Cryptographic Hash Functions

Bart Preneel ([1])
Katholieke Universiteit Leuven, Laboratorium ESAT-COSIC
K. Mercierlaan 94, B-3001 Heverlee, Belgium

Abstract. Hash functions were introduced in cryptology in the late seventies as a tool to protect the authenticity of information. Soon it became clear that they were a very useful building block to solve other security problems in telecommunication and computer networks. This paper sketches the history of the concept, discusses the applications of hash functions, and presents the approaches that have been followed to construct hash functions. In addition, it tries to provide the information which is necessary to choose a practical hash function. An overview of practical constructions and their performance is given and some attacks are discussed. Special attention is paid to standards dealing with hash functions.

1. INTRODUCTION

During the last decades, the nature of telecommunications has changed completely. Telecommunications more and more pervades every aspect of society. Recent developments in mobile telecommunications like the GSM system [127] make it possible to reach a person anywhere in Europe, independent of whether he is at home, in his office, or on the road. Electronic mail has become the preferable way of communication between researchers all over the world, and many companies have introduced this service. At the same time EDI (Electronic Data Interchange) is being introduced in order to extend the automatic information processing within a company to suppliers and clients into a single system. Home banking is becoming more and more popular and is the first step toward shopping from the home.

This evolution of telecommunications presents new security requirements, posing new challenges to the cryptologists. Handwritten letters offer reasonable privacy protection and the receiver can be sure of the authenticity, which encompasses two aspects: he knows who the sender is and he knows that the content has not been modified. Voice communications can be eavesdropped easily, but at least they offer a guarantee of the authenticity of the communication: one is sure that one is talking to a specific person, and that the conversation is not being modified. Electronic data communications however offer no protection of privacy or authenticity. An additional challenge is that it is not sufficient to design solutions for closed user groups, since one often requires worldwide systems that work in a variety of environments.

In this overview paper, we will discuss hash functions, which form an important cryptographic technique to protect the authenticity of information. The paper is organized as follows. Section 2 situates hash functions within the wider area of cryptology and discusses the application of hash functions to several problems. In section 3, we define three types of hash functions. Section 4 summarizes the main theoretical results on hash functions. Three approaches in cryptography are considered: the information theoretic approach, the complexity theoretic approach, and the system based or practical approach. Section 5 and section 6 give an overview of practical proposals for MDC's and MAC's respectively, and section 7 deals with their software performance. Section 8 presents the conclusions.

2. AUTHENTICATION AND PRIVACY

This section discusses the basic concepts of cryptography and clarifies the importance of hash functions in the protection of information authentication. At the end of this section other applications of hash functions are presented.

([1]) NFWO postdoctoral researcher, sponsored by the National Fund for Scientific Research (Belgium). Part of this work was done while visiting the EECS Department of the University of California at Berkeley.

2.1. *Privacy protection with symmetric cryptology*

Cryptology has been used for thousands of years to protect communications of kings, soldiers, and diplomats. Until recently, the protection of communications was almost a synonym for the protection of the secrecy of the information, which is achieved by encryption. In the encryption operation, the sender transforms the message to be sent, which is called the plaintext, into the ciphertext. The encryption algorithm uses as parameter a secret key; the algorithm itself is public, which is known as Kerckhoffs's principle. The receiver can use the decryption algorithm and the same secret key to transform the ciphertext back into the plaintext. An eavesdropper, who has no access to the secret key, will not be able to compute the plaintext from the ciphertext. The main idea of encryption is to replace the secrecy of a large amount of data by the secrecy of a short key that can be communicated via a secure channel (Fig. 1). Because the keys for encryption and decryption are equal, this approach is called symmetric cryptography.

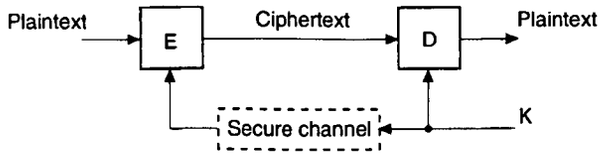

Fig. 1 - Model of a symmetric encryption system.

It was widely believed that protection of the authenticity would follow automatically from protection of the secrecy: if the receiver obtains a "meaningful" plaintext, he can be sure that the sender with whom he shares the key has actually sent this message. This belief is wrong: "meaningful" plaintext can only be distinguished from "random" plaintext based on redundancy, which is not always present. Even if the plaintext has redundancy, modifications can sometimes be made which will escape detection. This holds especially for additive ciphers, where the ciphertext is obtained by adding a key stream modulo two to the plaintext: complementing a ciphertext bit results in a complementation of the corresponding plaintext bit. However, in the old days the authenticity was protected by the intrinsic properties of the communication channel.

