leftarrow G_{U_w}(s \oplus K^{\text{xor}}_{w-1} \oplus x_i)$
$h \leftarrow s \oplus \text{pad}(x_m)$
return h

We note that PC-MAC$^{\bullet}_d[\mathbf{R}, G]$ is simply PC-MAC$^{\star}_d[\mathbf{R}, G]$ restricted to padded inputs and with the final application of R', R'' removed. In fact, we note that we can implement PC-MAC$^{\star}_d[\mathbf{R}, G]$ using PC-MAC$^{\bullet}_d[\mathbf{R}, G]$ as a component, as follows:

function PC-MAC$^{\star}_d[\mathbf{R}, G](x)$
let $x = x_1 \| \cdots \| x_m$ where $|x_i| = n$ for $i < m$ and $1 \le |x_m| \le n$
let $h \leftarrow$ PC-MAC$^{\bullet}_d[\mathbf{R}, G](x_1 \| \cdots \| x_{m-1} \| \text{pad}(\text{x}_\text{m}))$
if $|x_m| = n$ **then** $h \leftarrow R'(h)$
else $h \leftarrow R''(h)$
return h

Like PC-MAC$^{\star}_d[\mathbf{R}, G]$, we view PC-MAC$^{\bullet}_d[\mathbf{R}, G]$ as keyed function family, keyed by $\mathbf{R}$.

5 Results on provable security

5.1 Overview of results

We start by reviewing the security results of Minematsu and Tsunoo [19] ('MT' for short). MT give an upper bound on $\mathbf{Adv}^{\text{vilprf-max}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\rho,t)$:

Theorem 1 [19] *Let* $\text{PC-MAC}^{\circ}_{d}[E_K, L, G]$ *be as defined in Section 4.6.1, viewed as a function of keyspace* $\{0,1\}^{2n}$*. Then*

$$\mathbf{Adv}^{\text{vilprf-max}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\rho,t) \le \mathbf{Adv}^{\text{prp}}_{E}(\rho q + 4d, t') + \frac{2.5(\rho q + 4d)^2}{2^n} + (d\epsilon_{\text{dp}} + \epsilon_{\text{sdp}})\frac{q^2}{2} \qquad (11)$$

where $t' = t + O(\rho q)$*,* $\epsilon_{\text{dp}} = \text{MEDP}(G)$ *and* $\epsilon_{\text{sdp}} = \text{MESDP}(G)$*.*

We note that truncation cannot lessen the PRF-security of a function, so the bound Theorem 1 also applies to any truncated version of $\text{PC-MAC}^{\circ}_{d}[E_K, L, G]$.

In particular, letting $n = 128$, letting $E_K = \text{AES}_K$ and letting $G = 4\text{r}$, we obtain a corollary on $\text{PC-MAC-AES}^{\pi,d}$. Keliher et al. [14] proved that $\text{MEDP}(4\text{r}) \le 2^{-113}$, and, as observed by MT, it is easy to see that $\text{MESDP}(4\text{r}) = 2^{-128}$. Thus:

Corollary 1 [19] *Let* $\text{PC-MAC-AES}^{\pi,d}$ *be as defined in Section 4.5. Then*

$$\mathbf{Adv}^{\text{vilprf-max}}_{\text{PC-MAC-AES}^{\pi,d}}(q,\rho,t) \le \mathbf{Adv}^{\text{prp}}_{\text{AES-128}}(\rho q + 4d, t') + \frac{2.5(\rho q + 4d)^2}{2^{128}} + \left(\frac{d}{2^{114}} + \frac{1}{2^{129}}\right) q^2$$

where $t' = t + O(\rho q)$*.*

Then by (the computational analogue of) (9) and Corollary 1 we have, for example, that

$$\mathbf{Adv}^{\text{mac-max}}_{\text{PC-MAC-AES}^{\pi,d}}(q,\rho,t) \le \\ \frac{q}{2^{\pi}} + \mathbf{Adv}^{\text{prp}}_{\text{AES-128}}(\rho q + 4d, t') + \frac{2(\rho q + 4d)^2}{2^{128}} + \left(\frac{d}{2^{114}} + \frac{1}{2^{129}}\right) q^2 \qquad (12)$$

there $t' = t + O(\rho q)$.

We note that (12) is inequality (4) on page 5 of the self-evaluation report [2]. The self-evaluation report claims that "in practice, $\frac{q^2}{2^{114}}$ is the dominant term", and hence that PC-MAC-AES is secure against adversaries of time complexity at most 2^{56} (provided, say, $\pi \ge 64$, as recommended by the specification). However, (12) certainly does *not* show provable security against adversaries of time complexity 2^{56}. Indeed, an adversary that asks a single message of 2^{32} blocks followed by $2^{32} - 1$ messages of 1 block each (thus setting $\rho = q = 2^{32}$) has time complexity 2^{33}, but (12) gives a void bound for such an adversary. At issue is the fact (12) gives a bound on $\mathbf{Adv}^{\text{mac-max}}(q,\rho)$ of order $\rho^2 q^2/2^n$ rather than giving a bound on $\mathbf{Adv}^{\text{mac}}(q,\sigma)$ of order $\sigma^2/2^n$ (since σ is an accurate reflection of the time complexity of an adversary, whereas ρq can severly overestimate the time complexity of an adversary). This deficiency was already underlined in [19], though it is not mentioned in [2]. We return to this question in Section 6.

5.2 Proof overview

In this section we sketch the proof of Theorem 1. The proof sketch we give does little accounting (as usual) of the time complexities involved in the various reductions; suffice it to say that these reductions are efficient.

