Refinements of the ALRED Construction and MAC Security Claims

Joan Daemen[1] and Vincent Rijmen[2,3]

[1] STMicroelectronics Belgium
joan.daemen@st.com
[2] Dept. of Electrical Engineering/ESAT, K.U.Leuven and IBBT
vincent.rijmen@esat.kuleuven.be
[3] IAIK, Graz University of Technology

Abstract. We present three security claims for iterated MAC functions. Next, we propose ALRED, a construction method for MAC functions based on a block cipher that has provable security in the absence of internal collisions. We apply this construction to AES resulting in two MAC functions: ALPHA-MAC and PELICAN. We provide a model for describing different types of internal collisions in ALRED and provide evidence that the security claims we propose are usable for MAC functions that use the ALRED construction. Finally, we provide a motivation for the security claims that accompany PELICAN.

1 Introduction

Message Authentication Codes (MAC functions) are symmetric primitives, used to ensure authenticity of messages. They take as input a secret key and the message and in some cases a diversifier, and produce as output a short tag.

Basically, there are three approaches for designing MACs. The first approach is to design a new primitive from scratch, as for instance MAA [1], UMAC [2] and Poly1305-AES [3]. This approach allows to optimize the security-performance trade-off. The second approach is to define a new *mode of operation* for existing primitives. In this category, we firstly have numerous variants based on the CBC encryption mode for block ciphers, e.g. CBC-MAC [4], OMAC [5], and RMAC [6]. Secondly, there are the designs based on an unkeyed hash function: NMAC, HMAC [7]. Finally, one can design new MACs using components from existing primitives, e.g. MDx-MAC [8]. The ALRED construction falls in the last category: it builds MAC function using the round transformation of a block cipher.

1.1 Related literature

The ALRED construction and its first instantiations ALPHA-MAC and PELICAN were presented in [9, 10]. Shortly thereafter, the AES-based stream cipher LEX was presented [11]. Interestingly, the designer of LEX arrived independently at a design with striking similarities. After an initialization phase consisting of one application of AES, LEX uses one round of the AES as its state update function. After every update of the internal state, 32 bits of key stream are produced by extracting

4 bytes of the internal state. The positions of these 4 bytes correspond to the positions targeted by the injection layout of Alpha-MAC (see Section 5.1).

Minematsu and Tsunoo extended Alred to produce two provably secure MAC constructions: PC-MAC and MT-MAC [12]. When used with AES, their constructions have the same performance as Alpha-MAC and Pelican, but use more keying material. Simplício et al. based a parallelizable MAC on the Alred construction called MARVIN [13].

Huang et al. construct second preimages and internal collisions for Alpha-MAC when the internal state or the key are known [14]. There are no implications for the security of Alpha-MAC.

Yuan et al. present forgery attacks on Alred and Alpha-MAC in [15] and Wang et al. on Pelican, PC-MAC and MT-MAC in [16]. These attacks all make use of inner collisions. None of these attacks contradict the Alpha-MAC and Pelican security claims.

Biryukov et al. present a side-channel collision attack on Alpha-MAC [17]. Although side-channel attacks are principally a feature of an implementation rather than the MAC function itself, they also introduce a type of differentials that was not considered in the original security analysis of Alpha-MAC. We discuss these differentials in Section 6.

1.2 Structure of this paper

In Section 2, we discuss iterative MAC functions and their Achilles' heel: internal collisions. In Section 3, we improve the security claims given in [9] for iterated MAC functions, addressing the issue of internal collisions. In Section 4, we present Alred, a construction method for block cipher based MAC functions. In Section 5 we illustrate the construction method by two examples: Alpha-MAC and Pelican. In Section 6 we provide a new framework for describing the different types of internal collisions in MAC functions based on the Alred construction. In Section 7 we give a new analysis of the behaviour of the expected maximum collision probability in an idealized case. In Section 8 we motivate the security claims accompanying Pelican. We conclude in Section 9.

2 Iterative MAC functions and their internal collisions

The basic property of a MAC function is that it provides an unpredictable mapping between messages and the tag for someone who does not know the key. We find the following design objectives in [18, Table 9.2]:

1. *Key non-recovery*: the expected complexity of any key recovery attack is of the order of 2^{ℓ_k} MAC function executions.
2. *Computation resistance*: there is no forgery attack with probability of success above $\max(2^{-\ell_k}, 2^{-\ell_m})$.

Here ℓ_k is the key length and ℓ_m the tag length. By forgery one means the generation of a message-tag pair (m, t) using only information on pairs (m_i, t_i) with $m \neq m_i$ for all i.

2.1 Iterative MAC functions

An iterative MAC function operates on a working variable, called the *state*. The message is split up in a sequence of message blocks and after an *initialization* the message blocks are sequentially injected into the state by an *iteration function*. Then a *final transformation* may be applied to the state resulting in the tag. At least one of the initialization, iteration function and final transformation is keyed.

Iterative MAC functions can be implemented in hardware or software with limited amount of working memory, irrespective of the length of the input messages. They have the disadvantage that different messages may be found that lead to the same value of the state before the final transformation. This is called an *internal collision* [19]. More generally, internal collisions are a concern for all practical MAC functions, including hierarchic ones.

