Fast Message Authentication Using Efficient Polynomial Evaluation

Valentine Afanassiev[1], Christian Gehrmann[2], Ben Smeets[2]

[1] Institute for Problems of Information Transmission
of the Russian Academy of Science, Bolshoj Karetnyj 19,
GSP-4, Moscow, Russia,
Email: afanv@ippi.ac.msk.su

[2] Department of Information Thechnology, Lund University,
Box 118, S-221 00, Lund, Sweden,
Email: {chris,ben}@it.lth.se

Abstract. Message authentication codes (MACs) using polynomial evaluation have the advantage of requiring a very short key even for very large messages. We describe a low complexity software polynomial evaluation procedure, that for large message sizes gives a MAC that has about the same low software complexity as for bucket hashing but requires only small keys and has better security characteristics.

Key words: Message authentication, universal hash functions, polynomial evaluation, software MAC generation.

1 Introduction

The verification of the authenticity of a text document or a datafile is one of the main applications of cryptographic techniques. A common used technique for this purpose is the application of a *message authentication code* (MAC). Basically we have two users called the sender S (or signer) and the verifier V. S and V share a secret random key string and a publicly known MAC. The MAC maps a message string to a shorter, so called, tag string. The sender calculates the tag corresponding to the message string and the shared secret key string and sends the message to V together with the tag. V accepts a received message if the received tag is the same as the tag for the received message and the secret key. A good MAC is designed to make it hard for an adversary to send own messages or substitute observed messages by new ones, without being detected by the receiver.

Usually one distinguishes between so called *unconditionally secure*, *computationally secure*, and *provable secure* authentication codes, [1, page 392]. Codes belonging to the first category are codes for which the security of the MAC is independent of the computational power of the adversary. The security of these codes is expressed in the probability of success of an deception attack. A MAC is called computationally secure if the adversary is faced with the difficulty that all *known* computational methods to perform an attack require a infeasible amount of computation. MACs for which it can be shown that an successful attack implies that some other, usually well-known and presumed hard, problem can be solved are called provably secure. Traditionally computational and provably secure codes have been considered to be more of practical interest. However, unconditionally secure codes can easily be turned into practical provable secure codes by using a finite pseudo random function as was shown in [2], [3]. Usually the MAC computation has to be done in software. This is no problem when the message is of small size. Designing good efficient MACs for large message sizes is a challenging problem.

Carter and Wegman [4] introduced the concept of universal families of hash functions. They can be used to construct unconditionally secure MACs as shown in, for example, [4] and [5]. They also have numerous applications outside cryptography. In this paper we study how to construct a good family of universal hash functions that have a low complexity and which are suitable for software implementation. This problem was also the subject of the paper by Rogaway [2] where he introduced bucket hashing. Bucket hashing is a clever way to construct a family of universal hash function. Using bucket hashing the authentication of a long message requires only about 9-13 machine instructions per message word. Thus bucket hashing leads to very fast MACs. However it requires a key string of about the same size as the message string. Even if this string will be computed from a main key by some strong random number generator it is desirable to keep the string short. In [6] constructions were given of universal families of hash functions based on a relation between authentication codes and error correcting codes. These constructions require very small key size even for long messages and small probabilities of deception (there exist multi-round authentication constructions with smaller key size [7], [8], [9]). The constructions in [6] require polynomial evaluation in a finite field.

OUR CONTRIBUTION: This paper describes an efficient procedure for evaluating polynomials over a finite field to be used in the construction of a fast MAC. We obtain a MAC that can authenticate messages in about 7-13 instructions per word. Furthermore, the random (key) string that is required is much smaller than for the bucket hashing construction. Moreover, the probability of deception in our construction is uniformly (for all keys) bounded, where as this probability is given as an average for bucket hashing.

We begin with describing the construction of a universal family of hash functions that we are going to investigate. In Section 3 we propose fast procedures for polynomial evaluation in large fields. We calculate the complexity of the proposed procedures. Finally we give examples of MAC computations and investigate how fast they can be done in software. We compare the evaluation procedures with bucket hashing.

2 Universal hash functions based on polynomial evaluation

2.1 MAC construction

A *family of hash functions* is a finite multi-set H of functions, each $h \in H$ having the same nonempty domain set A and range set B. In what follows we will assume A and B to be sets over binary alphabet $\{0,1\}$. We recall the following definitions.

Definition 1 [5]. Let $\epsilon > 0$. A multi-set H of n functions from a set A to a q -set B is ϵ-almost universal$_2$ (ϵ - AU$_2$) if for every pair $a_1, a_2 \in A, a_1 \neq a_2$ the number $d_H(a_1, a_2) = |\{h \in H; h(a_1) = h(a_2)\}| \leq \epsilon \cdot n$.

Definition 2 [5]. Let $\epsilon > 0$. A multi-set H of n functions from a set A to a q -set B is ϵ-almost strongly universal$_2$ (ϵ - ASU$_2$) if:

1. for every $a \in A$ and $y \in B$, the number of elements of H mapping $a \mapsto y$ is n/q,
2. for every pair $a_1, a_2 \in A, a_1 \neq a_2$, and every pair $y_1, y_2 \in B$ the number of elements of H that map $a_1 \mapsto y_1$ and $a_2 \mapsto y_2$ is $\leq \epsilon \cdot n/q$.

Given a family of hash functions H we can directly construct an unconditionally secure MAC [4], [5] or a complexity-theoretic variant when used together with a pseudo random function [2].

We investigate a construction of ϵ - ASU$_2$ family based on a concatenation of an ϵ_1 - AU$_2$ family and an ϵ_2 - ASU$_2$ family. For such a concatenation the following can be shown.