The proof uses the so-called PRF/PRP switching lemma (folklore), which states that

$$|\mathbf{Adv}^{\mathrm{prf}}_{E}(q) - \mathbf{Adv}^{\mathrm{prp}}_{E}(q)| \leq \frac{q^2}{2^{n+1}}$$

for any blockcipher $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. Thus to show (11) it is sufficient to show

$$\mathbf{Adv}^{\text{vilprf-max}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\rho,t) \leq \mathbf{Adv}^{\mathrm{prf}}_{E}(\rho q + 4d, t') + \frac{2(\rho q + 4d)^2}{2^n} + (d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}})\frac{q^2}{2}. \tag{13}$$

In turn, (13) is implied, using a standard hybrid argument, by

$$\mathbf{Adv}^{\text{vilprf-max}}_{\text{PC-MAC}^{*}_{d}[R,L,G]}(q,\rho) \leq \frac{2(\rho q + 4d)^2}{2^n} + (d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}})\,\frac{q^2}{2}. \tag{14}$$

(Note that (14) is an information-theoretic statement—note also that, as explained in Section 4.6.2, $\text{PC-MAC}^{*}_{d}[R,L,G]$ is keyed by R as well as by L.) In fact, MT show the slightly stronger inequality:

$$\mathbf{Adv}^{\text{vilprf-max}}_{\text{PC-MAC}^{*}_{d}[R,L,G]}(q,\rho) \leq \frac{(\rho q + 4d)^2}{2^{n+1}} + \left(d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}} + \frac{2\rho^2+1}{2^n}\right)\frac{q^2}{2}. \tag{15}$$

A crucial ingredient in proving (15) is the following lemma, stated without proof in [19] but easy to prove from a similar lemma by Iwata and Kurosawa [16]. We note that in the statement of this lemma, "$\mathrm{mul2}(1)^i$" is our notation for "the constant 2 in $\mathrm{GF}(2^n)$, taken to the power i"; thus $\mathrm{mul2}(1)^0 \cdot L = L$ and $\mathrm{mul2}(1)^1 \cdot L = \mathrm{mul2}(L)$, etc.

Lemma 1 *Let $\mathcal{R}$ be the set of all functions from $\{0,1\}^n$ to $\{0,1\}^n$ and let $D = \{0,1\}^n \times \{0,1,2,3\}$. Let $\mathcal{K} = \mathcal{R} \times \{0,1\}^n$ and let $F : \mathcal{K} \times D$ be defined by $F_{R,L}(x,0) = R(x)$ and $F_{R,L}(x,i) = R(\mathrm{mul2}(1)^{i-1} \cdot L \oplus x)$ for $i = 1,2,3$, where $(R,L) \in \mathcal{K} = \mathcal{R} \times \{0,1\}^n$. Let $\mathcal{K}' = \mathcal{R}^4$, and let $F' : \mathcal{K}' \times D$ be defined by $F'_{R_0,R_1,R_2,R_3}(x,i) = R_i(x)$, where $(R_1,R_2,R_3,R_4) \in \mathcal{K}' = \mathcal{R}^4$. Then $\mathbf{Adv}^{\mathrm{cpa}}_{F,F'}(q) \leq \frac{q^2}{2^{n+1}}$.*

Since a PRF adversary against $\text{PC-MAC}^{*}_{d}[R,L,G]$ making q queries each of length at most ρ can be simulated with at most $\rho q + 4d$ queries to R, it is straightforward from Lemma 1 and the definition of $\text{PC-MAC}^{\star}_{d}[\mathbf{R},G]$ that (15) is implied by

$$\mathbf{Adv}^{\text{vilprf-max}}_{\text{PC-MAC}^{\star}_{d}[\mathbf{R},G]}(q,\rho) \leq \left(d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}} + \frac{2\rho^2+1}{2^n}\right)\frac{q^2}{2}. \tag{16}$$

To prove (16), MT use the following lemma of Black and Rogaway [8] (presented here in slightly disguised form, namely as it applies to $\text{PC-MAC}^{\star}_{d}$):

$$\mathbf{Adv}^{\text{vilprf-max}}_{\text{PC-MAC}^{\star}_{d}[\mathbf{R},G]}(q,\rho) \leq \max_{m_1,\ldots,m_q,\ m_s \leq \rho}\left(\sum_{1\leq i<j\leq q}\mathbf{Adv}^{\mathrm{coll}}_{\text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G]}(m_i,m_j)\right). \tag{17}$$

The heart of MT's analysis is the following lemma:

Lemma 2 ([19], Lemma 3) *We have*

$$\mathbf{Adv}^{\text{coll}}_{\text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G]}(m,m') \leq d\epsilon_{\text{dp}} + \epsilon_{\text{sdp}} + \frac{(m+m')^2+2}{2^{n+1}}$$

for all positive integers m, m'.

(In fact MT show a slightly stronger result than Lemma 2, but the bound they apply in the end is the one above.) From Lemma 2 and (17), one can show (16).

While the proof of Lemma 2 is really the heart of MT's analysis it is also quite technical and "beyond the scope" of this report. However, we have verified it in detail, and are satisfied of its correctness (and, by extension, of the correctness of the results mentioned in Section 5.1). Readers who succeed in reading Section 6.2 will also get a flavor for the proof of Lemma 2.

6 Provable security improvements

As mentioned in Section 5.1, (12) is not a really satisfactory security bound, since it only guarantees security against adversaries of time complexity $\approx 2^{32}$. Minematsu and Tsunoo actually mention the desirability of proving an upper bound on $\mathbf{Adv}^{\text{mac}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\sigma)$ rather than an upper bound on $\mathbf{Adv}^{\text{mac-max}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\rho)$ ([19] top of page 236). In this section, we show how to obtain such an upper bound in two different ways. The first is rather straightforward and yields an upper bound of ordoer $O(q\sigma^2/2^n)$, giving security against adversaries of time complexity 2^{42} (up from 2^{32}). The second method is harder but yields an $O(\sigma^2/2^n)$ upper bound. This is optimal and gives security against adversaries of time complexity 2^{56} (where the dominant term is no longer $\sigma^2/2^n$ but $d\epsilon_{\text{dp}}\frac{q^2}{2}$).

6.1 An $O(q\sigma^2/2^n)$ upper bound on $\mathbf{Adv}^{\text{mac}}_{\mathbf{PC\text{-}MAC}^{\circ}_{d}[E_K,L,G]}(q,\sigma)$

Our starting point is the following analogue of (17), also due to Black and Rogaway [8] (and also cited by MT [19]):

$$\mathbf{Adv}^{\text{vilprf}}_{\text{PC-MAC}^{\star}_{d}[\mathbf{R},G]}(q,\sigma) \leq \max_{m_1,\ldots,m_q,\sum_{r=1}^{q} m_r=\sigma} \left(\sum_{1\leq r<s\leq q} \mathbf{Adv}^{\text{coll}}_{\text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G]}(m_r,m_s) \right) \tag{18}$$

By Lemma 2, thus, we have

$$\begin{aligned} \mathbf{Adv}^{\text{vilprf}}_{\text{PC-MAC}^{\star}_{d}[\mathbf{R},G]}(q,\sigma) \quad \leq \quad & \max_{m_1,\ldots,m_q,\sum_{i=1}^{q} m_i=\sigma} \left(\sum_{1\leq i<j\leq q} \frac{(m_i+m_j)^2}{2^{n+1}} \right) \\ & + \left(d\epsilon_{\text{dp}} + \epsilon_{\text{sdp}} + \frac{1}{2^n} \right) \frac{q^2}{2} \end{aligned} \tag{19}$$

To apply (19) we need to upper bound the "max" term. Note that we can allow the m_i's to range over arbitrary nonnegative real numbers, since this can only increase the max. A straightforward maximization argument then shows that the max is obtained when (say) $m_1 = \sigma$ and $m_i = 0$ for $i > 1$.

