Cryptanalysis of a Message Authentication Code due to Cary and Venkatesan

Simon R. Blackburn and Kenneth G. Paterson*

Department of Mathematics
Royal Holloway, University of London
Egham, Surrey, TW20 0EX, U.K.
simon.blackburn@rhul.ac.uk, kenny.paterson@rhul.ac.uk

Abstract. A cryptanalysis is given of a MAC proposal presented at CRYPTO 2003 by Cary and Venkatesan. A nice feature of the Cary-Venkatesan MAC is that a lower bound on its security can be proved when a certain block cipher is modelled as an ideal cipher. Our attacks find collisions for the MAC and yield MAC forgeries, both faster than a straightforward application of the birthday paradox would suggest. For the suggested parameter sizes (where the MAC is 128 bits long) we give a method to find collisions using about $2^{48.5}$ MAC queries, and to forge MACs using about 2^{55} MAC queries. We emphasise that our results do not contradict the lower bounds on security proved by Cary and Venkatesan. Rather, they establish an upper bound on the MAC's security that is substantially lower than one would expect for a 128-bit MAC.

Keywords: Message authentication, MAC, matrix groups, cryptanalysis, birthday paradox.

1 Introduction

This paper is concerned with a proposal for a Message Authentication Code (MAC) presented at CRYPTO 2003 by Cary and Venkatesan [1]. Their idea is to take a MAC construction of Jakubowski and Venkatesan [3] based on linear operations over a finite field, and alter it by replacing finite field operations by operations in the ring of integers modulo some power of 2 (as the latter operations are more efficient on the current generation of processors). Cary and Venkatesan [1] have proved a lower bound on the security of their MAC. This paper presents two attacks on the MAC, and so establishes a corresponding upper bound on the MAC's

* This author supported by the Nuffield Foundation NUF-NAL 02.

security. The first attack shows that an adversary with access to a MAC oracle is able to find collisions of the MAC considerably faster than a straightforward application of the birthday paradox would suggest. For an introduction to MACs and birthday attacks on them, see [4]. The second attack does more: it derives most of the secret key material for the MAC (which enables MACs to be forged). This second attack works by exploiting certain collisions in the MAC; these collisions are found almost as efficiently as in the first attack. Thus the proposal of Cary and Venkatesan [1], while efficient and offering some interesting provable security properties, does not offer the level of security that a MAC of its output length should aspire to.

The next section describes the Cary–Venkatesan MAC. Sections 3 and 4 describe our two attacks on this MAC. The concluding section explains how our attacks impact on the practical level of security offered by the MAC when the suggested parameter sizes are used.

2 The Cary–Venkatesan MAC

Let ℓ, k and t be integers. The Cary–Venkatesan MAC operates on blocks consisting of t words $x_1, x_2, \ldots, x_t$ each word being of length ℓ bits. We regard the words x_i as ℓ-bit integers. The MAC has a $t(\ell-1)+k$-bit secret key. This key is made up of odd ℓ-bit integers $a_1, a_2, \ldots, a_t$ together with a k-bit string K.

The MAC consists of two parts, a *compression function* H and a *block cipher* E. The compression function H takes as input the vector $\underline{a} = (a_1, a_2, \ldots, a_t)$ and a block $x = x_1 x_2 \ldots x_t$ where $x_i \in \{0, 1, \ldots, 2^\ell - 1\}$; it returns a 4ℓ-bit string $h = H_{\underline{a}}(x)$. The block cipher operates on 4ℓ-bit blocks. It takes as input the key K and the output h of the compression function; the cipher returns the 4ℓ-bit value $E_K(h)$ and this is the output of the MAC.

Cary and Venkatesan allow the block cipher E to be any secure block cipher acting on 4ℓ-bit blocks with a k-bit key. They model E as an ideal cipher and concentrate their efforts on designing an efficient compression function H of the following form.