2.2 Internal collisions

Internal collisions can be used to perform forgery. Assume we have two messages m_1 and m_2 that result in an internal collision. Then for any string m_3 the two messages $m_1\|m_3$ and $m_2\|m_3$ have the same tag value. So given the tag of any message $m_1\|m_3$, one can forge the tag of the message $m_2\|m_3$. Internal collisions can often be used to speed up key recovery as well [19]. If the number of bits in the state is n, and the iteration function can be modeled as a random transformation, then one can expect to find a collision with about $2^{n/2}$ known pairs.

The presence of internal collisions makes that even the best iterative MAC function cannot fulfill the design objectives given above.

One approach to avoid the upper limit due to the birthday paradox in iterative MAC functions is *diversification*. The MAC function gets a third input that serves to diversify the MAC computation. This third input must be non-repeating or random. The method has several important drawbacks. First of all, not only the tag must be sent along with the message, but also this third parameter, typically doubling the overhead. In case of a random value, a random number generator must be implemented. Moreover, the workload of generating the random value should be taken into account in the performance evaluation of the primitive. In case of a non-repeating value the tag generation device must keep track of a counter. In some cases the randomization mechanism itself introduces subtle flaws [20].

In this paper we adopt another approach to avoid internal collisions: we impose an upper bound on the number of tags that may be generated with a given key. If this upper bound is large enough it does not impose restrictions in actual applications.

3 Security claims

We propose here a set of three orthogonal security claims, that an iterative MAC function should satisfy to be called secure. The claims are formulated in terms of three dimension parameters: the tag length ℓ_m, the key length ℓ_k and the *capacity* ℓ_c. The capacity is the size of the internal memory of an ideal iterative MAC function with equivalent resistance against internal collisions. A similar set of claims were first proposed in [9], but we think that some improvements were needed.

3.1 Tag guessing

Claim 1 *The probability of success of any forgery attack not involving key recovery or internal collisions is* $2^{-\ell_m}$.

This claim is a classical security claim for MAC functions, where we only add some restrictions to reflect what is achievable in an iterated MAC.

3.2 Key recovery

Claim 2 *The probability of success of any key recovery attack with a computational effort equivalent to n MAC function executions is not larger than* $n2^{-\ell_k}$.

This claim is also a classical security claim, used for both encryption algorithms and MAC functions.

3.3 Internal collisions

We propose a claim stating that for an attack consisting of calls to a MAC function instance with some unknown key, the success probability of generating inner collisions shall be lower than that of a generic attack on a MAC function with an internal state of ℓ_c bits, for the same total number of message bits.

Given the total number of message bits A, the generic attack consists of constructing as many different bitstrings possible with total length A, sending these strings as messages to the MAC function instance and observing whether there are messages with colliding tags. For simplicity we assume here that colliding tags imply an internal collision.

The probability of having an internal collision is equal to 1 minus the probability that the $n(A)$ messages lead to $n(A)$ different values in the final value of the internal state. If the size of the internal state is ℓ_c bits, the collision probability is given by $1 - \exp(-n(A)^2/2^{\ell_c+1})$.

Iterated MAC functions typically operate on the message as sequences of blocks of some fixed length ℓ_w, where first padding is applied. Therefore we express our security claim with the total number of message blocks, rather than bits.

Claim 3 *The probability of observing an inner collision in an instance of a MAC function with an unknown key by sending messages to it, where the total number of messages blocks is A, is not above* $1 - \exp(-A^2/2^{\ell_c+1})$.

Note that for $A < 1/4 \times 2^{\ell_c/2}$ we have $1 - \exp(-A^2/2^{\ell_c+1}) \approx A^2/2^{\ell_c+1}$.

In the best case there are no attacks better than generic attacks and the capacity ℓ_c is equal to the number of bits of the state. In [9] this claim was originally formulated as follows:

> *The probability that an internal collision occurs in a set of A ((adaptively) chosen message, tag) pairs, with* $A < 2^{\ell_c/2}$, *is not above* $1 - \exp(-A^2/2^{\ell_c+1})$.

In that formulation, there are problems with iterative MAC constructions based on an invertible iteration function. This can be seen as follows. We consider messages where all message blocks are equal. In a pure iterated construction, the iteration function becomes then a constant permutation, composed of a number of cycles. Each cycle gives rise to fixed points. The maximum length of a cycle of a permutation on a set of 2^n elements equals 2^n. It follows that if we iterate the permutation $2^n!$ times, then we obtain the identity transformation. Hence, a message consisting of a single block x_a and a message consisting of x_a followed by $2^n!$ times a block x_b will always result in an internal collision. Since the original formulation of Claim 3 says nothing about the length of messages, we would have to conclude that the capacity equals zero for any invertible iteration function and for any value of the input.

4 The ALRED construction

We describe here a way to construct a MAC function based on an iterated block cipher. The key length of the resulting MAC function is equal to that of the underlying block cipher. In our presentation, we call ℓ_w the *block length* of the MAC function, not to be confused with the block length of a block cipher used in its construction. For sake of simplicity, we assume that the block length of the block cipher equals the state size n.