We thus directly obtain

$$\mathbf{Adv}^{\mathrm{vilprf}}_{\text{PC-MAC}^{\star}_{d}[\mathbf{R},G]}(q,\sigma) \;\leq\; \frac{q\sigma^2}{2^{n+1}} + \left(d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}} + \frac{1}{2^n}\right)\frac{q^2}{2}$$

which implies

$$\mathbf{Adv}^{\mathrm{vilprf}}_{\text{PC-MAC}^{*}_{d}[R,L,G]}(q,\sigma) \;\leq\; \frac{(\sigma+4d)^2}{2^{n+1}} + \frac{q\sigma^2}{2^{n+1}} + \left(d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}} + \frac{1}{2^n}\right)\frac{q^2}{2}$$

and

$$\mathbf{Adv}^{\mathrm{vilprf}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\sigma) \;\leq\; \mathbf{Adv}^{\mathrm{prp}}_{E}(\sigma+4d) + \frac{1.5(\sigma+4d)^2}{2^{n+1}} + \frac{q\sigma^2}{2^{n+1}} + \left(d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}} + \frac{1}{2^n}\right)\frac{q^2}{2}.$$

Finally, this implies, by (5) (and including time complexities),

$$\mathbf{Adv}^{\mathrm{mac}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\sigma,t) \leq \\ \frac{q}{2^n} + \mathbf{Adv}^{\mathrm{prp}}_{E}(\sigma+4d,t') + \frac{1.5(\sigma+4d)^2}{2^{n+1}} + \frac{q\sigma^2}{2^{n+1}} + \left(d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}} + \frac{1}{2^n}\right)\frac{q^2}{2}$$

where $t' = t + O(\sigma)$. We note the dominant is $\frac{q\sigma^2}{2^{n+1}}$ and, when applied to PC-MAC-AES and $n = 128$, this bound yields security against adversaries of time complexity up to 2^{42} (with $\sigma = q = 2^{42}$).

6.2 An $O(\sigma^2/2^n)$ upper bound on $\mathbf{Adv}^{\mathrm{mac}}_{\mathbf{PC\text{-}MAC}^{\circ}_{d}[E_K,L,G]}(q,\sigma)$

In this Section we give our best bound on the provable security of PC-MAC-AES. Unfortunately, the material in this section is somewhat technical, in the same vein as Lemma 3 of [19].

We use the following analogue of (18), also due to Black and Rogaway [8], which is implicit in their proof of (18):

$$\mathbf{Adv}^{\mathrm{vilprf}}_{\text{PC-MAC}^{\star}_{d}[\mathbf{R},G]}(q,\sigma) \leq \\ \max_{\substack{x_1,\ldots,x_q \in (\{0,1\}^n)^+ \\ \sum_{r=1}^{q} |x_r|/n = \sigma \\ x_r \neq x_s, 1\leq r<s\leq q}} \Pr\left[\bigvee_{1\leq r<s\leq q} \text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G](x_r) = \text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G](x_s)\right] \tag{20}$$

We note that probability in (20) occurs only over the choice of $\mathbf{R}$ which, in the case of $\text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G]$, only affects R and the auxiliary key K_{aux} (through R'''). Thus the underlying probability space is sampled by sampling $\mathbf{R}$, which is equivalent to sampling a pair (R, K_{aux}) uniformly at random from $\mathsf{func}(n,n) \times (\{0,1\}^n)^{4d-1}$.

Fix now a sequence $x_1, \ldots, x_q \in (\{0,1\}^n)^+$ for which the maximum in (20) is obtained. Let $m_r = |x_r|/n$ be the block length of x_r (which is an integer).

Let

$$\mathsf{Coll} = \bigvee_{1\leq r<s\leq q} \text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G](x_r) = \text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G](x_s)$$

be the collision event. We wish to upper bound $\Pr[\mathsf{Coll}]$. The natural approach to upper bounding $\Pr[\mathsf{Coll}]$ would be to apply a union bound, thus using the inequality

$$\Pr[\mathsf{Coll}] \leq \sum_{1\leq r<s\leq q} \Pr[\text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G](x_r) = \text{PC-MAC}^{\bullet}_{d}[\mathbf{R},G](x_s)].$$

However, this union bound leads to a term of type $q\sum_r m_r^2/2^n$, and hence to no improvement over the result of Section 6.1.

Instead, we apply a union bound to a different decomposition of the Coll event. Let $l_r = \lceil \frac{m_r}{d+1} \rceil = \lceil \frac{|x_r|}{n(d+1)} \rceil$. Thus l_r calls to R are made during the computation of PC-MAC$^\bullet_d[\mathbf{R}, G](x_r)$, unless $m_r \equiv 1 \bmod (d+1)$ in which case $l_r - 1$ calls are made to R. Following the notations of Lemma 3 in [19], let Y_i^r be the i-th input of R in (the computation of) PC-MAC$^\bullet_d[\mathbf{R}, G](x_r)$ for $1 \le i \le l_r$ and $1 \le r \le q$ and where, if $m_r \equiv 1 \bmod d+1$, $Y_{l_r}^r = \text{PC-MAC}^\bullet_d[\mathbf{R}, G](x_r)$. Also let $l_{\text{lcp}}^{r,s} \ge 0$ be the largest integer such that the first $(d+1)(l_{\text{lcp}}^{r,s} - 1) + 1$ blocks of x_r and x_s coincide. Thus $Y_i^r = Y_i^s$ with probability 1 for $1 \le i \le l_{\text{lcp}}^{r,s}$.

