LFSR-based Hashing and Authentication

Hugo Krawczyk

IBM T.J. Watson Research Center
Yorktown Heights, NY 10598
(hugo@watson.ibm.com)

Abstract. We present simple and efficient hash functions applicable to secure authentication of information. The constructions are mainly intended for message authentication in systems implementing stream cipher encryption and are suitable for other applications as well. The proposed hash functions are implemented through linear feedback shift registers and therefore attractive for hardware applications. As an example, a single 64 bit LFSR will be used to authenticate 1 Gbit of information with a failure probability of less than 2^{-30}. One of the constructions is the cryptographic version of the well known cyclic redundancy codes (CRC); the other is based on Toeplitz hashing where the matrix entries are generated by a LFSR. The later construction achieves essentially the same hashing and authentication strength of a completely random matrix but at a substantially lower cost in randomness, key size and implementation complexity. Of independent interest is our characterization of the properties required from a family of hash functions in order to be secure for authentication when combined with a (secure) stream cipher.

1 Introduction

In this paper we deal with the application of traditional hashing techniques (*not* one-way hashing) to cryptographic authentication of information. This investigation was initiated by Carter and Wegman [21], and further developed in subsequent works [4, 6, 19, 11, 2, 7]. We concentrate, for the sake of clarity, in the case of *message authentication*, although these techniques have broader application to different scenarios of information authentication. We assume a typical communication scenario in which two parties communicate over an unreliable link where messages can be maliciously altered. The communicating parties share a secret key unknown to the adversary.

There are two basic approaches for the application of hashing in the message authentication scenario. Both approaches use a predetermined family of hash functions from which a particular function is secretly chosen by the parties for authentication. In the first case a new hash function is selected from the family for each transmitted message. In the second, the same hash function is applied to the authentication of multiple messages, but the resultant hash values are encrypted before transmission. The advantage in the second case is that it usually requires less random (or pseudo-random) bits, and that it allows for a less frequent (possibly off-line) generation of the specific hash function.

Y.G. Desmedt (Ed.): Advances in Cryptology - CRYPTO '94, LNCS 839, pp. 129-139, 1994.

Our work presents *very simple and practical* constructions of hashing schemes applicable to both approaches. Nevertheless, they are especially advantageous in the second case, i.e., for authentication of multiple messages, when the encryption is done via (additive) stream cipher encryption. We show these constructions to be *unconditionally secure* when used with a *perfect* one-time pad system, and therefore in most practical applications their security reduces to that of the stream cipher in use.

We start by proving a theorem on the minimal conditions required from a hash family in order to provide secure authentication when combined with one-time pad encryption, and later use this theorem to prove the security of our constructions. The characterization result is interesting independently of these specific constructions.

Our emphasis is in providing practical and secure methods for information authentication in systems that implement a stream cipher cryptosystem. We note that systems using block ciphers for secrecy often take advantage of these same ciphers for implementing a message authentication function. Additive stream ciphers, however, characterized by the use of pseudorandom generators which are essentially decoupled from the data, cannot directly be used to compute an authenticator on the data. On the other hand, the use of authentication in systems with stream cipher encryption is crucial because of the easy malleability of the plaintext through the corresponding ciphertext (e.g., flipping ciphertext bits is equivalent to flipping the same bits in the plaintext). In addition, authentication is required for validation of correct decryption and for detection of key synchronization loss. An important aspect related to stream ciphers is that these systems are often chosen for implementation (especially in hardware) because of their simplicity and efficiency. In such a case an equally simple and efficient authentication algorithm is required. *Our constructions are intended to fill this need.*

The first scheme we analyze is the cryptographic version of the well-known cyclic redundancy codes (CRC) used for non-cryptographic detection of information errors. It is based on the same operation of polynomial modular division and retains most of the simplicity of the regular CRC's except that in our case the dividing polynomial is variable. We prove the construction to be secure for authentication using the general results developed in Section 2.

The second construction is based on the well-known hashing technique that multiplies the data (seen as a binary vector) by a random matrix [5, 6]. This technique requires $m \cdot n$ random bits for specifying a hash function from m-bit messages into n-bit hash values which is prohibitive for many applications (e.g. large message and file authentication). Here, we show that essentially the same hashing and authentication effect can be achieved by choosing just n random bits and a random irreducible polynomial of degree n and then generating a Toeplitz matrix out of these initial values by a simple linear feedback shift register. In typical applications $n << m$ (n is the security parameter), and therefore the resultant savings (including number of random bits, size of hash description, key length) is enormous. It also makes possible the implementation of this technique in hardware as required in some authentication (and other hashing) applications.