Theorem 3 [5]. *Let H_1 be and ϵ_1 - AU_2 from A_1 to B_1 and let h_2 be an ϵ_2 - ASU_2 from B_1 to B_2. Then $H = H_1 \times H_2$ is an ϵ - ASU_2 from A_1 to B_2 with $\epsilon \leq \epsilon_1 + \epsilon_2 - \epsilon_1\epsilon_2$.*

We also need the following lemma.

Lemma 4 [6]. *Let π be some $\mathbb{F}_{q_0}$-linear map from $\mathbb{F}_Q$ onto $\mathbb{F}_q$, where $Q = q_0^m, q = q_0^r$ and q_0 a prime power. Then the following family of hash functions $H = \{h_{a,b}; h_{a,b}(x) = \pi(ax) + b\}$, where $a, x \in \mathbb{F}_Q, b \in \mathbb{F}_q$ is ϵ - ASU_2, with $\epsilon = 1/q$.*

For large message sizes the following construction realizes an ϵ - ASU$_2$ family which maps elements from a large set A to a relative small set B for given ϵ requiring a small value of $|H|$.

Construction [6]: Let $q = 2^r, Q = 2^m = 2^{r+s}, n = 1 + 2^s$ and π be the same as in Lemma 4. Let $f_{\mathbf{a}}(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1}$, where $x, y, a_0, a_1, \ldots, a_{n-1} \in \mathbb{F}_Q$, $z \in \mathbb{F}_q$ and

$$H = \{h_{x,y,z} : h_{x,y,z}(\mathbf{a}) = h_{x,y,z}(a_0, \ldots, a_{n-1}) = \pi(yf_{\mathbf{a}}(x)) + z\}.$$

Theorem 5 [6]. *H in the construction above is ϵ - ASU_2 with $\epsilon \leq 2/2^r, |A| = Q^n = 2^{(r+s)(1+2^s)}$ and $|H| = Q^2q$.*

The construction is a concatenation of a $1/2^r$ - AU$_2$ family of hash functions obtained from a Reed Solomon code and a $1/2^r$ - ASU$_2$ family of Lemma 4.

The construction above also gives us directly an unconditionally secure MAC. Considering the vector $\mathbf{a} = a_0, \dots, a_{n-1}$ as a message where $a_0, a_1, \dots, a_{n-1} \in \mathbb{F}_Q$ and $x, y \in \mathbb{F}_Q$, $z \in \mathbb{F}_q$ as the key parts gives us the corresponding MAC. This means that for the unconditionally secure MAC we have a key size of $\log_2 |H| = \log_2(Q^2 q) = 2m + r$ and a tag size of $\log_2 |B| = \log_2 q = r$ bits. The probability of deception is less than $\epsilon \leq 2/2^r$.

The following construction is also useful if for given message size parameters the realized bound on ϵ is too large.

Theorem 6. *Let H be a ϵ - ASU_2 with n functions, domain set A, and range set B. Then the family H^2 with domain set A and range set $B \times B$ defined by*

$$H^2 = \{h = (h_1, h_2) \in H \times H;\ h : A \ni a \mapsto (h_1(a), h_2(a)) \in B \times B\} \qquad (1)$$

is an ϵ^2 - ASU_2. Furthermore, $|H^2| = n^2$.

Proof. The domain and range sets of H^2 as well as $|H^2|$ follow from the definition of H^2. Furthermore, it also follows from the definition that condition 1) of Definition 2 is satisfied for H^2 . Since the component functions h_1 and h_2 of $h \in H^2$ can be chosen independently there will be at most $(\epsilon \cdot n/|B|)^2$. But $n^2 = |H^2|$ and the range set of H^2 has cardinality $|B|^2$. Thus H^2 is an ϵ^2 - ASU$_2$.

2.2 A multiple message MAC

Assume we want to authenticate l messages. Let $\mathbf{a}_1, \dots, \mathbf{a}_l$ be a message sequence of l messages. For $1 \leq i \leq l$ let

$$H_i = \{h_{x,y,z_i} : h_{x,y,z}(\mathbf{a}_i) = h_{x,y,z_i}(a_{i_0}, \dots, a_{i_{n-1}}) = \pi(y f_{\mathbf{a}_i}(x)) + z_i\},$$

where $a_{i_0}, \dots, a_{i_{n-1}} \in \mathbb{F}_Q$ and $x, y \in \mathbb{F}_Q, z_i \in \mathbb{F}_q$, be the hash function of the i-th message. It was shown in [4], [10] that this gives us an unconditionally secure multiple message MAC. The key parts x and y may in this construction be considered as "hidden" key parts and they can remain unchanged for all messages in the sequence. Only the part z_i has to be refreshed for each message. A complexity-theoretical secure MAC for multiple use can easily be obtained from the unconditionally secure by changing the key part z_i by the output of a pseudo random function, see for example [2].

Recently, in [11] a new method for multiple authentication was proposed. This method does not use a refreshed key part and, hence, it does not require the z_i (called counter in [11]) for multiple authentication. Consequently there is no problem if one or several messages are lost during transmission. However this method exhibits a growth of the key size which is quadratic in the number of messages to be authenticated! Our method can easily be extended to handle lost messages or synchronization loss by adding the index i to the message a_i to be authenticated.

We consider multiple message MAC as a way of balancing between the complexity of pre-calculations and the complexity of a MAC calculation. In the next section we propose an efficient method for the evaluation of a polynomial of a large degree over a large finite field.