For $1 \le r \le q$ define the event

$$\mathsf{IC}_r = \left(Y_i^r = Y_j^r \text{ for some } 1 \le i < j \le l_r\right)$$

Here IC stands for "internal collision". Then for $1 \le r < s \le q$ define the event

$$\mathsf{MIC}_{r,s} = \Big(\neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \wedge Y_i^r = Y_j^s \text{ for some } 1 \le i \le l_r, 1 \le j \le l_s \text{ such that } i \ne j \text{ if } i, j \le l_{\text{lcp}}^{r,s}\Big)$$

where MIC stands for "mutual internal collision". Finally for $1 \le r < s \le q$ define the event

$$\mathsf{Coll}_{r,s} = (\neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \wedge \neg\mathsf{MIC}_{r,s} \wedge \text{PC-MAC}^\bullet_d[\mathbf{R}, G](x_r) = \text{PC-MAC}^\bullet_d[\mathbf{R}, G](x_s))\,.$$

Obviously, then,

$$\mathsf{Coll} \implies \bigvee_{1 \le r \le q} \mathsf{IC}_r \vee \bigvee_{1 \le r < s \le q} \mathsf{MIC}_{r,s} \vee \bigvee_{1 \le r < s \le q} \mathsf{Coll}_{r,s}. \tag{21}$$

We shall upper bound $\Pr[\mathsf{Coll}]$ by doing a union bound over the events that appear (21). (The key point which will lead to the final improvement of the bound is that each event IC_r is counted just once in this union bound, as opposed to $q-1$ times (once for each value of $s \ne r$), as occurs in the MT proof.)

Just like equation (6) on page 233 of [19] we have that, for any possible value k of K_{aux},

$$\Pr[\mathsf{IC}_r] = \Pr[\mathsf{IC}_r \mid K_{\text{aux}} = k] \le \binom{l_r}{2}/2^n \le \frac{l_r^2}{2^{n+1}}. \tag{22}$$

To upper bound $\Pr[\mathsf{MIC}_{r,s}]$ and $\Pr[\mathsf{Coll}_{r,s}]$ we define some additional events; the first two are the analogues of the events D_{lcp} and D of [19]:

$$\begin{aligned}
\mathsf{D}_{\text{lcp}}^{r,s} &= \left(Y_i^r \ne Y_j^r \text{ for all } i \ne j, 1 \le i, j \le l_{\text{lcp}}^{r,s}\right) \\
\mathsf{D}^{r,s} &= \left(\neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \wedge Y_i^r \ne Y_j^s \text{ for all } 1 \le i \le l_r, 1 \le j \le l_s \text{ such that } i \ne j \text{ if } i, j \le l_{\text{lcp}}^{r,s}\right) \\
\mathsf{SC}^{r,s} &= \left(l_r > l_{\text{lcp}}^{r,s} \wedge l_s > l_{\text{lcp}}^{r,s} \wedge Y_{l_{\text{lcp}}^{r,s}+1}^r = Y_{l_{\text{lcp}}^{r,s}+1}^s\right) \\
\mathsf{TC}^{r,s} &= \left(Y_i^r = Y_j^s \text{ for some } l_{\text{lcp}}^{r,s} < i \le l_r, l_{\text{lcp}}^{r,s} < j \le l_s, (i,j) \ne (l_{\text{lcp}}^{r,s}+1, l_{\text{lcp}}^{r,s}+1)\right) \\
\mathsf{D}_{\text{lcp}+1}^{r,s} &= \left(Y_i^r \ne Y_j^r \text{ for all } i \ne j, 1 \le i \le \min(l_r, l_{\text{lcp}}^{r,s}+1), 1 \le j \le \min(l_s, l_{\text{lcp}}^{r,s}+1)\right)
\end{aligned}$$

(Thus $\mathsf{D}^{r,s} \implies \mathsf{D}^{r,s}_{\mathrm{lcp}}, \neg\mathsf{IC}_r \implies \mathsf{D}^{r,s}_{\mathrm{lcp}}, \neg\mathsf{IC}_s \implies \mathsf{D}^{r,s}_{\mathrm{lcp}}, \neg\mathsf{IC}_r \implies \mathsf{D}^{r,s}_{\mathrm{lcp}+1}, \neg\mathsf{IC}_s \implies \mathsf{D}^{r,s}_{\mathrm{lcp}+1}$; we can note also that $\mathsf{D}^{r,s} = \neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \wedge \neg\mathsf{MIC}_{r,s}$.)

Since $\mathsf{MIC}_{r,s} \implies (\neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s)$, one can check that

$$\mathsf{MIC}_{r,s} \implies \mathsf{SC}^{r,s} \vee \mathsf{TC}^{r,s}$$

and that, more particularly,

$$\mathsf{MIC}_{r,s} \implies (\mathsf{D}^{r,s}_{\mathrm{lcp}} \wedge \mathsf{SC}^{r,s}) \vee (\neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \wedge \neg\mathsf{SC}^{r,s} \wedge \mathsf{TC}^{r,s}). \tag{23}$$

We have that

$$\Pr[\mathsf{D}^{r,s}_{\mathrm{lcp}} \wedge \mathsf{SC}^{r,s}] \leq \Pr[\mathsf{SC}^{r,s} \,|\, \mathsf{D}^{r,s}_{\mathrm{lcp}}] \leq d\epsilon_{\mathrm{dp}} \tag{24}$$

as argued in the proof of Lemma 4 of [19] (case of a 'type (I) collision'). We then have, since $(\neg\mathsf{IC}_r \vee \neg\mathsf{IC}_s) \implies \mathsf{D}^{r,s}_{\mathrm{lcp}+1}$, that

$$\begin{aligned} & \Pr[\neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \wedge \neg\mathsf{SC}^{r,s} \wedge \mathsf{TC}^{r,s}] \\ = \; & \Pr[\neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \wedge \mathsf{D}^{r,s}_{\mathrm{lcp}+1} \wedge \neg\mathsf{SC}^{r,s} \wedge \mathsf{TC}^{r,s}] \\ \leq \; & \Pr[\mathsf{TC}^{r,s} \wedge \neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \,|\, \mathsf{D}^{r,s}_{\mathrm{lcp}+1} \wedge \neg\mathsf{SC}^{r,s}] \end{aligned}$$

To upper bound the latter inequality, let k be any possible value for K_{aux}; we actually upper bound

$$\Pr[\mathsf{TC}^{r,s} \wedge \neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \,|\, \mathsf{D}^{r,s}_{\mathrm{lcp}+1} \wedge \neg\mathsf{SC}^{r,s} \wedge K_{\mathrm{aux}} = k].$$