The advent of electronic computers and telecommunication networks created the need for a widespread commercial encryption algorithm. In this respect, the publication in 1977 of the Data Encryption Standard (DES) by the U.S. National Bureau of Standards [39] was an important milestone. The DES was designed by IBM in cooperation with the US National Security Agency (NSA). It later became an ANSI banking standard [2]. At the same time the need for specific measures to protect the authenticity of the information became obvious, since it was realized that authenticity does not come for free with secrecy protection. The first idea to solve this problem was to add a simple form of redundancy to the plaintext before encryption, namely the sum modulo two of all plaintext blocks [61]. This showed to be insufficient, and techniques to construct redundancy that is a complex function of the complete message were proposed. It is not surprising that the first constructions were based on the DES.

2.2. *Authentication with symmetric cryptology*

In the military world it was known for some time that modern telecommunication channels like radio require additional protection of the authenticity. One of the techniques applied was to append a secret key to the plaintext before encryption [121]. The protection then relies on the error propagating properties of the encryption algorithm and on the fact that the secret key for authentication is used only once.

In the banking environment, there is a strong requirement for protecting the authenticity of transactions. Before the advent of modern cryptology, this was achieved as follows: the sender computes a simple function of the transaction totals and a secret key; the result, which was called the test key, is appended to the transaction. This allows the receiver of the message, who is also privy to the secret key, to verify the authenticity of the transaction.

Although both solutions are not suited for a wider and less restrictive environment, they form the embryonal stadium of the concept of hash functions.

New techniques were proposed to produce redundancy under the form of a short string that is a complex function of the complete message (Fig. 2). A function that compresses its input was already in use in computer science to allocate as uniformly as possible storage for the records of a file. It was called a *hash function*, and its result was called a *hashcode*. If a hash function has to be useful for cryptographic applications, it has to satisfy some additional conditions. Informally, one has to impose that the hash function is one-way (hard to invert) and that it is hard to find two colliding inputs, i.e., two inputs with the same output. If the information is to be linked with an originator, a secret key has to be involved in the hashing process (this assumes a coupling between the person and his key), or a separate integrity channel has to be provided. Therefore, two basic methods can be identified:

- The first approach is analogous to the approach of a symmetric cipher, where the secrecy of large data quantities is based on the secrecy and authenticity of a short key. In this case the authentication of the information will also rely on the secrecy and authenticity of a key. To achieve this goal, the information is compressed with a hash function, and the hashcode is appended to the information. The basic idea of the protection of the integrity is to *add re-*

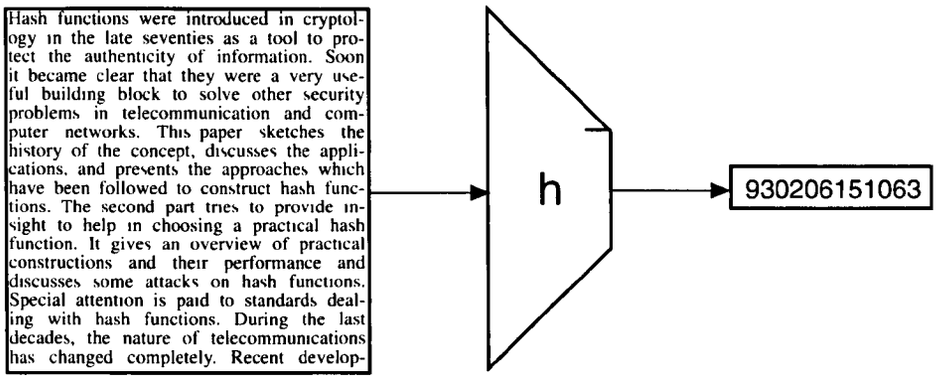

Fig. 2 - A hash function.

dundancy to the information. The presence of this redundancy allows the receiver to make the distinction between authentic information and bogus information.

In order to guarantee the origin of the data, a secret key that can be associated to the origin has to intervene in the process. The secret key can be involved in the compression process; the hash function is then called a Message Authentication Code or MAC. A MAC is recommended if authentication without secrecy is required. If the hash function uses no secret key, it is called a Manipulation Detection Code or MDC; in this case it is necessary to encrypt the hashcode and/or the information with a secret key. In addition, the encryption algorithm must have a strong error propagation: the ciphertext must depend on all previous plaintext bits in a complex way. Additive stream ciphers can definitely not be used for this purpose.