Our constructions achieve up to small constants the known lower bounds on size

of description and number of functions in the hash family. This is an important factor for the practicality of these constructions as they influence, for example, the amount of hardware required in their implementation. As an example, a single 64 bit LFSR will be used to authenticate 1 Gbit of information with a failure probability of less than 2^{-30}. (We stress that both of our constructions can be applied to variable length messages).

We also mention that it is possible to extend our LFSR-based Toeplitz construction to general *ε-biased sequences* [15, 1] of which LFSRs are a particular case.

RELATED WORK. Unconditionally secure authentication codes have been extensively studied in the literature (see [18] for a survey). Pioneering works were Gilbert, MacWilliams and Sloane [8], Carter and Wegman [21] and the foundational work by Simmons (see [17, 18]). Carter and Wegman were the first to interpret and construct authentication codes through hash functions. They also were first to show how one-time pad systems can be used in combination with hash functions to construct efficient authentication algorithms. This approach was further studied by Brassard [4] and Desmedt [6]. Our work follows and refines this line of research.

The hashing approach for unconditional authentication was further developed by Stinson [19] who presents improved constructions and lower bounds on the size of the required hash families. More recently, Bierbrauer, Johansson, Kabatianskii and Smeets [11, 2] and Gemmell and Naor [7] generalize and improve on the above works by noticing and exploiting the connection between hash functions and error correcting codes.

Works that directly use stream ciphers in their constructions of cryptographic checksums are Lai, Rueppel and Woollven [12] and Taylor [20]. Contrary to our approach, [12] uses for the checksum computation a number of pseudorandom bits *equal* to the number of message bits. In our construction the number of pseudorandom bits depends linearly on the security parameter (usually much smaller than the information size) and grows only *logarithmically* with the message length. The approach in [20] is essentially to generate a new member of a hash family for each message using the pseudorandom generator of the stream cipher. In our case, we reuse the same hash function for multiple messages and use the pseudorandom generator only for generating encryption pads for the hash values.

A very recent and independent paper by Johansson [10] uses a LFSR to generate an authentication matrix. However, that construction, as well as its goals, is essentially different from ours. Most notably, it requires a LFSR (and then also a key) of the length of the message itself as opposed to just the length of the security parameter as in our case.

2 Hash Functions and Message Authentication

In this section we introduce the basic concepts regarding hash functions as required in the context of message authentication, and the notion of security for the authentication functions. We also characterize the exact requirements from a hash family to be secure when combined with a one-time pad system.

2.1 Hash Functions

Definition 1. An (m, n)-*family* H *of hash functions* is a collection of functions that map the set of binary strings of length m into the set of binary strings of length n.

Notation: The notation $s \in_R S$ denotes that the element s is chosen with uniform probability from the set S. The expression $Pr_h(A(h))$ denotes the probability of the event $A(h)$ when $h \in_R H$, and H is a (usually implicit) set of hash functions. We will use M, M', etc., to denote arguments or inputs for the functions in the family H. When it is clear from the context, and for the sake of readability, we will omit explicit reference to the lengths of these inputs. We will usually denote the output of the hash functions by c.

A property of some hash functions that simplifies their analysis is being linear relative to the bitwise exclusive-or operation. This property is found in many natural constructions (including ours).

Definition 2. A family of functions H is $\oplus$-*linear* if for all M, M' we have $h(M \oplus M') = h(M) \oplus h(M')$.

The following property of a family of hash functions has a central role in our work, it states that elements are mapped into their images by these functions in a "balanced" way. Its importance in our context is given by Theorem 6.

Definition 3. A family of hash functions is called ε-*balanced* if

$$\forall M \neq 0,\ c,\quad Pr_h(h(M) = c) \leq \varepsilon.$$

2.2 Message Authentication

We assume a typical communication scenario in which two parties communicate over an unreliable link where messages can be maliciously altered. The communicating parties share a secret key unknown to the adversary.