Now conditioning on $\mathsf{D}^{r,s}_{\mathrm{lcp}+1} \wedge \neg\mathsf{SC}^{r,s}$ we find that the values $Y^r_{l_{\mathrm{lcp}}+1}$ and $Y^s_{l_{\mathrm{lcp}}+1}$ are distinct and also distinct from the values $Y^r_1, \ldots, Y^r_{l_{\mathrm{lcp}}}$, $Y^s_1, \ldots, Y^s_{l_{\mathrm{lcp}}}$. We now reveal, sequentially, the values $Z^r_{l_{\mathrm{lcp}}+1} = R(Y^r_{l_{\mathrm{lcp}}+1})$ (and hence learn the value $Y^r_{l_{\mathrm{lcp}}+2}$, since K_{aux} is known), the value $Z^r_{l_{\mathrm{lcp}}+2} = R(Y^r_{l_{\mathrm{lcp}}+2})$ (and hence lear the value $Y^r_{l_{\mathrm{lcp}}+3}$), etc, until we learn the value $Y^r_{l_r}$—but we do not reveal the value $Z^r_{l_r}$—and then, likewise, we reveal the values $Z^s_{l_{\mathrm{lcp}}+1} = R(Y^s_{l_{\mathrm{lcp}}+1})$, etc, up until we learn the value $Y^s_{l_s}$. In this process, either the condition $\neg\mathsf{IC}_r$ or $\neg\mathsf{IC}_s$ becomes violated, in which case the event $\mathsf{TC}^{r,s} \wedge \neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s$ does not occur, or else: (i) each Z^r-value that is revealed is independently distributed at uniform and has chance at most $1/2^n$ of triggering the event $\mathsf{TC}^{r,s}$ (by colliding with $Y^s_{l_{\mathrm{lcp}}+1}$), and (ii) or each Z^s-value that we reveal is independently distributed at uniform and has chance at most $(l_r - l_{\mathrm{lcp}})/2^n$ of triggering the condition $\mathsf{TC}^{r,s}$. (Note that Z^r_j (resp. $Z^s_{j'+1}$) induces a permutation on Y^r_{j+1} (resp. $Y^s_{j'+1}$) since K_{aux} is fixed.) Since $(l_r - l_{\mathrm{lcp}} - 1)$ Z^r-values are revealed and $(l_s - l_{\mathrm{lcp}} - 1)$ Z^s-values are revealed, we thus have

$$\begin{aligned} \Pr[\neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \wedge \neg\mathsf{SC}^{r,s} \wedge \mathsf{TC}^{r,s}] \; & \leq \; \Pr[\mathsf{TC}^{r,s} \wedge \neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \,|\, \mathsf{D}^{r,s}_{\mathrm{lcp}+1} \wedge \neg\mathsf{SC}^{r,s} \wedge K_{\mathrm{aux}} = k] \\ & \leq \; (l_r - l_{l_{\mathrm{lcp}}} - 1)\frac{1}{2^n} + (l_s - l_{l_{\mathrm{lcp}}} - 1)\frac{l_r - l_{\mathrm{lcp}}}{2^n} \\ & \leq \; \frac{l_r l_s}{2^n}. \end{aligned} \tag{25}$$

Combining (23), (24), (25) we thus obtain

$$\Pr[\mathsf{MIC}_{r,s}] \leq d\epsilon_{\mathrm{dp}} + \frac{l_r l_s}{2^n}. \tag{26}$$

Finally we need to upper bound $\Pr[\mathsf{Coll}_{r,s}]$. Note that since $\mathsf{D}^{r,s} \iff \neg\mathsf{IC}_r \wedge \neg\mathsf{IC}_s \wedge \neg\mathsf{MIC}_{r,s}$, we have

$$\begin{aligned}\Pr[\mathsf{Coll}_{r,s}] &= \Pr[\mathsf{D}^{r,s} \wedge \text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_r) = \text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_s)] \\ &\leq \Pr[\text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_r) = \text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_s) \,|\, \mathsf{D}^{r,s}]\end{aligned}$$

If $l_r = l_s = l_{\text{lcp}}$ then

$$\begin{aligned}\Pr[\mathsf{Coll}_{r,s}] &\leq \Pr[\text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_r) = \text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_s) \,|\, \mathsf{D}^{r,s}] \\ &= \Pr[\text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_r) = \text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_s) \,|\, \mathsf{D}^{r,s}_{\text{lcp}}] \\ &\leq \max(d\epsilon_{\text{dp}}, \epsilon_{\text{sdp}}, \frac{1}{2^n}) \qquad (27)\end{aligned}$$

(see equation (9) p. 234 of [19]). On the other hand, if $l_r > l_{\text{lcp}}$ or $l_s > l_{\text{lcp}}$, then

$$\Pr[\mathsf{Coll}_{r,s}] \leq \Pr[\text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_r) = \text{PC-MAC}^{\bullet}_d[\mathbf{R}, G](x_s) \,|\, \mathsf{D}^{r,s}] \leq \frac{1}{2^n} \qquad (28)$$

(see equation (8) p. 233 of [19]).

Now, from (26), (27), (28) and the observation that $\Pr[\mathsf{MIC}_{r,s}] = 0$ if $l_r = l_s = l_{\text{lcp}}$, we have

$$\Pr[\mathsf{MIC}_{r,s}] + \Pr[\mathsf{Coll}_{r,s}] \leq d\epsilon_{\text{dp}} + \epsilon_{\text{sdp}} + \frac{1}{2^n} + \frac{l_r l_s}{2^n}. \qquad (29)$$

Thus, combining (21) with (22) and (29) we obtain

$$\begin{aligned}\Pr[\mathsf{Coll}] &\leq \sum_{r=1}^{q} \frac{l_r^2}{2^n} + \sum_{1 \leq r < s \leq q} \left(d\epsilon_{\text{dp}} + \epsilon_{\text{sdp}} + \frac{1}{2^n} + \frac{l_r l_s}{2^n} \right) \\ &\leq \sum_{r=1}^{q} \frac{l_r^2}{2^n} + \sum_{1 \leq r < s \leq q} \frac{l_r l_s}{2^n} + \left(d\epsilon_{\text{dp}} + \epsilon_{\text{sdp}} + \frac{1}{2^n} \right) \frac{q^2}{2}. \qquad (30)\end{aligned}$$

The question thus reduces to computing

$$\max_{l_1, \ldots, l_q : \sum_r l_r = \sigma} \sum_{r=1}^{q} \frac{l_r^2}{2^n} + \sum_{1 \leq r < s \leq q} \frac{l_r l_s}{2^n}. \qquad (31)$$