3 Polynomial evaluation

3.1 The basic procedures and their complexity

The main step in the MAC construction is the evaluation of a polynomial

$$f_{\mathbf{a}}(x) = \sum_{i=0}^{n-1} a_i x^i \text{ where } a_i \in \{0,1\}^w$$

in some point of a finite field $\mathbb{F}_Q$. Let $Q = 2^m$, $w \leq m$ and $n < Q$. As the measure of computational complexity we will use the *binary complexity* and the *multiplicative complexity.*

Definition 7. The **binary complexity** $C_b(\Phi)$ of the calculation [12] of some function Φ is defined as the total (minimal) number of elementary bit (Boolean or bit-level) operations from the set $\{\vee, \wedge, \oplus\}$ used in the calculation of Φ.
The **multiplicative complexity** $C_m\,(\Phi)$ is defined as the total (minimal) number of multiplications over the finite field used in the calculation Φ.

We also can use the binary complexity reduced to a subset of $\{\vee, \wedge, \oplus\}$, for example, the number of modulo 2 additions $C_\oplus(\Phi)$.

Lemma 8. *For any pair of elements of a finite field* $\mathbb{F}_Q$, $Q = 2^m$, *the binary complexity of addition is* $C_b(+) = C_\oplus(+) = m$ *and an upper estimate of the binary complexity of multiplication is* $C_b(*) = O\left(m^2\right)$.

Remark: This lemma has been proven in several variants. The bound $C_b(+) = C_\oplus(+) = m$ is trivial. The bound $C_b(*) = O\left(m^2\right)$ has been obtained for bit-serial structures of multipliers over finite fields as the product of m clock-cycles and m functional gates ($\oplus$, for example) for a dual basis [13], for a standard basis [14] and for an optimal normal basis [15]. It is known as an estimate of the number of functional elements for bit-parallel structures over the standard basis [16], [17]. It is also known for systolic arrays and other structures. The lowest asymptotic upper bound $O\left(m \log^{1+\epsilon} m\right)$, $\epsilon > 0$ can be obtained through fast convolution that uses a FFT over a finite or a surrogate field. However, it becomes less complex only for very large m (more than 1000). A modification of Karatsuba's method (or recursive double length multiplication formula [21]) for fast polynomial multiplication [16] gives another upper bound, $O\left(m^{\log 3}\right)$, that is applicable for the range approximately from 50 to 1000. So, for the most practical range of finite fields ($<$ 100) we have to use a quadratic bound. It can be shown that $C_b(*) \leq 4m^2$ and $C_\oplus(*) \leq 2m^2$ for a multiplier with a parallel structure [16].

A well known procedure for the calculation of $f_{\mathbf{a}}(\alpha)$ for any element α of $\mathbb{F}_Q$ is

Horner's procedure:

For given $\alpha \in \mathbb{F}_{2^m}$ and message $f(x)$:
$u_1 = a_{n-1}; u_{i+1} = u_i * \alpha + a_{n-i};\ \ i = 1, \ldots, n;\ \ u_n = f_{\mathbf{a}}(\alpha)$.

This procedure takes at most n additions and exactly n multiplications in $\mathbb{F}_Q$. So, the upper estimate for binary complexity of Horner's procedure over $\mathbb{F}_Q$ is $C_b(Horner) \leq nC_b(*) + nm = nO(m^2)$.

We will consider the **MinPol procedure** that has a minimal multiplicative complexity. But, first we have to recall the definition of minimal polynomials [15] in a finite field. The minimal polynomial $\mu_\alpha(x)$ of any element $\alpha \in \mathbb{F}_Q/\{0\}$, $Q = 2^m$, is

$$\mu_\alpha(x) = \Pi_{j=0}^{t-1}\left(x + \alpha^{2^j}\right)$$

where the integer $t = m$ or $t \mid m$ is the minimal solution of $\alpha = \alpha^{2^t}$. If $t \mid m$, then α is an element of a subfield of $\mathbb{F}_{2^m}$. A minimal polynomial is clearly irreducible over the ground field $\mathbb{F}_2$ [15].

The following procedure for the computation of $f_{\mathbf{a}}(\alpha)$ has the minimal multiplicative complexity over $\mathbb{F}_{2^m}$:

MinPol procedure:

For given $\alpha \in \mathbb{F}_{2^m}$ and message $f(x)$:
1. Calculate the minimal polynomial $\mu_\alpha(x)$ of α.
2. Calculate $r_\mu(x) = f_{\mathbf{a}}(x) \bmod \mu_\alpha(x)$
3. Calculate $f_{\mathbf{a}}(\alpha) = r_\mu(\alpha)$

The first step of the MinPol procedure takes $t-1$ squarings in $\mathbb{F}_{2^m}$ for the calculation of the conjugated elements of α and $\leq t(t-1)/2$ multiplications and additions in $\mathbb{F}_{2^m}$ for the polynomial product calculation. The second step (step 2) takes $\leq (n-t)t$ additions in $\mathbb{F}_{2^m}$ and the last step (Horner procedure) takes t additions and t multiplications in $\mathbb{F}_{2^m}$. Total multiplicative complexity in $\mathbb{F}_{2^m}$ (steps 1 through 3) is $\leq 2t + t^2/2$. Thus the multiplicative complexity is independent of the degree of the polynomial $f_{\mathbf{a}}(x)$.

The following upper estimate is valid for the binary complexity over $\mathbb{F}_{2^m}$ of the MinPol procedure

$$C_b(MinPol) < nm^2 - \frac{m^3}{2} + m^2 O\left(m^2\right).$$

We see that it has the same main term, i.e., nm^2, but in the MinPol procedure this is a strong upper estimate while for the Horner procedure this term depends on the multiplication complexity for the field $\mathbb{F}_{2^m}$. Now we can formulate a new problem: *find an evaluation procedure with the main term of order nm instead of* nm^2. We will see that for some nontrivial subsets of elements in $\mathbb{F}_{2^m}$ this estimate is achievable.