Let $A_1, A_2, \ldots, A_{t-1}$ be fixed 2×2 matrices, and let z_0 and σ_0 be fixed column vectors of length 2; suppose all the entries of these matrices and vectors lie in the ring $\mathbb{Z}_{2^\ell}$ of integers modulo 2^ℓ. (These matrices and vectors are public, and some suggested examples are given in [1].) Vectors $v_1, v_2, \ldots, v_t \in (\mathbb{Z}_{2^\ell})^2$ are calculated as follows. Let $i \in \{1, 2, \ldots, t\}$ be fixed. Multiply the ℓ-bit integers a_i and x_i, to produce a 2ℓ-bit integer;

this product is then broken into two ℓ-bit integers, and the result v_i is regarded as an element of $(\mathbb{Z}_{2^\ell})^2$. The way in which the product $a_i x_i$ is split to form v_i is not specified in [1]; we assume that a natural choice of $v_i = \left[a_i x_i \bmod 2^\ell,\, a_i x_i \operatorname{div} 2^\ell\right]^T$ is used. (Another natural choice would be $v_i = \left[a_i x_i \operatorname{div} 2^\ell,\, a_i x_i \bmod 2^\ell\right]^T$. Our results are unaffected if this choice is used instead.)

The output h of H is defined to be the pair (z, σ), where

$$z = z_0 + v_1 + A_1 v_2 + A_1 A_2 v_3 + \cdots + A_1 A_2 \cdots A_{t-1} v_t$$

and where

$$\sigma = \sigma_0 + v_1 + v_2 + \cdots + v_t.$$

Here all operations are over $\mathbb{Z}_{2^\ell}$.

Cary and Venkatesan propose two variants of their MAC: a way of chaining the compression function so that it can compress more than one block into 4ℓ-bits (by making the 'initial values' z_0 and σ_0 used in the next block depend on z and σ above), and a method for doubling the length of output of the compression function (by computing the compression function above twice on the same block, using different keys, and then concatenating their outputs). Our attacks below can be adapted to apply to these variants as well, although we will not discuss the straightforward modifications that are needed.

3 The first attack

For MACs such as the one considered here, which consist of a relatively weak keyed compression function followed by a block cipher encryption, it is generally assumed that it is computationally infeasible to invert the block cipher E without knowledge of the secret key K. (If the block cipher can be inverted efficiently, the output of the compression function H is available to the cryptanalyst. The keys used in the compression function can then usually be derived from MACs of a few chosen messages. Once these keys are known, MACs of a wide variety of messages may be forged. This shows that, in practice, the security level offered by a MAC of this type cannot be greater than the length of the block cipher key. This is certainly the case with the proposal of [1].)

The final cipher E is often modelled as an ideal cipher, namely a set of random permutations indexed by the key K. An adversary has access to an oracle that adds MACs to messages; the adversary aims to generate a valid MAC for any message that has not been sent as a query to the oracle. In this ideal cipher model it is intuitively clear (and indeed formally provable) that finding two messages that are compressed by H to the same value h (finding a *collision*) is a prerequisite to breaking the MAC. Since the block length of the cipher E in the proposal of [1] is 4ℓ bits, the birthday paradox implies that a collision will be found with reasonable probability after approximately $2^{2\ell}$ oracle queries on random messages. A good scheme in this model should therefore have the property that it is impossible for an adversary to produce collisions with reasonable probability unless the number of oracle queries is close to this upper bound. However, we show that when the MAC proposed in [1] is used, an adversary can produce collisions using significantly fewer than $2^{2\ell}$ messages.

The basic idea of our collision finding attack is to construct a large set of messages with the property that the compression function H maps each message into a small subset of inputs to the block cipher (irrespective of the key $a_1, a_2, \ldots, a_t$ used). Because the block cipher is a permutation, there is a collision in the MAC output if and only if there is a collision at the input to the block cipher, i.e. at the output of the compression function H.

Let r be an integer that is as large as possible subject to the conditions that $0 \leq r < \ell$ and that $2^{t(\ell-r)}$ is at least $n = \lceil \sqrt{2^{4\ell-r}} \rceil$. (For most choices of parameters, $r = \ell - 1$ or $r = \ell - 2$ will suffice.)