4.1 Definition

We distinguish a number of steps. First the message is padded and split in message blocks. After padding, the length of the message must be a multiple of ℓ_w bits, where ℓ_w is a characteristic of a component in the MAC function. The message blocks are applied to a *state* that is initialized using the key and that afterwards undergoes a final step, again using the key.

Message padding and splitting: Pad the message, for instance by means of padding method 2 in [4]: append a single 1 bit followed by the minimum number of 0 bits so that the resulting length is a multiple of ℓ_w bits. Split the result in *message blocks* $x_1, x_2, \ldots x_q$ of ℓ_w bits each.

State initialization:

1. Initialize the state with the all-zero block.
2. Apply the *block cipher* to the state.

Chaining: for each message block perform an iteration:

1. Map the bits of the message block to an *injection input* that has the same dimensions as a sequence of r round keys of the block cipher. We call this mapping the *injection layout*.
2. Apply a sequence of r *block cipher round functions* to the state, with the injection input taking the place of the round keys.

Finalization:

1. Apply the *block cipher* to the state.
2. Truncation: the tag is the first ℓ_m bits of the state.

Let the message blocks be denoted by x_i, the state after iteration i by y_i, the key by k and the tag by z. Let ρ denote the iteration function, which consists of the combination of the injection layout and the sequence of r block cipher round functions. Then we can write:

$$y_0 = \mathrm{Enc}(k, 0) \tag{1}$$

$$y_i = \rho(y_{i-1}, x_i), \quad i = 1, 2, \ldots, q \tag{2}$$

$$z = \mathrm{Trunc}(\mathrm{Enc}(k, y_q)) \tag{3}$$

The construction is illustrated in Figure 1 for the case $r = 1$. With this approach, the design of the MAC function is limited to the choice of the block cipher, the number of rounds per iteration r, the injection layout and ℓ_m. The goal is to choose these such that the resulting MAC function fulfills the security claims for iterated MAC functions for some acceptable values of ℓ_m and ℓ_c.

4.2 Motivation

The block cipher application in the initialization prohibits off-line attacks to construct internal collisions. It also makes the difference propagation through the chaining phase, with its nonlinear iteration function, depend on the key. A remarkable feature of the ALRED construction method is that the iteration function *doesn't* take a secret key as input. In fact, our analysis does not reveal any benefit following from making the iteration function depend on the key.

The computational efficiency of ALRED depends on the length of the message blocks. Where in CBC based constructions for long messages there is one block cipher execution per block, ALRED takes merely r rounds per message block.

Note that for an adversary that has obtained the value of the initial state y_0 for a given key, generating internal collisions typically becomes feasible. Therefore, it is essential that implementations protect the secrecy of state value, e.g., against side channel attacks [17].

4.3 Provability

An ALRED MAC function is as strong as the underlying block cipher with respect to key recovery. *In the absence of internal collisions,* it is resistant against forgery if the block cipher is resistant against ciphertext guessing.

Observation: The proofs we give are valid for any chaining phase that transforms y_0 into y_{final} parameterized by a message. In the proofs we denote this by $y_{\mathrm{final}} = F_{\mathrm{cf}}(y_0, m)$.

This implies that the security of the underlying block cipher protects against forgery without internal collisions, even for the weakest function F_{cf}, e.g., a linear function.

Theorem 1. *Every key recovery attack on* ALRED*, requiring t (adaptively) chosen messages, can be converted to a key recovery attack on the underlying block cipher, requiring $t + 1$ adaptively chosen plaintexts.*

Proof: Let $\mathcal{A}$ be an attack requiring the t tag values corresponding to the t (adaptively) chosen messages m_j, yielding the key. Then, the attack on the underlying block cipher works as follows.

1. Request $c_0 = \mathrm{Enc}(k, 0)$, where '0' denotes the all-zero plaintext block.
2. For $j = 1$ to t, compute $p_j = F_{\mathrm{cf}}(c_0, m_j)$.
3. For $j = 1$ to t, request $c_j = \mathrm{Enc}(k, p_j)$.
4. Input the tag values $\mathrm{Trunc}(c_j)$ to $\mathcal{A}$ and obtain the key.

□

Theorem 2. *Every forgery attack on* ALRED *not involving internal collisions, requiring t (adaptively) chosen messages, can be converted to a ciphertext guessing attack on the underlying block cipher, requiring $t + 1$ adaptively chosen plaintexts.*

Proof: Let $\mathcal{B}$ be an attack, not involving internal collisions, requiring the t tag values corresponding to the t (adaptively) chosen messages m_j yielding a forged tag for the message m. Then, the ciphertext guessing attack on the underlying block cipher works as follows.

1. Request $c_0 = \mathrm{Enc}(k, 0)$, where '0' denotes the all-zero plaintext block.
2. For $j = 1$ to t, compute $p_j = F_{\mathrm{cf}}(c_0, m_j)$.
3. For $j = 1$ to t, request $c_j = \mathrm{Enc}(k, p_j)$.
4. Input the tag values $\mathrm{Trunc}(c_j)$ to $\mathcal{B}$ and obtain the tag for the message m.
5. Compute $p = F_{\mathrm{cf}}(c_0, m)$.
6. If there is a j for which $p = p_j$, then $\mathcal{B}$ has generated an internal collision, which conflicts with the requirement on $\mathcal{B}$. Otherwise, input the tag values $\mathrm{Trunc}(c_j)$ to $\mathcal{B}$ and obtain the tag, yielding the truncated ciphertext of p.