(The condition $l_1 + \cdots + l_q = \sigma$ is slightly generous, since one can have $l_r < m_r$, but this condition is simple to work with.) Taking, more generally, the max over all nonnegative real values of $l_1, \ldots, l_q$ such that $l_1 + \cdots + l_q = \sigma$, it is easy to see that the max is again obtained when (say) $l_1 = \sigma$ and $l_2 = \cdots = l_r = 0$, in which case the value of the max is $\sigma^2/2^n$. Thus

$$\Pr[\mathsf{Coll}] \leq \frac{\sigma^2}{2^n} + \left(d\epsilon_{\text{dp}} + \epsilon_{\text{sdp}} + \frac{1}{2^n} \right) \frac{q^2}{2}$$

and, by (20),

$$\mathbf{Adv}^{\text{vilprf}}_{\text{PC-MAC}^{\star}_d[\mathbf{R}, G]}(q, \sigma) \leq \frac{\sigma^2}{2^n} + \left(d\epsilon_{\text{dp}} + \epsilon_{\text{sdp}} + \frac{1}{2^n} \right) \frac{q^2}{2}. \qquad (32)$$

This implies, by the same reasoning as at the end of Section 6.1, that

$$\mathbf{Adv}^{\mathrm{vilprf}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\sigma,t) \leq$$
$$\mathbf{Adv}^{\mathrm{prp}}_{E}(\sigma+4d,t') + \frac{1.5(\sigma+4d)^2}{2^{n+1}} + \frac{\sigma^2}{2^n} + \left(d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}} + \frac{1}{2^n}\right)\frac{q^2}{2}. \qquad (33)$$

and

$$\mathbf{Adv}^{\mathrm{mac}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\sigma,t) \leq$$
$$\frac{q}{2^n} + \mathbf{Adv}^{\mathrm{prp}}_{E}(\sigma+4d,t') + \frac{1.5(\sigma+4d)^2}{2^{n+1}} + \frac{q\sigma^2}{2^{n+1}} + \left(d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}} + \frac{1}{2^n}\right)\frac{q^2}{2} \qquad (34)$$

where in both cases $t' = t + O(\sigma)$. If we consider the truncated version of PC-MAC, where π is the truncation length, the term $\frac{q}{2^n}$ in (34) must be replaced $\frac{q}{2^\pi}$.

Applied to the case of PC-MAC-AES, we note that the dominant term in (33) and (34) is $d\epsilon_{\mathrm{dp}}\frac{q^2}{2}$ since in that case $\epsilon_{\mathrm{dp}} = MEDP(4\mathrm{r})$ has (best known) upper bound 2^{-113}. When $d = 1$ this gives security of $\approx 2^{57}$, whereas if $d = 5$ the security is about 2^{56}. (These numbers being time complexities of adversaries, where the time complexity of the adversary is the total block length of its queries.) This is the best security bound we could prove for PC-MAC-AES, and seems satisfactory. Moreover, there are no known forgery attacks of this time complexity (but it is easy to devise forgery attacks of time complexity 2^{64}).

7 Key length improvements

Minematsu and Tsunoo [19] also discuss the possibility of, potentially, deriving the key L from K, in order to achieve only n bits of key instead of $2n$ bits of key—e.g., by putting $L = E_K(0^n)$. They mention that the proof techniques for OMAC (in which precisely only n bits of key are achieved by setting $L = E_K(0^n)$) do not apply in the PC-MAC-AES setting (problems arise due to the use of the simplified 4-round AES function) and, in fact, they show a counterexample in this regard.

However, another possibility, not discussed by Minematsu and Tsunoo, is to derive both K and L from a third n-bit key K^*. Given $K^* \in \{0,1\}^n$ (which is now the private key) we can set, for example, $K = E_{K^*}(0^n)$, $L = E_{K^*} = (1^n)$, and then define PC-MAC-AES as before.

That is, applied to the case of $E =$AES, we define a 1-key scheme 1-PC-MAC-AES$^{\pi,d}_{K^*}$ by

$$\text{1-PC-MAC-AES}^{\pi,d}_{K^*}(x) = \text{1-PC-MAC-AES}^{\pi,d}_{\mathrm{AES}_{K^*}(0^n),\mathrm{AES}_{K^*}(1^n)}(x).$$

Generalizing, we define a scheme 1-PC-MAC$^{\circ}_{d}[E,K^*,G]$ by

$$\text{1-PC-MAC}^{\circ}_{d}[E,K^*,G](x) = \text{PC-MAC}^{\circ}_{d}[E_{E_{K^*}(0^n)}, E_{K^*}(L), G](x)$$

for any blockcipher $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ and any keyed permutation $G : \{0,1\}^{3n} \times \{0,1\}^n \to \{0,1\}^n$.

The security of 1-PC-MAC$^{\circ}_{d}[E,K^*,G]$ is similar to that of PC-MAC$^{\circ}_{d}[E_K,L,G]$:

Proposition 1 *For any blockcipher $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ and any keyed permutation $G : \{0,1\}^{3n} \times \{0,1\}^n \to \{0,1\}^n$ we have*

$$\mathbf{Adv}^{\mathrm{vilprf}}_{\text{1-PC-MAC}^{\circ}_{d}[E,K^*,G]}(q,\sigma,t) \leq \mathbf{Adv}^{\mathrm{vilprf}}_{\text{PC-MAC}^{\circ}_{d}[E_K,L,G]}(q,\sigma,t) + \mathbf{Adv}^{\mathrm{prp}}_{E}(\sigma+4d,t') + \frac{1}{2^n} \qquad (35)$$

where $t' = t + O(1)$.

Proof. Let A be a PRF adversary for 1-PC-MAC$^\circ_d[E, K^*, G]$ making q queries of total length at most σ, and of running time t. We construct a PRP adversary B for E making $\sigma + 4d$ queries to E and of running time $t + O(1)$ as follows. When given oracle access to its n-bit to n-bit oracle F (which may be either E_{K^*} for a random key K^* or a random permutation P), B computes $(K, L) = (F(0^n), F(1^n))$. Then B simulates A on oracle PC-MAC$^\circ_d[E_K, L, G]$, and outputs the same answer as A. We note that

$$\Pr[K^* \xleftarrow{\$} \{0,1\}^n; B^{E_{K^*}} \to 1] = \Pr[K^* \xleftarrow{\$} \{0,1\}^n; A^{\text{1-PC-MAC}^\circ_d[E,K^*,G]} \to 1]$$

whereas

$$\Pr[P \xleftarrow{\$} \mathsf{perm}(n); B^P \to 1] \leq \Pr[(K, L) \xleftarrow{\$} \{0,1\}^{2n}; A^{\text{PC-MAC}^\circ_d[E_K,L,G]} \to 1] + \frac{1}{2^n}$$

since the statistical distance between $(P(0^n), P(1^n))$ and a random pair of n-bit strings is $\frac{1}{2^n}$. Moreover, B makes at most $\sigma + 4d$ queries to E. Thus