For simplicity we start assuming that the parties exchange (using that secret key) only one message of length m. In that case, the secret key consists of the description of a particular hash function h drawn randomly from an (m, n)-family of hash functions and a random pad r of length n. The sender of the message M, sends M together with the "tag" $t = h(M) \oplus r$, which at reception will be recomputed and checked for consistency by the receiver.

Although in the above scheme the authentication tag looks completely random to an adversary, and therefore it learns nothing about the specific hash function, it still can use its knowledge of the hash family to try to modify consistently the message and corresponding tag such that the message alteration is not discovered. Indeed, if the family H of hash functions is not chosen carefully such an attack *is possible* (see Section 3.1). Here we derive a necessary and sufficient condition on H to make the success probability of any attack no more than a pre-specified value ε.

Let M be a message of length m authenticated with the tag $t = h(M) \oplus r$, where $h \in_R H$ and $r \in_R \{0,1\}^n$. We say that an adversary that sees M and t succeeds in

breaking the authentication if it finds M' and t', where M' is different than M and $t' = h(M') \oplus r$. We assume that the adversary knows the family of hash functions, but not the particular value of h or the pad r.

Definition 4. A family H of hash functions is called *ε-otp-secure* if for any message M no adversary succeeds in the above scenario with probability larger than ε. [1]

Usually the value of ε for a given family of hash functions will depend on the parameters of this family (e.g. input and output size). The following theorem characterizes those families of hash functions that are ε-otp-secure.

Theorem 5. *A necessary and sufficient condition for a family H of hash functions to be ε-otp-secure is that*

$$\forall M_1 \neq M_2,\ c,\quad Pr_h(h(M_1) \oplus h(M_2) = c) \leq \varepsilon.$$

Proof Sketch. The pair (M_1, t_1) is successfully replaced by the pair (M_2, t_2) only if for the secret and random h and r used by the communicating parties we have $t_1 = h(M_1) \oplus r$ and $t_2 = h(M_2) \oplus r$, or equivalently, $t_1 \oplus t_2 = h(M_1) \oplus h(M_2)$. Therefore, the success probability of the adversary is bounded by $\max_{M_1, M_2, c} Pr_h(h(M_1) \oplus h(M_2) = c)$ where c represents the difference $t_1 \oplus t_2$. Notice that this success probability is achievable whenever the transmitted message is one of the messages in which the maximum is attained (just replace (M_1, t_1) by $(M_2, t_1 \oplus c)$). □

As an immediate consequence we have the following theorem that is the main tool for proving the security of our constructions.

Theorem 6. *If H is $\oplus$-linear then H is ε-otp-secure if and only if H is ε-balanced.*

Therefore, in order to prove the security of a particular family of hash functions for implementation of a message authentication scheme of the above kind (i.e., combined with a one-time pad) it is sufficient (and necessary) to show that the family has the condition stated in Theorem 5. In case the family is also $\oplus$-linear one has to prove it to be ε-balanced.

In the typical scenario where the parties exchange multiple messages, the hash function h can be reused for the different messages, but for each new message a different random pad (each of length n) will be used for encryption of the hash value. In this case, if the hash family is ε-otp-secure then the success probability of an adversary that tries to modify a single message is still at most ε. Indeed, the fact that the adversary sees many pairs of messages and corresponding tags is useless since these tags are completely random and therefore give no information on the value of h. If the adversary can modify k of the transmitted messages its probability of success is bounded by $k\varepsilon$.

When authenticating multiple messages with the same hash function, it is desirable that this function be applicable to variable length messages (as is the case

[1] We choose the term *otp-security* to stress the essential role of the one-time pad (otp) added to the hash value for the security of the authentication scheme.

for our constructions). In this case the adversary's success probability depends on the total length of messages he or she modifies.

Remark: The above definitions of security are stated in unconditional terms (i.e., against *any adversary*). This requires the communicating parties to exchange truly random pads (of the size of the hash output) for each transmitted message. In most practical applications, however, the successive pads r will be generated using a pseudorandom generator out of a secret seed shared by the parties. In this case the security of the authentication scheme reduces to the security of the pseudorandom generator or stream cipher in use; and the computational power of the adversary is assumed to be bounded depending on the security model of the stream cipher.
From a practical point of view our approach to message authentication is especially advantageous in systems implementing stream cipher encryption of the transmitted information. In these cases the hash value computed on the message is appended to the message before transmission and the combined information is then encrypted using the stream cipher.