□

5 ALPHA-MAC and PELICAN

We briefly reiterate here two instantiations. They both use AES [21, 22] as underlying block cipher, allowing for keys of 16, 24 or 32 bytes. They both were claimed to satisfy the security claims for iterative MAC functions for the three key lengths of AES with $\ell_m \leq 128$ and $\ell_c = 120$.

5.1 ALPHA-MAC

The ALPHA-MAC iteration function consists of a single round: $r = 1$ [9]. The message block length ℓ_w is equal to 32 bits. The AES round function takes as argument a 16-byte round key, represented in a 4×4 array. The injection layout positions the 4 bytes of the message block $[q_1, q_2, q_3, q_4]$ in 4 positions of this array, resulting in the following injection input:

$$\begin{bmatrix} q_1 & 0 & q_2 & 0 \\ 0 & 0 & 0 & 0 \\ q_3 & 0 & q_4 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4}$$

5.2 PELICAN

The iteration function of PELICAN consists of four rounds ($r = 4$). The message block length ℓ_w is equal to 128 bits. The injection layout copies the message block to the first round key and resets the three remaining round keys to zero. The final block of the padded message is simply XORed to the state prior to the output transformation.

PELICAN has roughly the same performance as ALPHA-MAC, but uses a simpler injection layout. The security analysis of PELICAN can be based on existing security analysis of AES to a much larger extent than is the case for ALPHA-MAC.

6 Internal collisions in ALRED

In order to motivate the claimed ℓ_c of an ALRED design, we need to analyse how difficult it is to construct internal collisions.

6.1 Pairs of message sequences and their colliding probability

When we apply the standard definitions and results from differential cryptanalysis for block ciphers on ALRED constructions, it is easy to get confused, because of the mismatch of inputs. Namely, the message blocks that are to be MACed, are applied via the "key input" of the round transformation of the block cipher, and the state applied via the "message input" of the round transformation, depends on the MAC key, which is unknown to the attacker (see Figure 2).

Therefore, we propose to discuss internal-collision based attacks on ALRED constructions using dedicated terminology: *pair of message sequences (PMS)* and their collision probability. Let

$$m = a_0 || \dots || a_{t-1}$$
$$m^* = a_0^* || \dots || a_{u-1}^*$$

denote a PMS with members of length t and $u \leq t$, respectively. A state y produces a collision for the PMS (m, m^*) if and only if

$$\rho(\rho(\rho(y, a_0), \dots), a_{t-1}) \oplus \rho(\rho(\rho(y, a_0^*), \dots), a_{u-1}^*) = 0. \tag{5}$$

We call the left-hand member of this equation the *characteristic function* of the PMS. The *collision number (CN)* of the PMS (m, m^*) is defined as the number of states y that satisfy (5). The *collision probability (CP)* is defined by $\mathrm{CP} = 2^{-n}\mathrm{CN}$.

When all states y occur with the same probability, the CP of a PMS corresponds to the probability that the PMS will cause an internal collision. Note that in a practical scenario, the value of y depends on the MAC key, and a possible prefix common to both messages. Since the value of y_0 is obtained by $y_0 = \mathrm{Enc}(k, 0)$, typically y_0 does not have a uniform distribution. However, for a secure block cipher it is generally believed to be hard to determine the probability of any given value of y_0. Hence, making the assumption that y_0 has a uniform distribution, is a commonly accepted practice. Clearly, both concepts PMS and CP are strongly related to concepts used in the differential cryptanalysis of block ciphers.

6.2 Types of internal collisions

Internal collisions can be produced by different types of PMS. Firstly, a t-round fixed point can be seen as a degenerate PMS where the second sequence is empty ($u = 0$). A second type of PMS consists of two sequences of the same length ($t = u > 1$) with first and last block different and all intermediate blocks equal. This type is called *extinguishing differential* in [9].

The third type of PMS consists of all other possibilities. Either two sequences of different nonzero length ($t > u > 0$) or of equal length but with at least one intermediate block different between the two members. An example of the first are *4-1-collisions* for Alpha-MAC, i.e. PMS with $t = 4$ and $u = 1$ [17]. The existence of such message pairs follows from:

Corollary 1 ([9, Corollary 1]). *Given y_{i-1}, the state value before iteration i, the map*

$$s : (x_i, x_{i+1}, x_{i+2}, x_{i+3}) \rightarrow y_{i+3}$$

from the sequence of four message blocks $(x_i, x_{i+1}, x_{i+2}, x_{i+3})$ to the state value before iteration $i + 4$ is a bijection.