$$\begin{aligned}
\mathbf{Adv}^{\text{prp}}_E(\sigma + 4d, t + O(1)) &\geq \mathbf{Adv}^{\text{prp}}_E(B) \\
&= \Pr[K^* \xleftarrow{\$} \{0,1\}^n; B^{E_{K^*}} \to 1] - \Pr[P \xleftarrow{\$} \mathsf{perm}(n); B^P \to 1] \\
&\geq \Pr[K^* \xleftarrow{\$} \{0,1\}^n; A^{\text{1-PC-MAC}^\circ_d[E,K^*,G]} \to 1] \\
&\quad - \Pr[(K, L) \xleftarrow{\$} \{0,1\}^{2n}; A^{\text{PC-MAC}^\circ_d[E_K,L,G]} \to 1] - \frac{1}{2^n} \\
&\geq \mathbf{Adv}^{\text{vilprf}}_{\textit{1-PC-MAC}^\circ_d[E,K^*,G]}(q, \sigma, t) \\
&\quad - \mathbf{Adv}^{\text{vilprf}}_{\textit{PC-MAC}^\circ_d[E_K,L,G]}(q, \sigma, t) - \frac{1}{2^n}
\end{aligned}$$

as desired. □

We similarly have that

$$\mathbf{Adv}^{\text{mac}}_{\textit{1-PC-MAC}^\circ_d[E,K^*,G]}(q, \sigma, t) \leq \mathbf{Adv}^{\text{mac}}_{\textit{PC-MAC}^\circ_d[E_K,L,G]}(q, \sigma, t) + \mathbf{Adv}^{\text{prp}}_E(\sigma + 4d, t') + \frac{1}{2^n} \qquad (36)$$

where $t' = t + O(1)$. The proof is identical to the proof of Proposition 1, with the modification that B outputs 1 or 0 according to whether A succeeds in forging or not.

The issue with such a proposal is indeed not the loss of security but, rather, the loss of efficiency. For example, Black and Rogaway once proposed a similar method for reducing the key length of XCBC from $3n$ bits to n bits [9], but the more efficient methods of Iwata and Kurosawa, that use only one key scheduling of the blockcipher, were eventually preferred [16, 17].

The situtaion with PC-MAC-AES, however, is not quite the same. PC-MAC-AES is not efficient anyway with respect to preprocessing. When $d = 1$, three applications of $E_K(\cdot)$ are necessary to compute the auxiliary key K_{aux}; when $d = 5$, 19 applications of $E_K(\cdot)$ are required (and we envisage using PC-MAC-AES with values of d closer to 5 than to 1). In such a case, the cost of two additional applications of E under a different key seems relatively minor. Moreover, the computation of K and L happens only once for the lifetime of the key: once K and L have been computed, K^* may be forgotten. Thus the memory requirements of the modified scheme are the same: only $2n$ bits of key material need to be kept in memory from one application of the MAC to the next (or $2n + (4d - 1)n$ bits of key material if we wish to also keep the auxiliary key in memory, which is likely). In fact,

the modified scheme offers the added flexibility of keeping only n bits of key material in memory between MAC applications, if this seems desirable, whereas the original scheme requires a minimum of $2n$ bits—but there is no obligation to store K^* in the modified scheme, if one wishes to simply store K and L.

A main advantage of having a smaller key, besides the aesthetic value of minimalism, is the possibility of reusing pre-existing key-exchange protocols and software that are designed or 128-bit keys. Disadvantages include complicating the already fairly involved preprocessing stage of PC-MAC-AES, as well as the (rather minor) loss in efficiency. Whether the advantages outweigh the disadvantages may depend on implementation aspects that I am not qualified to comment on.

8 Cryptanalysis

In this section we briefly review some cryptanalytic results on PC-MAC-AES. Most of the contents of this section is covered by the self-evaluation report [2] in a nearly similar amount of detail.

We note, at the onset, that there exist well-known forgery attacks on PC-MAC-AES of time complexity 2^{64}, and, more generally, on all (deterministic) CBC-like MACs using a 128-bit state. These are all based on finding inner collisions. For example, the classical extension attack of Preneel and Oorschot [15] allows to forge a MAC value by finding two messages x, x' with equal MACs; chances are an inner collision has occured, which will allow the MAC of $x'\|y$ to be computed from the MAC of $x\|y$ (assuming x, x' have the same length).

More powerful attacks than forgery attacks have also been mounted, notably the second preimage attacks of Jia et al. [18]. However, all these attacks have time complexity 2^{64}, namely are based in one way or another on birthday collisions, and it is no big surprise that all security breaks down once birthday-many queries are made. Moreover, as stated, these attacks apply to all CBC-like MACs, such as EMAC, XCBC, TMAC, OMAC, CMAC, PC-MAC and MT-MAC, and are therefore not specifically interesting to PC-MAC-AES (whose provable security is anyway quite inferior to 2^{64}). In the two subsections that follow we briefly outline some attacks that are more specific to PC-MAC-AES.

8.1 Differential cryptanalysis attack of Wang et al.

Wang et al. [20,23] detail some attacks that are applicable to MACs using the (simplified) 4-round AES function. These MACs include Pelican [12], MT-MAC-AES [19] and PC-MAC-AES [19]. The methods involve the use of near inner collisions and impossible differentials to recover an internal state. As it applies to PC-MAC-AES, this attack requires query complexity $2^{85.5}$ and offline time complexity 2^{128}. The attack recovers the full key of PC-MAC-AES (which is nontrivial since PC-MAC-AES has a 256-bit key). Despite its novelty, the attack has completely impractical time bounds and poses no apparent risk to the security of PC-MAC-AES.

We note that Wang et al. made certain corrections to their results in [21]. However, these corrections were not relevant to the attack of PC-MAC-AES.

8.2 Side-channel attack of Biryukov et al.

Biryukov et al. [10] give a side channel attack on Alpha-MAC, a MAC based on the 4-round AES function [11] which was a precursor of PELICAN [12] which is itself a precursor of PC-MAC-AES. Their attack is quite efficient and allows so-called "selective" forgeries, with the exception of a

128-bit suffix of the message—this means the adversary can choose any message $x \in (\P 0^{\circ} 1 \diamondsuit^{128})^{+}$ and forge the MAC of $x \| y$ for an n-bit value that is determined by x. As described, however, this attack applies only to Alpha-MAC, and it indeed does not seem as if the attack transfers to PELICAN [13] (let alone PC-MAC-AES).