3.2 A new low complexity procedure for polynomial evaluation

Definition 9. The Minimal W–nomial $\tau_{\alpha,w}(x)$, $\alpha \in \mathbb{F}_{2^m}$, is the solution with respect to n_i and t of the equation

$$\min_t \left(x^t + \sum_{i=1}^{w-2} x^{n_i} + 1 = 0 \mod \mu_\alpha(x) \right), t > n_i > 0, w \le W.$$

By definition $\tau_{\alpha,w}(\alpha) = 0$ and $\tau_{\alpha,w}(x) = \mu_\alpha(x)$ if and only if the weight of $\mu_\alpha(x)$ is $\le W$. If there is a multiple solution for different $w \le W$ then we have an optimization problem for the complexity estimate. For the moment our choice is for minimal t.

Now we give a construction of a new low complexity procedure based on Minimal W–nomials.

MinWal procedure:

```
For given $\alpha \in \mathbb{F}_{2^m}$ and message $f(x)$ :
1. Calculate the MinPol $\mu_a(x)$
2. Search for MinWal $\tau_{\alpha,w}(x) = 0 \mod \mu_\alpha(x)$
3. Calculate $r_\tau(x) = f(x) \mod \tau_{\alpha,w}(x)$
4. Calculate $r_\mu(x) = r_\tau(x) \mod \mu_\alpha(x)$
5. Calculate $f_{\mathbf{a}}(\alpha) = r_\mu(\alpha)$
```

To estimate the expected complexity we assume that a MinWal of degree $t < n$ exists (Step 2 is successful). Then Step 3 takes $\le (w-1)(n-t)$ additions and Step 4 takes $< (t-m)m$ additions in $\mathbf{F}_Q$. So if $t \lesssim n/(m-w+1)$ then we can expect that the main term of the binary complexity to be of order $\lesssim nwm$. The complexity of Step 2 and the search procedure depend both on w.

The existence of MinWals is a consequence of the existence of Hamming and other cyclic codes of minimal distance $\le w$ when $\mu_\alpha(x)$ is the factor of a generating polynomial of the code. *We call an element α bad if it has no corresponding W–nomial.* If α is an element of a subfield $\mathbb{F}_{Q'}$ of $\mathbb{F}_Q$ then there exists a binomial of degree $Q' \le \sqrt{Q}$ which can be used instead of 3-nomial. As we will show later, the maximal degree of a W–nomial is of order $Q^{1/w-1}$. If α is an element from the multiplicative subgroup of order L, then α is a root of a binomial of degree L. This binomial can be used instead of the W–nomial if $L < Q^{1/w-1}$. So the only open question is *the existence of irreducible polynomials of weight $> w$ and degree m (and $m-1$ for 4 or 5-nomials) such that they generate a cyclic code of minimal distance $> w$ and length $L > Q^{1/w-1}$.*

The class of irreducible shortened cyclic codes, i.e., generated by an irreducible polynomial, has been considered in [18], and other papers. It has been proven that the codes in this subclass satisfy the Gilbert bound. This result was generalized to the class of shortened cyclic codes in [19]. Taking into account these results we can expect that the irreducible shortened cyclic code related with a bad element α should be very short in comparison with the threshold $Q^{1/w-1}$. We are not able to prove a more exact statement about bad elements in $\mathbb{F}_Q$. However because the codes related to the bad elements are short with high probability, we expect, provided $L > Q^{1/w-1}$, that the probability to choose a bad element is extremely low for large Q.

As a demonstration of the fact that the search of W−nomials is not a too complicated problem we give the description of

MinTrinSearch procedure:

```
For a given minimal polynomial $\mu_a(x)$ and $r_i(x) = x^i, i = 0 \dots m-1$,
Calculate for $i = m, m+1, \dots.$
1. $r_i(x) = x * r_{i-1}(x) \mod \mu_a(x)$ and
2. Find first $j < i$ with $r_j(x) + 1 = r_i(x)$.
```

It is clear that $i \le T$, where T is the degree (minimal) of a MinTrin we expect to find in the search. If the storage of $r_i(x)$ is organized as a dichotomous (binary) tree then the complexity of the **MinTrinSearch** procedure is linear in the degree $t_3 \le T$ of the Minimal Trinomial (if it exists). In fact its binary complexity is $O(t_3 m)$. A similar search procedure could be used with other polynomials of a limited weight. The main trick is to organize the residues in a tree structure. However, the search procedure for 4 or 5− nomials has complexity $O(t_4^2 m)$ or $O(t_5^2 m)$.

3.3 New estimates for the complexity of polynomial evaluation

In this section we will give arguments to obtain a simple lower and upper estimates for the maximal degree of MinWal.

Definition 10. The maximal degree T_w of the Minimal W−nomial (MinWal) $\tau_{\alpha,w}(x)$ over $\mathbb{F}_Q$ (cf. Definition 9), is

$$T_w = \max_{\mu_\alpha(x)} \left(\deg \tau_{\alpha,w}(x) : \tau_{\alpha,w}(x) = 0 \mod \mu_\alpha(x)\right),$$

where $\alpha \in \mathbb{F}_Q$ and $\mu_\alpha(x)$ is the MinPol of α.

Proposition 11. *Lower and upper estimates for the maximal degree T_w of MinWal $\tau_{\alpha,w}(x)$ over $\mathbb{F}_Q$, $Q = 2^m$, are*

$$L_{w,m} = ((w-1)!\,2Q)^{1/(w-1)} <\approx T_w <\approx L_{w,m}\eta^{\frac{1}{w-1}} = U_{w,m},$$

where $\eta = 2m/\log m$.