Let $a = y_0$ denote the state after the initialization phase and x_1 an arbitrary selected single-block message. Let

$$b = y_1 = \rho(y_0, x_1).$$

According to Corollary 1, for any two states a, b, there exists a sequence of four message blocks $(x_1^*, x_2^*, x_3^*, x_4^*)$ such that

$$b = \rho(\rho(\rho(\rho(a, x_1^*), x_2^*), x_3^*), x_4^*).$$

Hence x_1 and $(x_1^*, x_2^*, x_3^*, x_4^*)$ form a 4-1-collision. Setting $a = b$ and removing x_1 from the previous analysis demonstrates the existence of 4-block fixed points.

We conclude that the security analysis of Alpha-MAC given in [9] is incomplete. In the meanwhile, Pelican has been introduced. We will therefore not try to complete the security analysis of Alpha-MAC, but instead concentrate our efforts on the new design.

7 Expected maximum CP

In this section, we derive the distribution of the maximum CP of pairs of message sequences when the iteration function f is a uniformly distributed random permutation and the initial value of the state is a uniformly distributed random variable.

We make the derivation for three particular cases: a two-block extinguishing differential, a single-block fixed point and a PMS consisting of a 2-block member and a 3-block member. Subsequently we generalize and draw conclusions.

7.1 Two-block extinguishing differential

We start with the case of two-block extinguishing differentials, because it is the most similar to a differential in a block cipher. The members of the PMS are $a_0||a_1$ and $a_0^*||a_1^*$. The characteristic function of this PMS is:

$$g(y) = f(y \oplus a_0) \oplus f(y \oplus a_0^*) \oplus a_1 \oplus a_1^* \ .$$

The CN of $(a_0||a_1, a_0^*||a_1^*)$ is given by:

$$\mathrm{CN} = \#\{y|g(y) = 0\} \ . \tag{6}$$

By a change of variable $y' = y \oplus a_0$, the characteristic function can be converted to

$$g(y') = f(y') \oplus f(y' \oplus a_0 \oplus a_0^*) \oplus a_1 \oplus a_1^* \ .$$

It follows that the CN only depends on the initial difference $a_0 \oplus a_0^*$ and the final difference $a_1 \oplus a_1^*$. This characteristic function has a symmetry property: $g(y \oplus a_0 \oplus a_0^*) = g(y)$. It follows that the CN of an extinguishing differential is always even. In fact, computing the distribution of the CP of this PMS corresponds to computing the distribution of the differential probability of the differential $(a_0 \oplus a_0^*, a_1 \oplus a_1^*)$ over f. If $a_0 \neq a_0^*$ and f is a random transformation, then CN/2 has a binomial distribution [23, 24]. The Poisson distribution with $\lambda = 1/2$ is a good approximation for this distribution. When f is a random *permutation*, the different y values are not independent. However, if $a_1 \neq a_1^*$, then this difference can be ignored for all practical values of n [23]. We now consider the distribution of

$$\max_{a_0 \oplus a_0^* \neq 0, a_1 \oplus a_1^* \neq 0} \mathrm{CN}(a_0||a_1, a_0^*||a_1^*).$$

If we assume that the CN values are independent, then this is the distribution of the maximum of $(2^n - 1)^2 \approx 2^{2n}$ variables (namely, all combinations of nonzero initial differences and final differences), that all have the same Poisson distribution.

For typical values of λ and n, the maximum has a very narrow distribution, with only non-zero probabilities in two CN values near its expected value [CN]. The expected value of the maximum CN is $[\mathrm{CN}] = 2Z$, with Z given by the following equation [24]:

$$Z = \frac{\ln(2)y - \frac{1}{2}\ln(2\pi Z) - \lambda}{\ln(\frac{Z}{\lambda}) - 1}, \tag{7}$$

where y is the binary logarithm of the total number of variables. For the case considered here we have $y = 2n$ and $\lambda = 1/2$.

7.2 Single-block fixed point

The first member of the PMS consists now of a single block a_0, while the second member is the empty string. We have a fixed point if $f(y \oplus a_0) = y$. The characteristic function $g(y)$ is given by $g(y) = f(y \oplus a_0) + y$. An important difference with the previous case is that here the characteristic function doesn't have a symmetry. The distribution of the CP and of the maximum of the CP over all non-zero values of a_0 can be studied in a way very similar to the first case.

The expected value of the maximum of the CN over all values of a_0 can again be obtained by solving (7), now with $[\mathrm{CN}] = Z$, $y = n$ and $\lambda = 1$.

7.3 A simple asymmetric PMS

As a final example, we consider a PMS with members $a_0||a_1||a_2$ and $a_0^*||a_1^*$. The characteristic function of this PMS is

$$g(y) = f(f(y \oplus a_0) \oplus a_1) \oplus f(y \oplus a_0^*) \oplus a_2 \oplus a_1^* \ .$$

By a change of variables we can convert this to

$$g(y) = f(f(y) \oplus a_1) \oplus f(y \oplus (a_0 \oplus a_0^*)) \oplus (a_2 \oplus a_1^*) \ .$$

and hence the CN depends only on three variables: the initial difference $a_0 \oplus a_0^*$, the intermediate block a_1 and the final difference $a_2 \oplus a_1^*$. This characteristic function is not symmetric in y. The expected value of the maximum of the CN over all values of the initial difference, the intermediate block and the final difference can be obtained by solving (7), with $y = 3n$ and $\lambda = 1$. For asymmetric PMS, we have $[\mathrm{CN}] = Z$.