- The second approach consists of basing the authenticity (both integrity and origin authentication) of the information on the authenticity of a Manipulation Detection Code or MDC. A typical example for this approach is an accountant who will send the payment instructions of his company over an insecure computer network to the bank. He computes an MDC on the file, and communicates the MDC over the telephone to the bank manager. The bank manager computes the MDC on the received message and verifies whether it has been modified. The authenticity of the telephone channel is offered here by voice identification.

Note that the addition of redundancy is necessary but not sufficient. Special care has to be taken against high level attacks, like a replay of an authenticated message.

2.3. *Asymmetric or public-key cryptology*

From a scientific viewpoint, the most important breakthrough of the last decennia in cryptology is the invention of public-key cryptology in the mid seventies by W. Diffie and M. Hellman [38], and independently by R. Merkle [81]. Public-key cryptology has brought two important insights:

- Sender and receiver do not need to share a secret key: it is sufficient that they use an authentic channel to communicate a key.
- One can produce an electronic equivalent of a handwritten signature: the digital signature.

As a by-product of their results, it became clear that secrecy and authenticity are two independent properties of a cryptosystem: if the encryption key is public, anyone can use it to send an enciphered message to a certain receiver. Protection of the authenticity of the information is possible, but this requires a second independent operation.

There are several reasons why conventional techniques are still widely used in spite of the development of public-key cryptology. The most important one is certainly that no efficient public-key cryptosystems are known. In the first years after the invention of public-key cryptosystems, serious doubts have been raised about their security. A good example is the rise and fall of the knapsack-based schemes, which were very attractive because of their good performance. Unfortunately, almost all public-key cryptosystems based on knapsacks were shown to be insecure [13, 37]. Other schemes have been more successful: the Diffie-Hellman scheme, proposed in 1976, is widely used for key agreement, and the RSA scheme proposed by R. Rivest, A. Shamir, and L. Adleman in 1978 [110] is used for both digital signatures and public-key encryption. However, it has taken more than 10 years before these schemes have reached the market. The disadvantages of both schemes are that they are two to three orders of magnitude slower than all conventional systems, and that the key and block size are about 10 times larger.

Soon it was realized that one could have the best of both worlds, i.e., more flexibility, a less cumbersome key management, and a high performance, by using hybrid schemes. One uses public-key techniques for key establishment, and subsequently a conventional algorithm like DES or triple-DES to encipher large quan-

tities of data. If one wants to take a similar approach to authenticity protection, one can use cryptographic hash functions as follows: one first compresses the data with a fast hash function to a short string of fixed length. The slow digital signature scheme is then used to protect the authenticity of the hashcode. These hybrid techniques have extended the applicability of public-key cryptography. The choice between symmetric and public-key techniques is application dependent: while public-key seems to be the choice for worldwide protection of email, symmetric techniques have been selected for the GSM system [127].

2.4. *Other applications of hash functions*

Hash functions have been designed in the first place to protect the authenticity of information. When efficient and secure hash functions became available, it was realized that under certain assumptions they can be used for many other applications. For some applications it is required that the hash function behaves as a "random" function. This implies that there is no correlation between input and output bits, no correlation between output bits, etc. The most important applications are the following:

- Protection of pass-phrases: passphrases are passwords of arbitrary length. One will store the MDC corresponding to the passphrase in the computer rather than the password itself.
- Construction of efficient digital signature schemes: this comprises the construction of efficient signature schemes based on hash functions only [82], as well as the construction of digital signature schemes from zero-knowledge protocols.
- Building block in practical protocols including entity authentication protocols, key distribution protocols, and bit commitment.
- Construction of encryption algorithms: while the first hash functions were based on block ciphers, the advent of fast hash functions has led to the construction of encryption algorithms based on hash functions. Some theoretical support for this construction can be found in the work of M. Luby and C. Rackoff [75] on randomizers.

3. DEFINITIONS

In section 2 two classed of hash functions have been introduced, namely Message Authentication Codes or MAC's (which use a secret key), and Manipulation Detection Codes or MDC's, which do not make use of a secret key. According to their properties, the class of MDC's will be further divided into one-way hash functions (OWHF) and collision resistant hash functions (CRHF). The relation between the different hash functions has been summarized in Fig. 3.

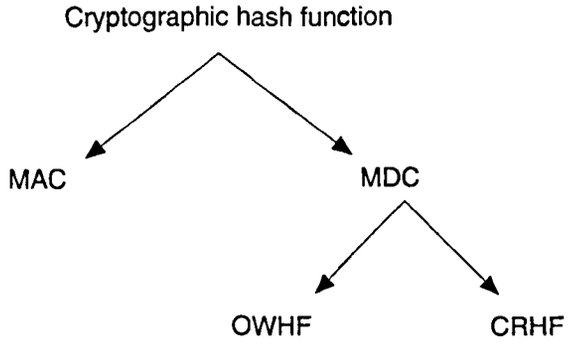

Fig. 3 - A taxonomy for cryptographic hash functions.