Proof. Let $T_w = T$. A W−nomial of degree t is *normal* if it is written as $1 + \sum_{i=1}^{w-2} x^{n_i} + x^t$ with $1 \le \dots < n_{i-1} < n_i < \dots < t$. The number $D_{w,T}$ of normal W−nomials of degree $t \le T$ is

$$D_{w,T} = \sum_{t=w-1}^{T} \binom{t-1}{w-2} = \binom{T}{w-1} < \frac{T^{w-1}}{(w-1)!}$$

We are going to estimate $L_{w,m}$ as the necessary T such that this set of normal $W-$nomials includes all MinWals over $\mathbb{F}_Q$ and only partially their multiples by squaring and other MinWals over $\mathbb{F}_{2^m}$ for $m < n \leq T$. The estimate $U_{w,m}$ will be given as the sufficient T for the same condition.

Let M_m be the number of MinWals for the field $\mathbb{F}_Q$. Clearly each MinWal can be squared less than m times if $T < 2^m$. Assuming that one MinWal is related to one of the irreducible polynomials of degree $\leq m$ we have an estimate $M_m \approx \sum_{k \leq m} I_k$ where I_k is the number of irreducible polynomials of degree k. Taking into account that $\sum_{k \leq m} I_k \ll \sum_{k \leq T} I_k$ for $T \geq 2m$ we can estimate $L_{w,m}$ as the solution of $D_{w,T} > mM_m$ with respect to T. A well known asymptotic estimate for I_k is $2^k/k$ [15] and for $\sum_{k \leq m} I_k$ is $2^{m+1}/m$. So we can write

$$T > L_{w,m} = ((w-1)!\, 2Q)^{1/(w-1)} .$$

Now we assume that each reducible MinWal of degree t over $\mathbb{F}_Q$ has one factor of degree $\leq m$ and the others of degree $> m$. We can expect that any binary polynomial of degree $t \leq T$ has $\leq \eta$ different irreducible factors of degree $\leq T$. This holds for $W-$nomials and MinWals as well. Now we can declare that $\frac{1}{m} D_{w,T}$ is less than the number of normal $W-$nomials of degree $t < T$ and is less than total number of their different irreducible factors. Taking into account that $M_m \approx \sum_{k \leq m} I_k \ll \sum_{k \leq T} I_k$ for $T > 2m$ we can estimate the total number of different irreducible factors as ηM_m . Thus we have $\frac{1}{m} D_{w,T} <\approx \frac{2Q}{m}\eta$ and

$$T < U_{w,m} = (2\,(w-1)!\, Q\eta)^{1/(w-1)} .$$

The exact upper bound for the number of different monic divisors of a binary polynomial of degree $T < 2^m$ is proved in [19] in the form $\eta \leq \frac{m}{2+\log m}\,(1+o\,(1))$. For our purposes we can simplify a little this bound to the form $\eta \leq \frac{2m}{\log m}$.

The last proposition is more a hypothetical than a strong result because we have to rely on some reasonable but unproven assumptions. Let us consider another upper estimates for T_w. The total number of $W-$nomials can be estimated as the number of codewords A_w of weight w of a Hamming code (excluding their cyclic shifts) times the number of different codes. So, the estimate $\binom{T_3}{w} \approx A_w \frac{2Q}{m} <\approx \frac{1}{T}\binom{T_3}{w}\frac{2Q}{m}$ leads to an upper bound $T_3 < \frac{2Q}{m}$. Because each cyclic code of the distance $\leq w$ gives only one MinWal (of weight w) we can estimate the number of MinWals through the number of cyclic codes of that distance. If we estimate the number of codes through all pairs of irreducible polynomials of degree m or less we have for 5-nomials $T_5 \leq \left(\frac{48Q^2}{m}\right)^{1/4}$.

The estimation of the complexity of a polynomial evaluation concludes this section. It is evident that the calculation of $f_{\mathbf{a}}(\alpha)$ modulo the W$-$nomial needs $\leq (w-1)\,(n-T_w)$ additions in $\mathbb{F}_Q$, where T_w is the degree of the W$-$nomial. Returning to the **MinWal** and **MinPol** procedures we can estimate the binary complexity of the **MinWal** procedure for evaluation of a polynomial of degree $n \leq Q-2$ in a point of the finite field $\mathbb{F}_Q$, $Q = 2^m$, as

$$C_b \leq m\,(n\,(w-1) + (m-w+1)\,U_{w,m}) + O\left(m^3\right) .$$

field size	real values		bounds (max)	
Q	mean	max	$\sqrt{4Q}$	$\sqrt{4Q\eta}$
2^9	29	61	45	107
2^{10}	40	83	64	157
2^{11}	64	143	90	227
2^{12}	85	217	128	331
2^{13}	128	337	181	480
2^{14}	181	473	256	694
2^{15}	257	801	362	1003
2^{16}	409	1285	512	1448

Table 1. Complete statistics for the degree of trinomials compared with lower and upper bounds for maximal degree of trinomials for some small fields.

This estimate is of order $nm\beta$ when $U_{w,m} \approx \alpha n$, $\alpha = \frac{\beta-w+1}{m-w+1} < 1$, and β is a constant $w-1 < \beta < m$.

A hash procedure that uses a new evaluation point for each message, would demand a message length of order of the square of the minimal 4−nomial or 5−nomial. In contrast, when using the evaluation for multiple MAC calculations the 4−nomial or 5−nomial could be considered as part of the "hidden" key. Thus they only have to be calculated once and can then be used to authenticate a large number of messages. We will discuss these aspects further in Section 4.