7.4 Summary

The expected value of the maximum CN value over all PMS with members of given lengths t, u depends only on these lengths and the fact whether the PMS is an extinguishing differential or not. We have:

- Extinguishing differentials ($t = u > 1$): $y = t\ell_w$ and $\lambda = 1/2$. The degrees of freedom are the initial difference $a_0 \oplus a_0^*$, the final difference $a_{t-1} \oplus a_{t-1}^*$ and the intermediate values $a_i = a_i^*$.
- Fixed points ($t > u = 0$): $y = t\ell_w$ and $\lambda = 1$. The degrees of freedom are the t values of a_0 to a_{t-1}.
- Other cases ($t \geq u > 1$): $y = (t + u - 2)\ell_w$ and $\lambda = 1$. The degrees of freedom are the initial difference $a_0 \oplus a_0^*$, the final difference $a_{t-1} \oplus a_{u-1}^*$, the $t - 2$ intermediate values a_1 to a_{t-2} and the $u - 2$ intermediate values a_1^* to a_{u-2}^*. (Note that the case $t > u = 1$ can be reduced to a $t - 1$-block fixed point and that the case $t = u = 1$ cannot result in an internal collision.)

Using equation (7) we can now compute the maximum collision probability for PMS with members of given length $t + u$. We count the cost of applying a PMS by the number of blocks it contains. Then given an attack budget of a fixed number of blocks A, the success probability is given by the total number of pairs $A/(t + u)$ times the collision probability CP of the PMS. So, when deciding whether to take a PMS with a short or large length, the collision probability divided by the length of the PMS is relevant. Figure 3 plots the expected maximum collision probability [CN] (scaled by a factor 2^{128}) divided by the number of blocks of PMS as a function of the total number of blocks of the PMS for the case of 128-bit blocks. Clearly the best choice is to use fixed points and extinguishing differentials of a small number of blocks.

In this attack scenario we only consider internal collisions between pairs of messages that exhibit the given PMS. In general, when applying N messages, for each of the $N(N-1)/2$ pairs of messages among them, there may be an internal collision. This also favors short PMS because this allows to have more messages and hence more pairs.

8 The collision-resistance of Pelican

We claim that Pelican has an internal-collision resistance corresponding to $\ell_c = 120$. In this section we motivate this choice. Nicer would be to be able to *prove* this claim, but we believe that this would require an extensive additional research effort.

In our opinion, two-block extinguishing differentials are the only case where the structural properties of the iteration function are relevant. When PMS become longer, the structural properties tend to become less important. In asymmetric PMS, including fixed points, structural properties are also less important. Hence we will argue here that there exist no two-block extinguishing differentials with DP $\geq 2^{-119}$.

The CP of a two-block extinguishing differential with initial difference a' and final difference b' is the same as the DP of the differential (a', b') over the sequence of 4 rounds of AES with all-zero round keys. No bounds are known for this DP. There are bounds on the EDP of 4-round AES differentials [25], but they are not tight. Based on our results in [26], we estimate that the maximum EDP over four rounds is smaller than 4×2^{-128}.

The DP value of a differential (a', b') is the sum of the DP values of the trails with input difference a' and output difference b'. A differential trail Q over 4 rounds of AES has $\mathrm{EDP}(Q) \leq 2^{-150}$ [22]. A plateau trail with *height* h has 2^h right pairs for a fraction $\mathrm{EDP}2^{128-h}$ of the expanded keys and zero right pairs otherwise. Almost all trails over 4 rounds of AES are *plateau trails*, where the vast majority have $h = 1$ [27, 28].

As a first approximation, assume that all trails have $h = 1$. Then, making some standard independence assumptions, we obtain that the distribution of the DP values of a differential (a', b') can be approximated by a Poisson distribution with $\lambda = 2^{127}\mathrm{EDP}(a', b')$. The maximum over large sets of such differentials with some given EDP can be estimated using (7). If we make the approximation that all 2^{256} differentials have the estimated EDP value 4×2^{-128}, then we obtain an estimate for the maximum DP, that we believe to be on the safe side. Solving (7) with $\lambda = 2$ results in a DP value of $136 \cdot 2^{-128} \approx 2^{-121}$.

The trails with $h > 1$ increase the maximum DP of differentials compared to what is predicted by the Poisson distribution. For two-round trails, we proved that heights up to 5 occur [27]. The maximal height of trails over larger numbers of rounds remains a topic of further research. As long as the increase caused by deviations from the Poisson distribution results in a DP increase less than $2^{-119} - 136 \cdot 2^{-128} = 376 \cdot 2^{-128}$, the capacity claim remains upright.

9 Conclusions

We conclude by summarizing the main advantages of the Alred construction in general, and Pelican in particular.

Performance: One iteration of Pelican corresponds roughly to 4 rounds of AES, hence roughly 4/10 of an AES-128 encryption. It is actually better because the XORs with 0 in the round key addition don't have to be implemented. Hence, for long messages Pelican is more than a factor 2.5 faster than AES-128 encryption.