In the following the hash function will be denoted with h, and its argument, i.e., the information to be protected with X. The image of X under the hash function h will be denoted with $h(X)$. The general requirements are that the computation of the hashcode is "easy" if all arguments are known. Moreover it is assumed that the description of the hash function is public; for MAC's the only secret information lies in the secret key.

3.1. *One-way hash function (OWHF)*

The first informal definition of a OWHF was given by R. Merkle [81, 83] and M. Rabin [108].

Definition 1 A one-way hash function is a function h satisfying the following conditions:

1) The argument X can be of arbitrary length and the result $h(X)$ has a fixed length of n bits (with $n \geq 64$).
2) The hash function must be one-way in the sense that given a Y in the image of h, it is "hard" to find a message X such that $h(X) = Y$, and given X and $h(X)$ it is "hard" to find a message $X' \neq X$ such that $h(X') = h(X)$.

The first part of the second condition corresponds to the intuitive definition of one-wayness, namely that it is "hard" to find a preimage of a given value in the range. For permutations or injective functions only this part is relevant. The second part of this condition, namely that finding a second preimage should be hard, is a stronger condition, which is relevant for most applications. The meaning of "hard" still has to be specified. In the case of "ideal security", introduced by X. Lai and J. Massey [71], producing a (second) preimage requires 2^n operations. However, it may be that an attack requires a number of operations that is smaller than 2^n, but is still computationally infeasible.

3.2. *Collision resistant hash function (CRHF)*

The first formal definition of a CRHF was given by I. Damgård [26]; an informal definition was given by R. Merkle [83].

Definition 2 A collision resistant hash function is a function h satisfying the following conditions:

1) The argument X can be of arbitrary length and the result $h(X)$ has a fixed length of n bits (with $n \geq 128$).
2) The hash function must be one-way in the sense that given a Y in the image of h, it is "hard" to find a message X such that $h(X) = Y$, and given X and $h(X)$ it is "hard" to find a message $X' \neq X$ such that $h(X') = h(X)$.
3) The hash function must be collision resistant: this means that it is "hard" to find two distinct messages that hash to the same result.

Under certain conditions one can argue that the first part of the one-way property follows from the collision resistance [27]. Again several options are available to specify the word "hard". In the case of "ideal security" [71], producing a (second) preimage requires 2^n operations and producing a collision requires $O(2^{n/2})$ operations (section 4.3.2.). This can explain why both conditions have been stated separately. One can however also consider the case where producing a (second) preimage and a collision requires at least $O(2^{n/2})$ operations, and finally the case where one or both attacks require less than $O(2^{n/2})$ operations, but the number of operations is still computationally infeasible (e.g., if a larger value of n is selected).

The choice between a OWHF and a CRHF is application dependent. A CRHF is stronger than a OWHF, which implies that using a CRHF is playing safe. A OWHF can only be used if the opponent cannot exploit the collisions, e.g., if the argument is randomized before the hashing operation. On the other hand, it should be noted that designing a OWHF is easier, and that the storage for the hashcode can be halved (64 bits instead of 128 bits). A disadvantage of a OWHF is that the security level decreases with the number of applications of h: an outsider who knows s hashcodes has increased his probability to find an X' with a factor of s. This limitation can be overcome by using a parameterized OWHF.

3.3. *Message Authentication Code (MAC)*

Message Authentication Codes have developed from the test keys in the banking community. However, these algorithms did not satisfy this strong definition.

Definition 3 A MAC is a function h satisfying the following conditions:

1) The argument X can be of arbitrary length and the result $h(K, X)$ has a fixed length of n bits (with $n \geq 32 \ldots 64$).
2) Given h and X, it is "hard" to determine $h(K, X)$ with a probability of success "significantly higher" than $1/2^n$. Even when a large number of pairs $\{X_i, h(K, X_i)\}$ are known, where the X_i have been selected adaptively by the opponent, it is "hard" to determine the key K or to compute $h(K, X')$ for any $X' \neq X_i$. This last attack is called an adaptive chosen text attack.

Note that this last property implies that the MAC should be both one-way and collision resistant for someone who does not know the secret key K. This definition leaves open whether a MAC should be one-way or collision resistant for someone who knows K. An example where this property could be useful is the authentication of multi-destination messages [87].