3.4 Experimental results

As we can see there is a gap between the lower and the upper estimate of the maximal degree of the MinWal. Furthermore, they are only estimates. We are therefore interested in investigating how tight our estimates are. We continue by discussing experimental results for the degree of minimal trinomial, quadronomial and pentanomial for different field sizes. We have calculated the complete statistics for the average and maximum degree of minimal trinomials for field size between 2^9 and 2^{16}. These values together with the corresponding values obtained from our lower, $L_{3,m} = \sqrt{4Q}$, and upper, $U_{3,m} = \sqrt{4Q\eta}$, $\eta = 2m/\log m$, estimates on the maximum degree of the minimal trinomial are listed in the Table 1. It can be seen from Table 1 that the mean degree of the trinomial is rather close to $L_{3,m}$.

It is not possible to calculate the complete statistics for fields of large sizes in reasonable time. To see how the degree of the minimal 3, 4 and 5− nomials are distributed for larger fields, we have chosen random elements from fields of size less or equal 2^{31}. We have investigated the distribution of minimal W−nomial degree for a logarithmic scale. We tested a *log-normal* approximation for the distribution of the minimal degrees and it turns out to give good agreement in χ^2 tests. The means of the degrees for the tests are close to our lower bounds

W	Field	# elm.	Exper.degree		Lower	Upper	log degree		Pr{t>U$_{3,m}$}
	size		mean	max	$L_{3,m}$	$U_{3,m}$	mean	STD	
3	2^{15}	compl.	257	801	362	1003	5.36	0.67	$1.0 \cdot 10^{-2}$
	2^{16}	compl.	409	1285	512	1448	5.81	0.68	$1.5 \cdot 10^{-2}$
	2^{19}	3200	1049	3517	1448	4331	6.79	0.64	$6.7 \cdot 10^{-3}$
	2^{25}	3200	8399	29505	8437	27684	8.88	0.64	$1.7 \cdot 10^{-2}$
	2^{31}	800	66937	210074	92682	327872	10.9	0.62	$2.2 \cdot 10^{-3}$
W	Field	# elm.	Exper.degree		$L_{4,m}$	$U_{4,m}$	log degree		Pr{t>U$_{4,m}$}
4	2^{15}	compl.	55	151	73	145	3.940	.39	$3.7 \cdot 10^{-3}$
	2^{16}	compl.	70	141	92	185	4.17	.37	$2.5 \cdot 10^{-3}$
	2^{19}	2400	138	298	185	384	4.85	.40	$3.0 \cdot 10^{-3}$
	2^{25}	2400	551	1164	738	1630	6.23	.42	$3.0 \cdot 10^{-3}$
	2^{31}	400	2179	4608	2954	6858	7.6	.42	$1.8 \cdot 10^{-3}$
W	Field	# elmn.	Exper.degree		$L_{5,m}$	$U_{5,m}$	log degree		Pr{t>U$_{5,m}$}
5	2^{15}	compl.	28	51	35	59	3.29	.266	$1.5 \cdot 10^{-3}$
	2^{16}	compl.	33	57	60	104	3.47	.277	$1.1 \cdot 10^{-5}$
	2^{19}	3200	56	106	168	304	3.98	.296	$2.4 \cdot 10^{-9}$
	2^{25}	2400	156	321	476	895	5.0	.316	$6.6 \cdot 10^{-9}$
	2^{31}	400	433	726	566	1066	6.0	.316	$1.1 \cdot 10^{-3}$

Table 2. Minimal degree of W−nomial ($W = 3, 4, 5$) for random elements or complete statistics ($\mathbb{F}_{2^{15}}$ and $\mathbb{F}_{2^{16}}$) in different fields, the corresponding bounds for W−nomial degree t and the probability for a random element to take a larger degree for the W−nomial than the U$_{w,m}$, given a normal distribution with the estimated mean and standard deviation for a logarithmic scale.

for the maximal degree of the polynomials. This can be seen for the examples in Table 2.

It can be seen in Table 2 that the variance varies a little with field size in the three different cases, but that it is about the same value independent of the field size. The experimental maximal value for the tested fields are not the true maximum value, but just the maximum value for the tested elements. As we can see the hypothesis on η has a good experimental support .

To explain the results presented in Table 2 we can use estimates for the average number η_{avr} of different irreducible factors of a polynomial of degree $t \leq 2^m$, $\eta_{avr} = \log m + c + O\left(\frac{\log m}{m}\right)$ [20] where c is some constant. Thus, by using η_{avr} instead of η we can estimate $T_{w,avr} \approx L_{w,m}\left(\eta_{avr}\right)^{1/w-1}$. Now we can estimate the span for the log of the W−nomial degree as

$$\log T_w - \log T_{w,avr} = \frac{\log m - 2\log\log m}{w-1}.$$

This estimate can be related linearly to the standard deviation of the experimental distribution of the degree of the W−nomial. As we can see from numeric calculations the last estimate varies very little within a range $m \leq 256$ for a fixed w. This estimate is more sensitive to w for $w \leq 5$.

4 Software MAC computations

It is often necessary to compute the MACs in software on a workstation or personal computer. In this section we investigate how fast the previous polynomial evaluation procedure would be if it is implemented in software. We make a comparison with the Bucket hashing method.

The evaluation point or "hidden" key in the polynomial evaluation MAC generation could be unchanged for several messages. For the trinomial procedure the complexity of the calculation of trinomial is linear in the degree and would not give a main contribution to the overall complexity of the MAC calculation. The 4 or 5−nomial calculations in our evaluation procedures only depend on the evaluation point and can be precomputed and used as part of the "hidden" key. We will in the comparison with Bucket hashing not include the key generation in the calculations, but only the message authentication for a message given a secret key. As we will show below Bucket hashing demands a much longer key than the polynomial evaluation method and would in general for long messages be more complex to generate than the 4-nomial or 5-nomial for the evaluation procedure.