There are $2^{t(\ell-r)}$ messages $x = x_1 x_2 \cdots x_t$ with the property that 2^r divides x_i for all $i \in \{1, 2, \ldots, t\}$. We denote this set of messages by X and let Y be a subset of X consisting of n such messages: a subset of this size exists, by our choice of r. We obtain the MACs of all the messages in Y from the MAC oracle. Define B to be the set of all pairs $(z, \sigma) \in ((\mathbb{Z}_{2^\ell})^2)^2$ such that the first component of $\sigma - \sigma_0$ is divisible by 2^r. Our condition on the elements x_i for messages in X implies that the first component of each vector v_i is divisible by 2^r, and so the same is true for the vector $\sigma - \sigma_0$. Hence the image of X under H lies in B, whatever the value of the secret key $a_1, a_2, \ldots, a_t$. Now,

$$|B| = 2^{4\ell-r} < n^2.$$

Since we have requested the MACs of the n messages in the set $Y \subseteq X$, the birthday paradox implies that we will find a collision with reasonable probability (in fact, with probability about 0.63). Note that n is considerably less than $2^{2\ell}$. Indeed, when $r = \ell - 1$ or $r = \ell - 2$, we have that n is approximately $2^{3\ell/2}$. So we have found collisions for the function H, and hence for the MAC, considerably faster than a straightforward use of the birthday paradox would imply.

We have assumed in this analysis that the image of X under H is uniformly distributed in B. A non-uniform distribution only enhances the probability of collisions.

4 The second attack

Our first attack found collisions efficiently. However, it is not clear how knowledge of these collisions could be used to forge MACs. We now present a second method for finding collisions, almost as efficient as the method above, with the extra feature that collisions may be used to find the key words a_i, and hence to forge MACs.

We begin by choosing two integer parameters d and s, which affect the efficiency and probability of success of the attack. The value of d is the number of collisions we need to find in the compression function, and is chosen as follows. Let V be a $\mathbb{Z}_2$-vector space of dimension $t-1$. Then d is chosen so that the probability of a set of d randomly chosen vectors from V forming a spanning set is large (say at least 0.5). A choice of $d=t$ or $d=t+1$ will suffice in most situations. The parameter s relates to the size of a set Y' which is analogous to the set Y in our first attack. We choose s to be a positive integer such that 2^{ts} is just greater than $w=\lceil\sqrt{2dt(2^{3\ell+s})}\rceil$. For most parameter sets, this means that s is small ($s=2$ or $s=3$, say). Our choice of the values of d and s will become clear in the description of the attack below.

The attack proceeds in 3 stages. In stage 1, we find many collisions in the compression function H by requesting MACs of messages of a special form. In stage 2, we use data gathered from these collisions to form a system of linear equations in the unknown key words a_i. Solving these equations and performing a moderate amount of additional computation allows us to find the key $\underline{a}$ for the compression function H. In stage 3, we use knowledge of $\underline{a}$ to quickly find a collision in H without querying the MAC. This immediately leads to a MAC forgery.

Stage 1: We ask for the MACs of a subset Y' taken from the set of messages $X'=\{x_1x_2\cdots x_t \mid 0\leq x_i<2^s,\ \forall i\in\{1,2,\ldots,t\}\}$. There are $2^{ts}\geq w$ messages in X' and we take $|Y'|=w$. Define B' to be the set of all pairs $(z,\sigma)\in((\mathbb{Z}_{2^\ell})^2)^2$ such that the second component of $\sigma-\sigma_0$ lies between 0 and $t2^s$. Our condition on the elements x_i for messages in X' implies that $x_ia_i<2^{s+\ell}$ for all i. So the second component of v_i is less than 2^s and hence the second component of $\sigma-\sigma_0$ is less than $t2^s$. Thus the image of X' under H is contained in B', whatever the value of the secret key $a_1, a_2, \ldots, a_t$.

Notice that B' is of size $u = t2^{3\ell+s}$ and $Y' \subset X'$ is of size $w = \lceil \sqrt{2dt(2^{3\ell+s})} \rceil$. Our choice of parameters implies that, by the birthday paradox, we will find collisions in the MAC function, and hence in the compression function, for the set Y'. In fact, we have chosen d, s and w so that $|Y'|$ is a little larger than is needed for the birthday paradox to apply. This is so that we are likely to find many collisions. Indeed, applying results of [2], we have that for large w and u, the frequency distribution $Q(x)$ of the number of collisions x can be approximated by a Poisson distribution $P_\lambda(x)$ with parameter $\lambda = w^2/2u$. The error in approximating $Q(x)$ by $P_\lambda(x)$ can be bounded using results of [2]. We have:

$$|Q(x) - P_\lambda(x)| \leq \frac{5}{w}x^2 + \frac{3w}{u}x + \frac{w^2}{3u^2}. \tag{1}$$