4. THREE APPROACHES TO HASH FUNCTIONS

In this section a taxonomy for cryptographic hash functions will be presented. Our taxonomy deviates from the approach by G. Simmons [121], and is based on the taxonomy for stream ciphers of R. Rueppel [115]. A first approach is based on information theory, and it offers unconditional security, i.e., security independent of the computing power of an adversary. The complexity theoretic approach starts from an abstract model of computation and assumes that the opponent has limited computing power. The system based approach tries to produce practical solutions; the security estimates are based on the best algorithm known to break the system and on realistic estimates of the necessary computing power or dedicated hardware to perform the algorithm. In [121] the second and third approach are lumped together as computationally secure, and in [115] a fourth approach is considered, in which the opponent has to solve a problem with a large size (namely examining a huge publicly accessible random string); it can be considered as both computationally secure and information theoretically secure.

4.1. *Information theoretic approach*

This approach results in a characterization of unconditionally secure solutions, which implies that the security of the system is independent of the computing power of the opponent. E.g., in case of privacy protection, it has been shown by C. Shannon [119] that unconditional privacy protection requires that the entropy of the key is as least as large as the entropy of the plaintext. It should be remarked that both unconditional privacy and unconditional authenticity are only probabilistic: even if the system is optimal with respect to some definition, the opponent has always a non-zero probability of success. However, this probability can be made exponentially small. The advantage of this approach lies in the unconditional security. The main disadvantage is that the key material can be used only once (or a limited number of times).

The first result on unconditionally secure authentication appeared in 1974 in a paper by E. Gilbert, F. MacWilliams, and N. Sloane [45]. Subsequently the theory has been developed by G. Simmons, analogous to the theory of secrecy systems that was invented by C. Shannon [119]. An overview of this theory can be found in [120, 121]. In recent work by T. Johansson, G. Kabatianskii, and B. Smeets [60], important connections have been established between authentication codes and error correcting codes. From the brief but clear summary by J. Massey in [76] we cite the following statement *"The theory of authenticity is in many ways more subtle than the corresponding theory of secrecy. In particular, it is not at all obvious how "perfect authenticity" should be defined"*. This is caused by the fact that there are different bounds that can be met with equality.

We will restrict ourselves to authentication codes which offer no secrecy (Cartesian or systematic authentication codes) and which are deterministic. In this case there is a direct connection to a Message Authentication Code. It will also be assumed that message, key, and MAC are binary strings, with length in bits equal to m, k, and n respectively. For a more detailed overview, the reader is referred to [76, 102, 121, 122].

In the simplest model of an unconditionally secure authentication scheme, one has three players: the sender Alice, the receiver Bob, and the active eavesdropper Eve. Eve can perform three types of attacks:

- Eve can send a fraudulent cryptogram to Bob as if it came from Alice (*impersonation attack*).
- Eve can wait until she observes a cryptogram and replace it by a different cryptogram (*substitution attack*).
- Eve can choose freely between both strategies (*deception attack*).

The probability of success (when the strategy of Eve is optimal) will be denoted with P_i, P_s, and P_d respectively. A first result that follows from Kerckhoffs's assumption (namely that the strategy to choose the key is known by Eve) is that $P_d = \max(P_i, P_s)$. Extensions of this model are discussed in [121].

The following bounds have been established:

Combinatorial bound for impersonation: $P_i \geq 1/2^n$.

Combinatorial bound for substitution: $P_s \geq (2^m - 1)/(2^{m+n} - 1)$.

Authentication channel capacity: $P_i \geq 2^{-I(h(K,X);K)}$. Here $I(X; Y)$ denotes the mutual information between X and Y. For the shortest proof known until now and a discussion of the recent improvements on this bound by R. Johannesson and A. Sgarro the reader is referred to [76]. This bound has the following corollary: $P_d \geq 1/2^{k/2}$.

An authentication code is called *perfect* if equality holds in the equation for the authentication channel capacity. For a perfect authentication code where P_i and P_s meet the combinatorial bound, the number of key bits per message bit is at least n/m, which is much larger than 1 if m is small. Note that n cannot be small, since this would imply that P_i is large [76]. In [123], D. Stinson has given an overview of characterizations of this type of authentication codes. In [124], he has shown that if $P_i = P_s = 1/2^n$, the number 2^m of possible messages can grow at most linearly with the number 2^k of possible keys.

In order to increase the efficiency in terms of key usage, one has to allow for a larger value of P_s. This idea was first put forward by J. Carter and M. Wegman when they introduced the concept of a universal hash function [17]. The formal connection between universal hash functions and authentication codes was first established in [124]. It was shown in [60] that if P_s exceeds P_i by an arbitrarily small positive amount, the number of possible messages will grow exponentially with the number of possible keys.