3 Constructions

We present two simple and practical constructions of $\oplus$-linear hash functions that are ϵ-balanced with ϵ being exponentially small in the length of the hash value, and therefore suitable for information authentication in the sense described in Section 2. Both schemes can hash variable length messages.

3.1 Cryptographic CRC

The first construction is based on the operation of division modulo an irreducible polynomial over GF(2). It is a cryptographic variant of the well known Cyclic Redundancy Codes (CRC) which are commonly used as a standard error detection mechanism in data networks. CRC's are used to detect non-malicious errors and therefore there is no need for a secret key or even a family of functions; they are implemented as a fixed, public function. The simplicity of implementation and provable properties of these constructions have made them so popular; many of these advantages are inherited by the stronger cryptographic version.

To our knowledge, the first to use these functions in the cryptographic setting was Rabin [16] who proposed their use for fingerprinting information. As opposed to Rabin's application where the fingerprint value is kept secret, our setting requires the transmission of this value. Interestingly enough, even if one encrypts the fingerprint before transmissiom using a *perfect* one-time pad that scheme is insecure for message authentication. The construction presented here introduces a seemingly minor technical modification that solves that problem.

In what follows we will explicitly or implicitly associate each binary string S with the polynomial $S(x)$ over GF(2) with coefficients corresponding to the bits of S.

THE CONSTRUCTION *(Cryptographic CRC)*: We define an (m, n)-family of hash functions as follows. For each irreducible polynomial $p(x)$ of degree n over GF(2) we associate a hash function h_p such that for any message M of binary length m, $h_p(M)$ is defined as (the coefficients of) $M(x) \cdot x^n \bmod p(x)$.

Rabin's construction is essentially the same except for the multiplication by the x^n factor in the modular operation (that has the practical effect of shifting the message by n positions). Without this change the resultant hash family is only 1-balanced, and therefore breakable with probability 1. (The flipping of any bits among the n least significant bits of the message and the same bits in the encrypted authenticator will be undetected; even if the message itself is encrypted under a one-time pad!). After its modification we can prove that the proposed scheme is secure for authentication when combined with a one-time pad or secure stream cipher.

Theorem 7. *For any values of n and m the above defined family of hash functions is $\oplus$-linear and ε-balanced for $\varepsilon \leq \frac{m+n}{2^{n-1}}$, and therefore ε-otp-secure.*

Proof. The CRC family is $\oplus$-linear since division modulo a polynomial is a linear operation where addition is equivalent to a bitwise exclusive-or operation. To show that the family is also ε-balanced notice that for any polynomial $p(x)$ of degree n, any non-zero message M of length m and any string c of length n,

$$h_p(M) = c \text{ iff } M(x) \cdot x^n \bmod p(x) = c(x) \text{ iff } p(x) \text{ divides } M(x) \cdot x^n - c(x).$$

Denote $q(x) = M(x) \cdot x^n - c(x)$. Clearly, $q(x)$ is a non-zero polynomial of degree (at most) $m + n$, and $p(x)$ is an irreducible polynomial of degree n that divides $q(x)$. Because of the unique factorization property there are at most $\frac{m+n}{n}$ irreducible factors of $q(x)$ each of degree n. In other words, there are at most $\frac{m+n}{n}$ hash functions in the CRC family that map M into c. On the other hand, there are more than $\frac{2^{n-1}}{n}$ elements in this family (as the number of irreducible polynomials over GF(2) of degree n). Therefore, $Prob(h_p(M) = c) \leq \frac{(m+n)/n}{2^{n-1}/n} = \frac{m+n}{2^{n-1}}$. □

The practical consequence of this theorem and Theorem 6 is that one can safely use this very practical method of hashing for cryptographic authentication when combined with a cryptographically strong pseudorandom generator (i.e., secure stream cipher). We briefly consider some practical aspects of this construction.

VARIABLE-LENGTH MESSAGES. The hash functions in the CRC family are essentially defined by the polynomial $p(x)$ and not by the length of the messages. Therefore, they can be applied to messages of different lengths as it is desirable in practice. In this case, one has to treat the polynomial $M(x)$ corresponding to the message M as having a leading coefficient '1' (i.e., if M is of length m, then $M(x)$ is of proper degree m). This determines a 1-1 mapping between messages and polynomials and, in particular, prevents changing the message by just appending zeros to it. Also, the value of ε in Theorem 7 depends on m being the maximum size of the fake message inserted by the adversary rather than by the length of the original message.