For our choice of parameters and for $x \leq d$, the error term in eqn. (1) can be bounded by $8d^{3/2}/u^{1/2}$. In our situation, d will be much smaller than u. Thus the approximation by a Poisson distribution will be excellent when $x \leq d$. For our choice of parameters, we have $\lambda = d$. So in the case of interest to us (where d is approximately equal to t and so is reasonably large), the Poisson distribution has mean d, and median approximately d. Putting all of this together, we see that the probability that we find d or more collisions in Y' is approximately equal to 0.5. The above analysis can of course be refined, but suffices for our purposes.

Stage 2: We now assume that at least d collisions of the above type have been found. We proceed by examining the first component of σ for each of these collisions. Each gives an equation of the form

$$x_1^{(j)}a_1 + x_2^{(j)}a_2 + \cdots + x_t^{(j)}a_t = x_1'^{(j)}a_1 + x_2'^{(j)}a_2 + \cdots + x_t'^{(j)}a_t \mod 2^\ell,$$

for $j \in \{1, 2, \ldots, d\}$. Writing $y_i^{(j)} = x_i^{(j)} - x_i'^{(j)}$, we obtain a system of d equations in t unknowns $a_1, \ldots, a_t \in \mathbb{Z}_{2^\ell}$:

$$y_1^{(j)}a_1 + y_2^{(j)}a_2 + \cdots + y_t^{(j)}a_t = 0 \bmod 2^\ell, \quad 1 \leq j \leq d. \tag{2}$$

Define vectors $y^{(j)} \in (\mathbb{Z}_{2^\ell})^t$ by $y^{(j)} = (y_1^{(j)}, y_2^{(j)}, \ldots, y_t^{(j)})$. The number of solutions to the system (2) depends on the linear independence properties of the vectors $y^{(j)}$ considered modulo 2. Let $z^{(j)}$ denote $y^{(j)} \bmod 2$ and let V denote the dimension $t-1$ subspace of $\mathbb{Z}_2^t$ that consists of all vectors of even parity. Because the a_i are odd and the equations (2) hold, we have that $z^{(j)} \in V$ for $1 \leq j \leq d$. It is then elementary to show that the system (2) has a unique solution up to a $\mathbb{Z}_{2^\ell}$ scalar multiple if and only if the d vectors $z^{(j)}$ span the space V.

The probability that the d vectors $z^{(j)}$ span V is at least $1-1/2^{d-t+1}$, assuming the vectors to be random. (To see this, notice that the vectors fail to span V if and only if they all lie in some subspace U of co-dimension 1 in V. There are exactly $2^{t-1}-1$ such subspaces U. Assuming the vectors $z^{(j)}$ to be randomly distributed in V, the probability that they all lie in any given U is equal to 2^{-d}. Hence the probability that the vectors do not span V is at most $(2^{t-1}-1)\cdot 2^{-d} \leq 1/2^{d-t+1}$.) This probability is close to 1 as soon as d is slightly greater than $t-1$. Given that the vectors $z^{(j)}$ do span V, a standard Gaussian elimination procedure over $\mathbb{Z}_{2^\ell}$ can be used to produce $b_1, b_2, \ldots, b_t \in \mathbb{Z}_{2^\ell}$ such that there exists an odd constant c with the property that $a_i = cb_i \bmod 2^\ell$ for all $i \in \{1, 2, \ldots, t\}$.

Stage 1 of our attack has given us d pairs of messages that collide under the compression function. To find c, we simply try each of the $2^{\ell-1}$ possibilities in turn and check whether the compression function with key $a_i = cb_i$ produces collisions for these pairs of messages. It is highly likely that a single value of c will produce all the correct collisions; this value will be the correct value of c. Thus we have recovered the value of the key words a_i.

Stage 3: Finally, we produce a MAC forgery as follows. We search for collisions in the compression function as in Sect. 3; however, since we now know the key to the compression function, we do not need to query the MAC oracle to obtain these collisions. After about $2^{3\ell/2}$ trials, we find a collision in the compression function: $H(x) = H(x')$ for distinct messages x and x'. We then query the MAC oracle on the message x. The resulting MAC will be valid for the message x', and so we have forged a MAC as required.