A very elegant construction was proposed in [79] and independently in [36, 60]. The key consists of 2 elements of $GF(2^n)$ denoted with μ and ν. The argument X is split into t elements of $GF(2^n)$ denoted with $x_1, x_2, \ldots, x_t$, hence $m = t \cdot n$. The function is then defined as follows:

$$h(X) = \mu + \sum_{i=1}^{t} x_i \cdot \nu^i$$

where the addition and the multiplication are in $GF(2^n)$. It can be shown that this yields an authentication code with $P_i = 1/2^{-n}$ and $P_s = t/2^n$. If $m = n$ this construction reduces to the scheme of E. Gilbert, F. MacWilliams, and N. Sloane [45]. In [60] it was shown that this authentication code corresponds to the well-known Reed-Solomon (R-S) code of length 2^n. In addition, the authors of [60] showed that this scheme is asymptotically optimal in the sense that for a given value of P_s (which determines t), it will result in the maximal possible message length $m = t \cdot n$ for a given key length $k = 2n$.

If stronger conditions are imposed on P_s, it seems to be more difficult to find efficient constructions. The case $P_i = 1/2^n$ and $P_s \leq 2P_i$ was first studied in [128]. Subsequent improvements were made in [124] and [5] by using Reed-Solomon and Algebraic Geometry codes. For $n = 20$ and $m = 2^{28}$, the best known construction requires 100 key bits, while existence results in coding theory tell us that there exist codes that require only 52 key bits. Therefore, one can expect further progress in this direction.

If these schemes are used in a practical setting, it remains a disadvantage that a single key can be used to authenticate only one message; this can be avoided by encrypting the MAC with the Vernam scheme, which

means that n additional key bits per message are required. Other solutions are discussed in [102].

4.2. *Complexity theoretic approach*

The approach taken here is to define a model of computation, like a Turing machine [1] or a Boolean circuit. All computations in this model are parameterized by a security parameter, and only algorithms or circuits that require asymptotically polynomial time and space in terms of the size of the input are considered feasible. The next step is then to design cryptographic systems that are provably secure with respect to this model. This research program has been initiated in 1982 by A.C. Yao [130] and tries to base cryptographic primitives on general assumptions. Examples of *cryptographic primitives* are: secure message sending, cryptographically secure pseudo-random generation, digital signatures, and Collision Resistant Hash Functions (CRHF). Examples of *general assumptions* to which these primitives can be reduced are the existence of one-way functions, injections, or permutations, and the existence of trapdoor one-way permutations. A third aspect is the *efficiency of the reduction*, i.e., the number of executions of the basic function to achieve a cryptographic primitive, and the number of interactions between the players in the protocol.

An important research goal is to reduce cryptographic primitives to weaker assumptions, with as final goal to prove that the reduction is optimal. One can also try to improve the efficiency of a reduction, possibly at the cost of a stronger assumption. If someone wants to build a concrete implementation, he will have to choose a particular one-way function, permutation, etc. The properties of a particular problem that is believed to be hard can be used to increase the efficiency of the solutions. Examples of problems that have been intensively used are the factoring of a product of two large primes and the discrete logarithm problem modulo a prime and modulo a composite that is the product of two large primes.

The complexity theoretic approach has several advantages:

1) It results in *provably secure* systems, based on a number of assumptions.
2) The construction of such proofs requires *formal definitions* of the cryptographic primitives and of the security of a cryptographic primitive.
3) The *assumptions* on which the security of the systems is based are also defined formally.

The disadvantage is that the complexity theoretic approach has only a limited impact on practical implementations, due to limitations that are inherently present in the models:

1) In complexity theory, a number of operations that is *polynomial* in the size of the input is considered feasible, while a superpolynomial or exponential number of operations in the size of the input is infeasible. In an asymptotic setting, abstraction is made from both constant factors and the degrees of the polynomials. This implies that this approach gives no information on the security of concrete instances (a practical problem has a finite size): e.g., no distinction is made between an attack requiring $O(2^n)$ and an attack requiring $O(2^{n/2})$ operations. Secondly, the scheme might be impractical because the number of operations to be carried out is polynomial in the size of the input but impractically large.
2) The complexity theoretic approach yields only results on the *worst case or average case* problems in a general class of problems. However, cryptographers studying the security of a scheme are more interested in the subset of problems that is easy.
3) Complexity theory usually deals with *single isolated instances* of a problem. A cryptanalyst often has a large collection of statistically related problems to solve.