HARDWARE IMPLEMENTATION. Implementing the above hash functions in hardware is simple and very efficient. The operation of division modulo a polynomial

over GF(2) is implemented through a simple linear feedback shift register with taps or connections determined by the dividing polynomial. Since this same operation is used for standard CRC's there are plenty of references in the literature on its implementation. (Even the multiplication by x^n in our construction is implemented in many cases without penalty in hardware or performance). However, recall that in the standard CRC the dividing polynomial is fixed and known in advance, and most circuits that implement it have the particular taps hardwired into the circuit. A cryptographic CRC as proposed here needs an implementation where the connections (determined by the polynomial) are programmable. The actual value for these connections is the key for the hashing which should be changeable (and secret). We stress that CRC circuits with variable connections are already designed even for implementation of regular CRC's. One reason for that is the need to support different CRC standards (each one determines a different polynomial), and in particular different polynomial degrees. See [3] for one such example.

SOFTWARE IMPLEMENTATION. Efficient implementations of CRC's in software exist too. In these implementations significant speed up is achieved by using precomputation tables. These tables depend on the particular key polynomial. Therefore, they are computed only once per key which is affordable in many applications.

CHOOSING KEYS. The keys for the cryptographic CRC functions is a random irreducible polynomial. The time complexity of generating such a polynomial of degree n is $O(n^3)$ bit operations or, in a software implementation, $O(n^2)$ word operations (mostly XOR's and SHIFT's). Therefore, it is efficient enough for applications (as suggested here) where the key is changed only sporadically (e.g. at the beginning of a network session). Algorithms for generating random irreducible polynomials can be found in [9, 16].

A NOTE OF CARE. Efficient stream ciphers (especially in hardware) sometimes use constructions based on LFSRs. In these cases using a cryptographic CRC is especially attractive because of the similar hardware structure. However, the security of these stream ciphers (as any other encryption system) is claimed only heuristically. Therefore, special care and attention need to be devoted to the interaction between these constructions.

3.2 LFSR-based Toeplitz

Our second construction is based on the following hashing method that uses random binary matrices. Let A be an $n \times m$ Boolean matrix. Let M be a message consisting of m bits. Define $h_A(M)$ to be the Boolean multiplication of the matrix A by the column vector composed of M's bits. Carter and Wegman [5] showed that the family of functions $\{h_A : A \text{ is a } n \times m \text{ Boolean matrix }\}$, is a universal$_2$ family of hash functions. This family is according to our terminology ε-balanced for $\varepsilon = 2^{-n}$, and then its affine version (namely, $h'_{A,b}(M) = A \cdot M + b$, where b is a binary vector of length n) is strongly universal$_2$.

The description of such a hash function takes $n \cdot m$ bits (or $n(m+1)$ in the affine case). A related family with the same properties as above but with much smaller description is obtained by restricting the Boolean matrix A to be a *Toeplitz matrix*.

These are matrices where each left-to-right diagonal is fixed, i.e., if $k - i = l - j$ for any indices $1 \leq i, k \leq n$, $1 \leq j, l \leq m$, then $A_{i,j} = A_{k,l}$. (For a proof of the universality of these hash functions see e.g. [14]). Notice that by defining the first column and first row of the matrix all other entries are uniquely determined. Therefore, only $m+n-1$ bits define the whole matrix, a significant savings relative to the $n \cdot m$ bits necessary to describe the original family. However, even these functions require a description that is as long as the input (or message) to be hashed. When these inputs are significantly longer than the required output (i.e., the security parameter) then this description can be prohibitively expensive.

Our construction modifies the above Toeplitz family by restricting it even more. Indeed, we use Toeplitz matrices where consecutive columns are the consecutives states of a LFSR of length n. To see that such a construction is indeed a Toeplitz matrix just notice that Toeplitz matrices are characterized by the property that each column in the matrix is determined by shifting (down) the previous column and adding a new element to the top of the column. Therefore in our construction a function is specified by defining a particular LFSR (i.e., its connection polynomial) and its initial state, a total of $2n$ bits (recall that usually $n << m$). Interestingly enough, by limiting the connections of the LFSRs to irreducible polynomials our construction keeps most of the strength of the original Carter-Wegman family but with a much shorter description size. The price we pay is that our functions are only ε-balanced for a small ε instead of being perfectly balanced as the original. For the purpose of authentication this small ε represents no substantial loss, while the lower description size makes them significantly more practical.