9 Conclusion

PC-MAC-AES is a carefully designed MAC which offers an interesting mixture of provable security and efficiency. In a sense it inherits these dual features (separately) from its two closest direct ancestors, TMAC [16] and PELICAN [12].

While we originally found the existing provable security results on PC-MAC-AES to not offer satisfactory levels of security (these securing only against adversaries of time complexity 2^{32}), we found good security (up from 2^{32} to 2^{56}) could be achieved after doing some modifications to the proofs. (At this juncture, we would like to emphasize that all the "hard work" in the security proof is done by Minematsu and Tsunoo [19]; our own contribution, while nontrivial, is very dependent on their original insights.)

While security against adversaries of time complexity 2^{56} is not quite as good as other comprable MACs like OMAC (which achieve closer to 2^{64}), it seems sufficient. Moreover the known (non-side-channel) attacks all have complexity 2^{64}, and it seems probable that 2^{56} is still not the best security provable. In particular, a better bound on the MEDP of the 4-round AES function would immmediately imply a better security bound for PC-MAC-AES.

A main drawback of PC-MAC-AES is the complicated preprocessing stage and the large amount of key material that must be kept in memory. Whether the gain in speed offered by PC-MAC-AES, accrued over many MACs effected with the same key, offsets the disadvantages of the preprocessing stage is something that practitioners will be better able to judge than me. (This might depend on the device, the choice of d, the average number of messages MAC'ed with the same key, etc.) The question of whether the proposed key length reduction to n bits described in Section 7 is "worth it" or not (including the inconvenience of changing the specification) is also likely best left to practitioners to decide.

While the differential cryptanalysis of PC-MAC-AES and related schemes seems to have garnered sufficient attention from leading researchers, some concerns remain (on my part) with respect to side channel attacks, as it is not clear to me whether any researchers in that area have closely considered PC-MAC-AES. It may be good to get a second (short) opinion on this matter from someone more knoweldgeable in the field, e.g▷ one of the authors of [10]. However, it also worth considering that SCA countermeasures devised for AES will quite likely apply to the AES round function as well, and hence be deployable to the simplified 4-round function used by PC-MAC-AES.

10 Acknowledgements

I would like to thank Vincent Rijmen for helpful feedback.

References

[1] NEC Corporation, Specification of Cryptographic Technique PC-MAC-AES, 2010. Available at http://www.cryptrec.go.jp /english/method.html.

[2] NEC Corporation, Self-evaluation Report for PC-MAC-AES, 2010. Available at http://www.cryptrec.go.jp /english/method.html.

[3] NIST FIPS-197, http://csrc.nist.gov/publications/fips/fips197/fips197.pdf

[4] M. Bellare, R. Canetti and H. Krawczyk, Keying hash functions for message authentication, CRYPTO 1996, LNCS vol. 1109 (Springer-Verlag), pp. 1–15.

[5] M. Bellare, O. Goldreich and A. Mityagin, The Power of Verification Queries in Message Authentication and Authenticated Encryption, 2004, http://eprint.iacr.org.

[6] M. Bellare, J. Killian and P. Rogaway, The security of the cipher-block chaining message authentication code, Journal of Computer and System Sciences, vol. 61, no. 3, Dec. 2006, pp. 362–399.

[7] J. Black and P. Rogaway, A blockcipher mode of operation for parrelelizable message authentication, EUROCRYPT 2002, LNCS vol. 2332 (Springer-Verlag), pp. 384–397.

[8] J. Black and P. Rogaway, CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions, J. of Cryptology, vol. 18 (2005), no. 2, pp. 111–131.

[9] J. Black and P. Rogaway, Comments to NIST concerning AES modes of operations: A suggestion for handling arbitrary-length messages with the CBC MAC, *Second modes of operation workshop*, available at http://www.cs.ucdavis.edu/ rogaway.

[10] A. Biryukov, A. Bogdanov, D. Khovratovich and T. Kasper, Collision Attacks on AES-based MAC: Alpha-MAC, *Cryptographic Hardware and Embedded Systems*–CHES 2007, LNCS 4727 (Springer-Verlag), pp. 166–180.

[11] J. Daemen and V. Rijmen, A New MAC Construction ALRED and a Specific Instance ALPHA-MAC, FSE 2005, LNCS 3557 (Springer-Verlag), pp. 1–17.

[12] The PELICAN MAC function, IACR eprint archive (2005), http://eprint.iacr.org/2005/088

[13] J. Daemen and V. Rijmen, Refinement of the ALRED Construction and MAC Security Claims, Information Security vol. 4 September 2010, pp. 149–157.

[14] J. Keliher and J. Sui, Exact maximum expected differential and linear cryptanalysis for two-round Advanced Encryption Standard, IET Information Security, vol. 1, no. 2 (2007), pp. 53–57.

[15] B. Preneel, P. C. van Oorschot, MDx-MAC and Building Fast MACs from Hash Functions, CRYPTO 1995, LNCS 963, pp. 1–14.

[16] T. Iwata and K. Kurosawa, TMAC: Two-key CBC MAC, Topics in Cryptology — CT-RSA 2002, LNCS vol. 2271.

[17] T. Iwata and K. Kurosawa, OMAC: One-key CBC MAC, Fast Software Encryption 2003, LNCS vol. 2887.

[18] K. Jia, X. Wang, Z. Yuan and G. Xu, Distinguishing attacks and second-preimage attacks on the CBC-like MACs, Cryptology eprint archive, report 2008/542.

[19] K. Minematsu and Y. Tsunoo, Provably Secure MACs from Differentially-uniform Permutations and AES-based Implementations, FSE 2006, LNCS 4047 (Springer-Verlag), pp. 226–241.

[20] W. Wang, X. Wang and G. Xu, Impossible Differential Cryptanalysis of Pelican, MT-MAC-AES and PC-MAC-AES, Cryptology ePrint Archive, Report 2009/005, 2009, http://eprint.iacr.org/.

[21] S. Wu, M. Wang and Z. Yuan, A Flaw in the Internal State Recovery Attack on ALPHA-MAC, Cryptology ePrint Archive, Report 2010/160, 2010, http://eprint.iacr.org.

[22] Z. Yuan, K. Jia, W. Wang and X. Wang, Distinguishing and Forgery Attacks on Alred and its AES-based instance ALPHA-MAC, Cryptology ePrint Archive, Report 2008/516, 2008, http://eprint.iacr.org.

[23] Z. Yuan, W. Wang, K. Jia, G. Xu and X. Wang, New Birthday Attacks on Some MACs based on Block Ciphers, CRYPTO 2009, LNCS 5667 (Springer-Verlag), pp. 209–230.