Assume we use a computer with $k-$bit architecture (typical $k = 32$ or $k = 64$). We also assume that the message is represented as a vector of words of k bits. Reducing modulo a $W-$nomial and modulo a minimal polynomial is implemented for this $k-$bit architecture with no relation to the given finite field. However at the last stage of the **MinWal** procedure we have to implement the arithmetic of the given field.

The latter has also implications for the cache hit rate when executing the algorithm on a usual computer. After the first reduction by the $W-$nomial which coincides with reading the message (e.g. from disk), most of the computations occur 'locally' which gives high cache hit rates.

As we have shown previously the modulo calculations demand $w - 1$ additions for each message word using a $W-$nomial. For the typical structure of a CPU instruction based on a bank of internal register we have the following approximations for the number of instructions for the MAC calculations by **MinWal** procedure: $\# = 3w - 2$ instructions per word (7, 10 or 13 using the 3, 4 or 5-nomial, respectively). If the CPU XOR instruction includes internal register and an address in main memory and contains the read and write cycle then we have the lower approximation: $\# = 2w - 1$. This can be compared with the Bucket hashing method, which needs about $9 - 13$ instructions per word [2].

We have investigated the method for different parameters. According to the experimental results in the previous section the expected $W-$nomial degree are a little bit below the lower bounds for the maximal degree. Hence, we have used the lower bounds as an estimate for the mean of a log normal distribution for the different fields. Assume $\log t \in N(x, \sigma)$, i.e., a normal distribution with mean

x and variance σ. Then the mean x' of the log *normal* distribution is given by $x' = e^{x+\sigma^2/2}$. Hence, we would have the following estimate

$$x_w = \log L_{w,m} - \sigma_w^2/2$$

for the mean of the normal distribution. According to the experimental results the variance for the normal distributions for W−nomial, $W = 3$, 4, 5, are likely to be values around 0.4, 0.18 and 0.1 respectively. We have used these values as estimates for the normal distribution of the degrees using a logarithmic scale. We have calculated parameters for using polynomial evaluation for different message lengths n and polynomial degrees T for the different field sizes in Table 3 for a probability of deception 2^{-30}, 2^{-40}, and 2^{-60} respectively. We have assumed that the computer word size is 32 bits. The degrees T are chosen to have a small estimated probability $\Pr\{t > T\}$, according to the approximations above. If we use the evaluation procedure for fewer number of words, the total complexity would of course be less, but the number of instructions per word would increase.

To lessen the effect that if we require a small ϵ we need large messages to get an efficient evaluation procedure we use the result of Theorem 6 to obtain a new MAC from a given MAC with ϵ_0 that has twice the key size, identical message size, and $\epsilon = \epsilon_0^2$. Although at first glance this will also double the time to compute the tag but it will in fact be less, say 50% only, if properly implemented. For some of the examples in the tables we have used this doubling of tag and key size to get an efficient MAC with the desired probability of deception.

In the Table 3 we give examples where we have chosen the minimal field size for the given probability of deception. We are able to freely choose a proper field for the MAC because the first steps of the evaluation procedure are independent of field size. When we calculated the table we assumed the 4− nominal and 5−nominal to be part of the key and hence the key size is given by key $\text{size}_w = 2m + r + (w-1)\log_2 U_{w,m}$ where the last term could be omitted in the case of a 3-nomial. The tag size for the polynomial evaluation MAC calculation is for all examples much smaller than that for Bucket hashing. In [11] other examples with different parameters are given. There is also comparisons with other authentication codes. It can be seen in the table that for shorter messages the 5− nominal is better to use than 3−nominal or 4−nominal. It is not possible to have a low number of instructions per word for shorter messages and still have a low probability of deception ϵ. The short authentication tag and small key size for the evaluation procedure and its suitability for very fast implementation makes it much better to use than Bucket hashing for long messages.

5 Conclusion

We suggested an efficient polynomial evaluation procedure based on calculations modulo a 3, 4 or 5−nomial that leads to a fast MAC. The complexity of our procedure is closely related to the degree of these low weight polynomials. We derived estimates of lower and upper bounds on the degrees of these polynomials and compared these bounds with experimental results.

$\epsilon = 2^{-30}$	W-nomial method						Bucket hashing [2]	
n	field	W	tag size	key size	T	$\Pr\{t > T\}$	tag size	key size
2^{25}	41	3	2×16	2×98	2^{22}	0.1937	18784	$9.26 \cdot 10^8$
2^{30}	46	3	2×16	2×108	2^{25}	0.0790	595521	$3.50 \cdot 10^{10}$
2^{17}	33	4	2×16	2×133	2^{13}	0.0633	4256	$2.78 \cdot 10^6$
2^{23}	54	4	31	211	2^{20}	0.0633	11840	$2.15 \cdot 10^8$
2^{16}	47	5	31	197	2^{14}	0.0212	4128	$1.38 \cdot 10^6$
$\epsilon = 2^{-40}$	W-nomial method						Bucket hashing [2]	
n	field	W	tag size	key size	T	$\Pr\{t > T\}$	tag size	key size
2^{31}	52	3	2×21	2×125	2^{28}	0.0790	75040	$7.21 \cdot 10^{10}$
2^{17}	38	4	2×21	2×154	2^{15}	0.0191	12608	$3.40 \cdot 10^6$
2^{29}	70	4	41	271	2^{26}	0.0044	47296	$1.70 \cdot 10^{10}$
2^{13}	34	5	2×21	2×149	2^{11}	0.0050	12576	$2.24 \cdot 10^5$
2^{23}	64	5	41	261	2^{18}	0.06940	14400	$2.22 \cdot 10^8$
$\epsilon = 2^{-60}$	W-nomial method						Bucket hashing [2]	
n	field	W	tag size	key size	T	$\Pr\{t > T\}$	tag size	key size
2^{31}	52	3	3×21	3×125	2^{28}	0.0790	131200	$7.73 \cdot 10^{10}$
2^{17}	38	4	3×21	3×154	2^{15}	0.0191	126752	$4.82 \cdot 10^6$
2^{26}	57	4	2×31	2×223	2^{21}	0.0633	126880	$2.41 \cdot 10^9$
2^{13}	34	5	3×21	3×149	2^{11}	0.0050	126752	$4.20 \cdot 10^5$
2^{20}	51	5	2×31	2×209	2^{15}	0.0212	126752	$3.77 \cdot 10^7$