Style: The construction is simple, and doesn't require much more than calls to the round transformation of the block cipher. A software implementation of Pelican benefits from hardware accelerated AES instructions. DPA countermeasures implemented for AES can be used to protect Pelican.

Provability: The construction comes with security proofs, *without* relying on nonces.

Acknowledgements

Vincent Rijmen is sponsored by the Research Fund K.U.Leuven (OT/08/027), by the IAP Programme P6/26 BCRYPT of the Belgian State (Belgian Science Policy) and by the European Commission through the ICT Programme under Contract ICT-2007-216676 (ECRYPT II). The information in this paper is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

References

1. Donald W. Davies, "A message authenticator algorithm suitable for a mainframe computer," Advances in Cryptology – Proceedings of Crypto '84, LNCS 196, G. R. Blakley and D. Chaum, Eds., Springer-Verlag, 1985, pp. 393–400.
2. John Black, Shai Halevi, Hugo Krawczyk, Ted Krovetz, Phillip Rogaway, "UMAC: Fast and Secure Message Authentication," Advances in Cryptology – Crypto '99, LNCS 1666, M.J. Wiener, Ed., Springer-Verlag, 1999, pp. 216–233.
3. Daniel J. Bernstein, "The Poly1305-AES message-authentication code," Fast Software Encryption 2005, LNCS 3557, H. Gilbert, H. Handschuh, Eds., Springer-Verlag, 2005, pp. 32–49.
4. ISO/IEC 9797-1, Information technology - Security Techniques - Message Authentication Codes (MACs) - Part 1: Mechanisms using a block cipher, ISO 1999.
5. Tetsu Iwata and Kaoru Kurosawa, "OMAC: One-key CBC MAC," Fast Software Encryption 2003, LNCS 2887, T. Johansson, Ed., Springer-Verlag, 2003, pp. 129–153.
6. E. Jaulmes, A. Joux and F. Valette, "On the security of randomized CBC-MAC beyond the birthday paradox limit: A new construction," Fast Software Encryption 2002, LNCS 2365, J. Daemen and V. Rijmen, Eds., Springer-Verlag, 2002, pp. 237–251.
7. Mihir Bellare, Ran Canetti and Hugo Krawczyk, "Keying hash functions for message authentication," Advances in Cryptology - Crypto 96, LNCS 1109, N. Koblitz, Ed., Springer-Verlag, 1996, pp. 1–15.
8. Bart Preneel and Paul C. van Oorschot, "MDx-MAC and building fast MACs from hash functions," Advances in Cryptology, Proceedings Crypto'95, LNCS 963, D. Coppersmith, Ed., Springer-Verlag, 1995, pp. 1–14.
9. Joan Daemen and Vincent Rijmen, "A new MAC Construction Alred and a Specific Instance Alpha-MAC," Fast Software Encryption 2005, LNCS 3557, H. Gilbert, H. Handschuh, Eds., Springer-Verlag, 2005, pp. 1–17.
10. Joan Daemen and Vincent Rijmen, "The Pelican MAC function," Cryptology ePrint Archive, Report 2005/088, 2005.
11. Alex Biryukov, "A new 128-bit key stream cipher LEX," submission to the eSTREAM project, revised version, 2006, http://www.ecrypt.eu.org/stream/lexp3.html
12. Kazuhiko Minematsu, Yukiyasu Tsunoo, "Provably secure MACs from differentially-uniform permutations and AES-based implementations," Fast Software Encryption 2006, LNCS 4047, M. Robshaw, Ed., Springer-Verlag, 2006, pp. 226–241.
13. Marcos Simplício Jr., Pedro d'Aquino Barbuda, Paulo Barreto, Tereza Carvalho and Cintia Margi, "The MARVIN message authentication code and the LETTERSOUP authenticated encryption scheme", Security and Communication Networks, Vol. 2, No. 2, 2009, pp. 165–180.