The most important results on authentication will be summarized briefly. M. Naor and M. Yung have introduced the concept of a Universal One-Way Hash Function (UOWHF) [94]. The philosophy behind a UOWHF is that first the input is selected and subsequently (and independently) the hash function. In this case it does not help an opponent to find collisions for the hash function (section 3.2.). They showed how to use a UOWHF to build a signature scheme. An important result is that it is sufficient to have a UOWHF that compresses a single bit to construct a UOWHF that compresses an arbitrary number of bits. Several authors have subsequently improved their construction. A key result by J.Rompel [114] is a (very inefficient) construction for a UOWHF based on any one-way function, which is the weakest possible assumption. I. Damgård has studied a CRHF in this setting [27]. A practical version of an important result is discussed in section 4.3.1.

4.3. *System based or practical approach*

In this approach schemes with fixed dimensions are designed and studied, paying special attention to the efficiency of software and hardware implementations. The goal of this approach is to ensure that breaking a cryptosystem is a difficult problem for the cryptanalyst.

By trial and error procedures, several *cryptanalytic principles* have emerged, and the designer intends to avoid attacks based on these principles. Typical examples are statistical attacks and meet-in-the-middle attacks.

The second aspect is to design *building blocks with provable properties*. These building blocks are not only useful for cryptographic hash functions, but also for the design of block ciphers and stream ciphers. Typical examples are statistical criteria, diffusion and confusion,

correlation, and nonlinearity criteria.

Thirdly, *the assembly of basic building blocks* to design a cryptographic hash function can be based on theorems. Results of this type are often formulated and proven in a complexity theoretic setting, but can easily be adapted to a more practical definition of "hardness" that is useful in a system based approach. This will be illustrated in the following section.

A general model for a hash function will now be presented, together with two important theorems and an overview of the most important attacks on hash functions.

4.3.1. General model

The general model allows for a compact description of specific constructions. Almost all known hash functions are based on a compression function with fixed size input; they process every message block in a similar way. This has been called an "iterated" hash function in [71]. The information is divided into t b-bit blocks X_1 through X_t. If the total number of bits is not a multiple of the block length, the information has to be padded to the required length. The hash function can subsequently be described as follows:

$H_0 = IV$

$H_i = f(X_i, H_{i-1}) \qquad i = 1, 2, \ldots, t$

$h(X) = H_t$

The result of the hash function is denoted with $h(X)$ and IV is the abbreviation for Initial Value. The function f mapping an n-bit and a b-bit string to an n-bit string is called the *round* function, and the H_i's are called the *chaining variables*. Two iterations of an iterated hash function are shown in Fig. 4.

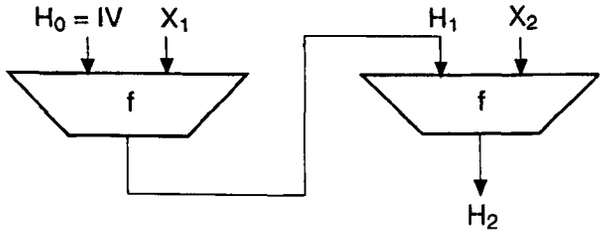

Fig. 4 - Two iterations of an iterated hash function.

Two elements in this definition have an important influence on the security of a hash function: the choice of the padding rule and the choice of the IV. It is recommended that the padding rule is unambiguous (i.e., there exist no two messages that can be padded to the same message), and that it appends at the end the length of the message. The IV should be considered as part of the description of the hash function. In some cases one can deviate from this rule, but this will make the hash function less secure and may lead to trivial collisions or second preimages.

Research on hash functions has been focussed on the question: which conditions should be imposed on f to guarantee that h satisfies certain conditions? Two main results have been shown on the properties of the round function f of an MDC. The first result is by X. Lai and J. Massey [71] and gives necessary and sufficient conditions for f in order to obtain an "ideally secure" hash function h.

Theorem 1 Assume that the padding contains the length of the input string, and that the message X (without padding) contains at least 2 blocks. Then finding a second preimage for h with a fixed IV requires 2^n operations if and only if finding a second preimage for f with arbitrarily chosen H_{i-1} requires 2^n operations.

This theorem implies that if f can be inverted with less than 2^n operations, it will require less than 2^n operations to find a second preimage for h.

A second result was proved by I. Damgård [27] in a complexity theoretic setting. It establishes a sufficient condition for h to be a CRHF. We give here the system based version that was shown independently by R. Merkle [83].

Theorem 2 Assume that the padding is unambiguous and contains the length of the input string. If f is a collision resistant function, then h is a CRHF.

Both theorems assume that the round function is symmetric in both arguments. This has the advantage that the size of the hashcode can be chosen arbitrarily. Several proposals for practical hash functions do not follow this approach, mainly to achieve a higher performance.