Table 3. Construction parameters using polynomial evaluation and Bucket hashing with a computer word size of 32 bits for different message lengths for a probability of deception less than 2^{-30}, 2^{-40}, and 2^{-60}. Using the MAC more than once is marked with × (for example 2 × 21) in the key size column.

We investigated how fast the evaluation can be made in software. In terms of speed our procedure can be compared with bucket hashing [2]. But, our method requires much shorter key than those based on bucket hashing. In Bucket hashing a key size of about the same size as the message is required. By using polynomial evaluation it is possible to reduce the key to a size of about as large as that of the tag. When 4 or 5-nomials are used for the evaluation, they have to be precomputed and should be a part of the key. The search for a 4 or 5-nomial can be of rather high complexity. However, the same 4 or 5-nomial can be used to calculate MAC's for several messages. Loosely speaking one can say that our polynomial evaluation procedure is efficient for large messages and for multiple authentication.

Our evaluation method can be used in other constructions of universal families of hash functions to get similar complexity reductions.

References

1. G.J. Simmons, "A survey of information authentication", in *Contemporary Cryptology, The Science of Information Integrity*, ed. G.J. Simmons, IEEE Press, New York, 1992.

2. P. Rogaway, "Bucket hashing and its application to fast message authentication", *Proceedings of CRYPTO '95*, Springer Verlag, pp. 29-42, August, 1995.
3. M. Bellare, J. Kilian, and P. Rogaway, "The security of cipher block chaining", *Proceedings of CRYPTO' 94*, Springer Verlag, pp. 341-358, August, 1994.
4. M. Wegman and L. Carter, "New hash functions and their use in authentication and set equality", *J. of Computer and System Sciences 22*, pp. 265-279, 1981.
5. D. Stinson, "Universal hashing and authentication codes", *Designs, Codes and Cryptography*, Vol. 4, pp. 369-380, 1994.
6. J. Bierbrauer, T. Johansson, G. Kabatanskii and B. Smeets, "On families of hash functions via geometric codes and concatenation", *Proceedings of CRYPTO '93*, Springer Verlag, pp. 331-342, 1994.
7. P. Gemmell and M. Naor, "Codes for interactive authentication", *Proceedings of CRYPTO '93*, Springer Verlag, pp. 355-367, 1994.
8. C. Gehrmann, "Cryptanalysis of the Gemmell and Naor multiround authentication protocol", *Proceedings of CRYPTO '94*, Springer Verlag, pp. 121-128, 1994.
9. C. Gehrmann, "Secure multiround authentication protocols", *Proceedings of Eurocrypt '95*, Springer Verlag, pp. 158-167, 1995.
10. T. Johansson, *Contribution to Unconditionally Secure Authentication*, Ph. D. thesis, Lund 1994.
11. M. Atici and D. R. Stinson, "Universal hashing and multiple authentication", *Proceedings of CRYPTO '96*, Springer Verlag, pp. 16-30, 1996.
12. J.E. Savage, "The complexity of decoders. Computational work and decoding time", *IEEE. Trans. Inform. Theory*, Vol. 17, pp. 77-85, January, 1971.
13. E.R. Berlekamp, "Bit-serial Reed-Solomon encoder", *IEEE. Trans. Inform. Theory*, Vol. 28, pp. 869-874, November, 1982.
14. M.A. Hasan and V.K. Bhargava, "Division and bit-serial multiplication over $GF(q^m)$", *IEE Proceedings-E*, Vol.139, No. 3, May, 1992.
15. D. Jungnickel, *Finite fields: structure and arithmetics*, Wissenschaftsverlag, Mannheim-Leipzig-Wien-Zurich, 1993.
16. V.B. Afanassiev, "On the complexity of finite field arithmetic", *Fifth Soviet-Swedish Int. Workshop on Inform. Theory, Moscow*, January, 1991.
17. E.D. Mastrovito, *VLSI Designs for Computations over Finite Fields $GF(2^m)$*, Internal Report LiTH-ISY-I, Linköping Univ., Sweden, 1988.
18. T. Kasami, "An upper bound on k/n for affine-invariant codes with fixed d/n", *IEEE Trans. Inform. Theory*, Vol. 15, pp. 174-176, January, 1969.
19. Ph. Piret, "On the number of divisors of a polynomial over $GF(2)$", Springer Verlag, Lecture Notes in Comp. Sci. 228, pp. 161-168, 1985.
20. I. E. Shparlinski, *Computational and algorithmic problems in finite fields*, Kluwer, Dordtrecht-Boston-London, 1992.
21. H. Riesel, *Prime Numbers and Computer Methods for Factorization*, Birkhuser, Boston-Basel-Stuttgart, 1985.