14. Jianyong Huang, Jennifer Seberry and Willy Susilo, "On the internal structure of Alpha-MAC," Vietcrypt 2006, LNCS 4341, P.Q. Nguyen, Ed., Springer-Verlag, 2006, pp. 271-285.
15. Zheng Yuan, Keting Jia, Wei Wang and Xiaoyun 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/.
16. Wei Wang, Xiaoyun Wang and Guangwu Xu, "Impossible differential cryptanalysis of PELICAN, MT-MAC-AES and PC-MAC-AES," Cryptology ePrint Archive, Report 2009/005, 2008, http://eprint.iacr.org/.
17. Alex Biryukov, Andrey Bogdanov, Dmitry Khovratovich and Timo Kasper, "Collision attacks on AES-Based MAC: Alpha-MAC," Cryptographic Hardware and Embedded Systems - CHES 2007, LNCS 4727, P. Paillier, I. Verbauwhede, Eds., Springer-Verlag, 2007, pp. 166-180.
18. Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.
19. Bart Preneel and Paul C. van Oorschot, "On the security of iterated Message Authentication Codes," IEEE Trans. on Information Theory, Vol. IT-45, No. 1, 1999, pp. 188–199.
20. Lars R. Knudsen and Chris J. Mitchell, "Partial key recovery attack against RMAC," Journal of Cryptology, Vol. 18, No. 4, 2005, pp. 375–389.
21. Federal Information Processing Standard 197, Advanced Encryption Standard (AES), National Institute of Standards and Technology, U.S. Department of Commerce, November 2001.
22. Joan Daemen and Vincent Rijmen, The design of Rijndael — AES, The Advanced Encryption Standard, Springer-Verlag, 2002.
23. Luke O'Connor, "On the Distribution of Characteristics in Bijective Mappings," Advances in Cryptology, Proceedings of Eurocrypt '93, LNCS 765, T. Helleseth, Ed., Springer-Verlag, 1993, pp. 360–370.
24. Joan Daemen and Vincent Rijmen, "Probability distributions of correlation and differentials in block ciphers," Journal of Mathematical Cryptology, No. 1, Springer-Verlag, 2007, pp. 221-242.
25. Liam Keliher and Jerry 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.
26. Joan Daemen, Mario Lamberger, Norbert Pramstaller, Vincent Rijmen, Frederik Vercauteren, "Computational aspects of the expected differential probability of 4-round AES and AES-like ciphers," Computing, Vol. 85, No. 1-2, Springer-Verlag, April 2009, pp. 85-104.
27. Joan Daemen and Vincent Rijmen, "Plateau characteristics and AES," IET Information Security, Vol. 1, No. 1, March 2007, pp. 11–17.
28. Joan Daemen and Vincent Rijmen, "New Criteria for Linear Maps in AES-like Ciphers," Cryptography and Communications, Vol. 1, No. 1, Springer-Verlag, 2008, pp. 47–69.

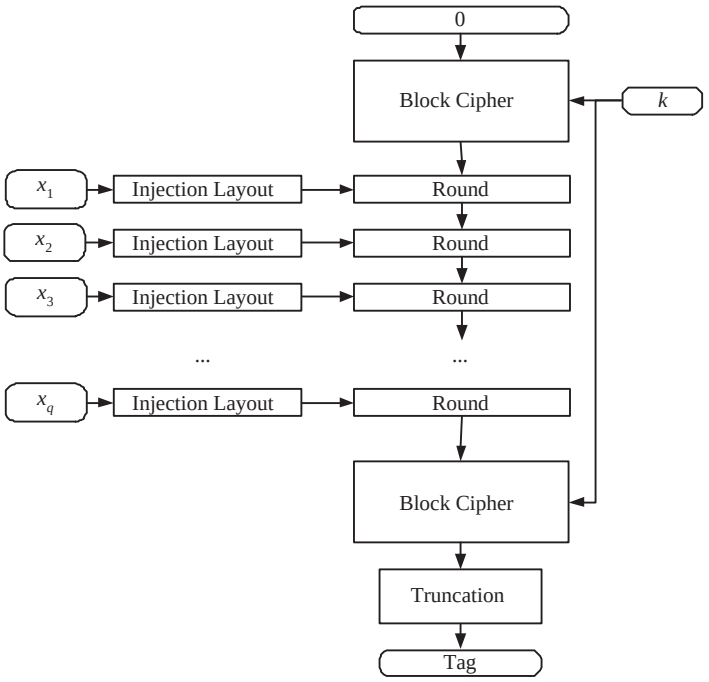

Fig. 1. Scheme of the ALRED construction with $r = 1$.

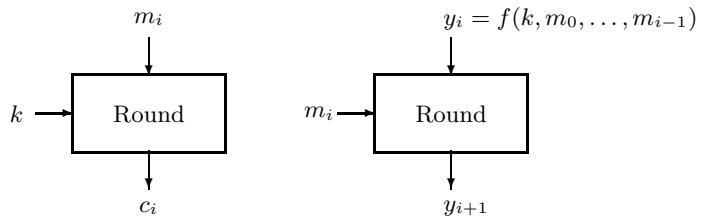

Fig. 2. Inputs of the round transformation in a block cipher (left) and an ALRED construction (right).

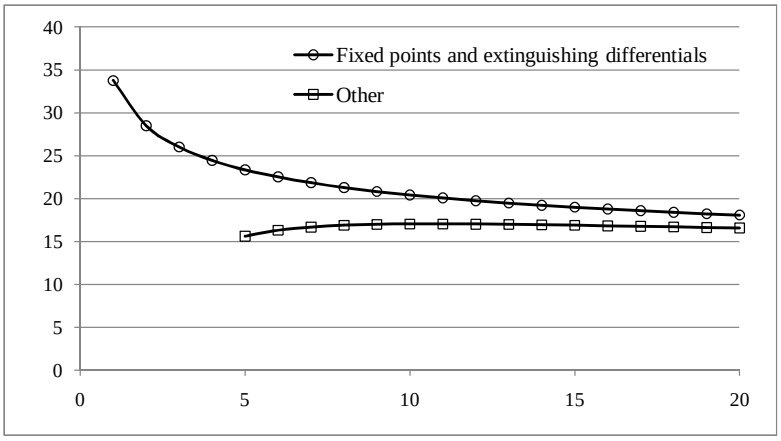

Fig. 3. [CN] per 128-bit block as a function of PMS length