THE CONSTRUCTION *(LFSR-based Toeplitz)*: Let $p(x)$ be an irreducible polynomial over GF(2) of degree n. Let $s_0, s_1, \ldots$ be the bit sequence generated by a LFSR with connections corresponding to the coefficients of $p(x)$ and initial state $s_0, s_1, \ldots, s_{n-1}$. For each such polynomial $p(x)$ and initial state $s \neq 0$ we associate a hash function $h_{p,s}$ such that for any message $M = M_0 M_1 \ldots M_{m-1}$ of binary length m, $h_{p,s}(M)$ is defined as the linear combination $\bigoplus_{j=0}^{m-1} M_j \cdot (s_j, s_{j+1} \ldots s_{j+n-1})$.

In simple words, the LFSR advances its state with each message bit. If this bit is '1' the corresponding state is accumulated into an accumulator register, if it's '0' the state is not accumulated (see Figure 1).

The main technical theorem regarding this construction is the following characterization of LFSR-based Toeplitz hashing. The proof is omitted from this abstract.

Theorem 8. *Let $p(x)$ be an irreducible polynomial of degree n over GF(2) and let $s = (s_0, \ldots, s_{n-1})^T$ be an initial state for the LFSR defined by the connection polynomial $p(x)$. Let M be an m-bit long message. Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be the n (different) roots of $p(x)$ (over GF(2^n)). Then,*

$$h_{p,s}(M) = B D_{M,p} B^{-1} s$$

where B is a non-singular $n \times n$ matrix which depends on $p(x)$ only and $D_{M,p}$ is an $n \times n$ diagonal matrix with $M(\lambda_i)$, $1 \leq i \leq n$, as its i-th diagonal entry.

From this we can derive that our construction has the required ε-balanced property.

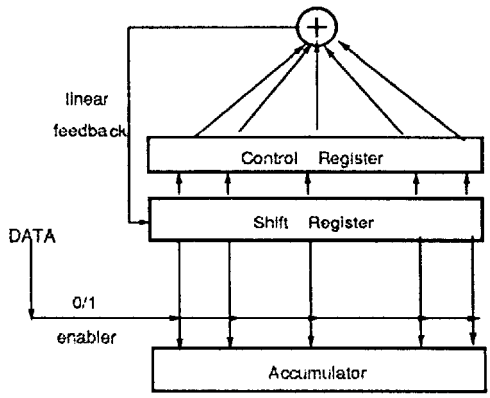

Fig. 1. A schematic implementation of the LFSR-based Toeplitz hashing

Theorem 9. *The LFSR-based Toeplitz construction defined above is ε-balanced for $\varepsilon \leq \frac{m}{2^{n-1}}$.*

Proof. Fix a message $M \neq 0$ and a hash value c. We need to bound the probability that $h_{p,s}(M) = c$ for randomly chosen irreducible polynomial $p(x)$ and initial state $s \neq 0$. We use Theorem 8 and the fact that $M(x)$ has a common root with $p(x)$ if and only if $p(x)$ divides $M(x)$. We distinguish between two cases according to the value of c.

Case I: Let $c = 0$ (i.e., c is the all-zeros vector). Since we choose $s \neq 0$ then $h_{p,s}(M) = 0$ may happen only if $D_{M,p}$ is singular (the matrices B and B^{-1} are not). This is the case only if for some i, $M(\lambda_i) = 0$, or equivalently only if $p(x)$ divides $M(x)$. The probability of such an event is at most as the number of possible irreducible factors of $M(x)$ divided by the total number of irreducible polynomials of degree n, i.e. at most $\frac{m/n}{2^{n-1}/n} = \frac{m}{2^{n-1}}$.

Case II: Let $c \neq 0$. In order for $h_{p,s}(M)$ to equal c, we need $D_{M,p}$ to be non-singular and s be the *unique* vector that is mapped by $BD_{M,p}B^{-1}$ into c. The vector s assumes this value with probability of $\frac{1}{2^{n-1}}$, and therefore $h_{p,s}(M) = c$ happens with at most this probability.