To summarise, we have forged a MAC after making

$$w + 1 = \lceil \sqrt{2dt(2^{3\ell+s})} \rceil + 1$$

oracle queries, and a comparatively small amount of additional effort (which mainly consists of storing the oracle outputs, together with computing the compression function about $2^{3\ell/2}$ times). The probability that our attack works is approximately $(1 - 1/2^{d-t+1})/2$.

The probability of success can be made arbitrarily close to 1, firstly by increasing the value of d and secondly by taking a larger number of MAC queries to increase the probability that d pairs of collisions will result. We omit the routine details of these enhancements.

5 Consequences for suggested parameter sizes

In [1, Sect. 5], Cary and Venkatesan give details of an implementation of their MAC for the parameters $\ell = 32$ and $t = 50$. They are able to prove that the resulting MAC offers 54 bits of security, which can be interpreted as meaning that collisions for the MAC should not be found until after at least 2^{27} MAC queries have been made.

Our attack in Sect. 3 shows that, for these parameters, MAC collisions can be found using about $2^{48.5}$ MAC queries. (The attack will set $r = 31$; then the space B will be of size 2^{97} and $2^{48.5}$ MAC queries will be needed to obtain a collision with probability 0.63.). Taking $d = t = 50$, our attack in Sect. 4 uses $s = 2$ and finds MAC forgeries with probability about 0.25 using approximately 2^{55} MAC queries.

We have conducted experiments on cut down versions of the MAC. Using $t = 50$ and $8 \leq \ell \leq 14$, collisions in the compression function occurred as frequently as our analysis predicts. While we have not implemented our attacks for $\ell = 32$, we see no reason why our attack should not scale as predicted.

For both of our attacks, the complexity is significantly less than the 2^{64} queries implied by a standard application of the birthday paradox, though a good deal greater than the level of security that has been established for the MAC. It seems fair to say that the MAC proposal of [1] does not offer the security levels that one would expect of a strong MAC algorithm with a 128 bit output. We expect that more sophisticated attacks than the ones presented here may reduce the complexity of finding and exploiting collisions further.

In response to the attacks presented in this paper, the authors of [1] have suggested a new scheme, namely that the output of their compression function should first be passed through SHA-1 and then 54 bits of the result be taken as output. The intention of this construction is to match the proved security level of the original compression function with the length of the MAC. (The security level of this construction can be no greater than the 54 bits proved for the original MAC.) Note that the attacks presented in this paper do not carry over to this new scheme, since the overwhelming majority of collisions in the MAC will be caused by collisions in the final stage rather than by collisions in the compression function. Of course, a proof of security for this modified MAC can no longer rely on modelling a block cipher as a random permutation. Instead, this assumption would need to be replaced by an assumption concerning the collision properties of SHA-1 (or perhaps by modelling SHA-1 as a random function).

Acknowledgements

The authors would like to thank Sean Murphy for his help with background to the birthday paradox arguments in Sect. 4, Bart Preneel for bringing reference [2] to our attention, and Chris Mitchell for his comments on an earlier draft of this paper. The authors would also like to thank the referees of the paper, for their constructive comments and suggestions.

References

1. M. Cary and R. Venkatesan, "A message authentication code based on unimodular matrix groups", in D. Boneh, editor, *Advances in Cryptology – Proc. CRYPTO 2003*, Lecture Notes in Computer Science Volume 2729, (Springer, Berlin, 2003), 500-512.
2. M. Girault, R. Cohen and M. Campana, "A generalized birthday attack", in C. G. Gnther, editor, *Advances in Cryptology – Proc. EUROCRYPT'88*, Lecture Notes in Computer Science Volume 330, (Springer, Berlin, 1988), 129-156.
3. M.H. Jakubowski and R. Venkatesan, "The chain and sum primitive and its applications to MACs and stream ciphers", in K. Nyberg, editor, *Advances in Cryptology – Proc. EUROCRYPT '98*, Lecture Notes in Computer Science Volume 1403, (Springer, Berlin, 1998), 281-293.
4. A. Menezes, P.C. van Oorschot and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, 1997.