In either case the probability that $h_{p,s}(M) = c$ is at most $\frac{m}{2^{n-1}}$ and then our construction is $\frac{m}{2^{n-1}}$-balanced. □

PRACTICAL CONSIDERATIONS. Most of the remarks in Section 3.1 regarding practical implementation of CRC's hold here. We just stress that the decoupling of the LFSR from the data and having a shift register without modifications between internal stages (see Figure 1) permits significant parallelism and pipelining in the LFSR implementation which is crucial for achieving very high speeds and can be advantageous relative to the CRC construction. Notice that the LFSR-based construction can be used also for the same non-cryptographic applications where the CRC is regularly encountered.

Acknowledgment: I would like to thank Phil Rogaway, Ronny Roth and Moti Yung for their help and comments.

References

1. Noga Alon, Oded Goldreich, Johan Hastad, and Rene Peralta. Simple constructions of almost k-wise independent random variables. In 31^{th} *Annual Symposium on Foundations of Computer Science, St. Louis, Missouri*, pages 544–553, October 1990.
2. Bierbrauer J., Johansson T., Kabatianskii G., and Smeets, B., "On Families of Hash Functions via Geometric Codes and Concatenation", *Proc. of Crypto'93*, pp. 331-342.
3. Birch, J., Christensen, L.G., and Skov, M., "A programmable 800 Mbit/s CRC check/generator unit for LANs and MANs", *Comp. Networks and ISDN Sys.*, 1992.
4. Brassard, G., "On computationally secure authentication tags requiring short secret shared keys", *Proc. of Crypto'82*, pp.79-86.
5. Carter, J.L. and Wegman, M.N., "Universal Classes of Hash Functions", *JCSS*, 18, 1979, pp. 143-154.
6. Desmedt, Y., "Unconditionally secure authentication schemes and practical and theoretical consequences", *Proc. of Crypto'85*, pp.42-55.
7. Gemmell, P., and Naor, M., "Codes for Interactive Authentication", *Proc. of Crypto'93*, pp. 355-367.
8. Gilbert, E.N., MacWilliams, F.J., and Sloane, N.J.A., "Codes which detect deception", *Bell Syst. Tech. J.*, Vol. 53, 1974, pp. 405-424.
9. John A Gordon, "Very simple method to find the minimal polynomial of an arbitrary non-zero element of a finite field", *Electronics Letters* , Vol. 12, 1976, pp. 663-664.
10. Johansson T., "A Shift Register Construction of Unconditionally Secure Authentication Codes", *Design, Codes and Cryptography*, 4, 1994, pp. 69-81.
11. Johansson T., Kabatianskii G., and Smeets, B., "On the Relation Between A-Codes and Codes Correcting Independent Errors", *Proc. of Eurocrypt'93*, pp. 1-11.
12. Lai, X., Rueppel, R.A., and Woollven, J., "A Fast Cryptographic Checksum Algorithm Based on Stream Ciphers", *Auscrypt'92*, pp. 339-348.
13. Lidl, R., and Niederreiter, H., "Finite Fields", in *Encyclopedia of Mathematics and Its Applications*, Vol 20, Reading, MA: Addison-Wesley, 1983.
14. Mansour, Y., Nisan, N., and Tiwari, P., "The Computational Complexity of Universal Hashing", *STOC'90*, pp. 235-243.
15. Joseph Naor and Moni Naor. Small bias probability spaces: efficient construction and applications. In *Proceedings of the* 22^{nd} *Annual ACM Symposium on Theory of Computing, Baltimore, Maryland*, pages 213–223, May 1990.
16. Rabin, M.O., "Fingerprinting by Random Polynomials", Tech. Rep. TR-15-81, Center for Research in Computing Technology, Harvard Univ., Cambridge, Mass., 1981.
17. Simmons, G.J., "Authentication theory/coding theory", *Proc. of Crypto'84*, 411-431.
18. Simmons, G.J., "A Survey of Information Authentication", in Gustavos J. Simmons, editor, *Contemporary Cryptology, The Science of Information*, IEEE Press, 1992.
19. Stinson, D.R., "Universal hashing and authentication codes", *Proc. of Crypto'91*, pp. 74-85.
20. Taylor, R., "An integrity check value algorithm for stream ciphers", *Proc. of Crypto'93*, pp. 40-48.
21. Wegman, M.N., and Carter, J.L., "New Hash Functions and Their Use in Authentication and Set Equality", *JCSS*, 22, 1981, pp. 265-279.