4.3.2. Attacks on hash functions

We will restrict the discussion to attacks that depend only on the size of the external parameters (size of the hashcode and possibly size of the key); they are thus independent of the nature of the algorithm. In order to assess the feasibility of these attacks, it is important to know that for the time being 2^{56} operations is considered to be on the edge of feasibility. Here one operation corresponds to a single evaluation of the round function. Since the speed of computers is multiplied by four every three years, 2^{64} operations is sufficient for the next 5 to 10 years, but it will be only marginally secure within 20 years. For applications with a time frame of 20 years or more, one should try to design the scheme such that an attack requires at least 2^{80} operations.

Random attack: The opponent selects a random mes-

sage and hopes that the hashcode will remain unchanged. In case of a good hash function, his probability of success equals $1/2^n$ with n the number of bits of the hashcode. The feasibility of this attack depends on the action taken in case of detection of an erroneous result, on the expected value of a successful attack, and on the number of attacks that can be carried out. For most applications this implies that $n = 32$ bits is not sufficient.

Birthday attack: This attack can only be used to produce collisions. The idea behind the birthday attack [131] is that for a group of 23 people the probability that at least two people have a common birthday exceeds 1/2. Intuitively one would expect that the group should be significantly larger. This can be exploited to attack a hash function in the following way: an adversary generates r_1 variations on a bogus message and r_2 variations on a genuine message. The probability of finding a bogus message and a genuine message that hash to the same result can be approximated by

$$1 - \exp\left(-\frac{r_1 \cdot r_2}{2^n}\right) \qquad \text{for } r_1,\ r_2 = O\left(2^{n/2}\right)$$

which is about 63% when $r_1 = r_2 = 2^{n/2}$. Note that in case of a MAC the opponent is unable to generate the MAC of a message. He could however obtain these MAC's with a chosen plaintext attack. A second possibility is that he collects a large number of messages and corresponding MAC's and divides them into two categories, which corresponds to a known plaintext attack. The involved comparison problem does not require r^2 operations: after sorting the data, which requires $O(r \log r)$ operations, comparison is easy. R. Jueneman has shown in 1986 [63] that for $n = 64$ the processing and storage requirements were feasible in reasonable time with the computer power available in every large organization. A time-memory-processor trade-off is possible.

If the function can be called as a black box, one can use the collision search algorithm proposed by J. J. Quisquater [105], which requires about $\sqrt{\pi/2} \cdot 2^{n/2}$ operations and negligible storage. To avoid this attack with a reasonable safety margin, n should be at least 128 bits. This explains the second condition in Definition 2 of a CRHF. P. C. van Oorschot and M. J. Wiener have recently developed a parallel version of this algorithm; implemented on a $10 million custom machine, finding a collision for a 128 bit hashcode would require about 24 days.

In case of digital signatures, a sender can attack his own signature or the receiver (or a third party) could offer the signer a message he is willing to sign and replace it later with the bogus message. Only the last attack can be thwarted through randomizing the message just prior to signing. If the sender attacks his own signature, the occurrence of two messages that hash to the same value might make the signer suspect, but it will be very difficult to prove the fraud to a third party.

Exhaustive key search: This attack is only relevant in case of a MAC. It is a known plaintext attack, where an attacker knows M plaintext/MAC pairs for a given key and will try to determine the key by trying all possible keys until a match is found. The expected number of trials equals 2^{k-1}, with k the size of the key in bits. To determine the key uniquely, M has to be slightly larger than k/n.

5. AN OVERVIEW OF MDC PROPOSALS

In this section we attempt to summarize the large number of proposals for practical MDC's and we discuss their status. The hash functions have been divided into four classes: hash functions based on a block cipher, hash functions based on modular arithmetic, hash functions based on a knapsack, and dedicated hash functions. For a more detailed discussion, the reader is referred to [102].

5.1. *Hash functions based on a block cipher*

Two arguments can be indicated for designers of hash functions to base their schemes on existing encryption algorithms. The first argument is the minimization of the design and implementation effort: hash functions and block ciphers that are both efficient and secure are hard to design. Many examples to support this view can be found in the literature. Moreover, existing software and hardware implementations can be reused, which will decrease the cost. A major advantage however is that the trust in existing encryption algorithms can be transferred to a hash function. It is impossible to express such an advantage in economical terms, but it certainly has an impact on the selection of a hash function. It is important to note that for the time being significantly more research has been spent on the design of secure encryption algorithms compared to the effort to design hash functions. Also, it is not obvious at all that the limited number of design principles for encryption algorithms is valid for hash functions too. The main disadvantage of this approach is that dedicated hash functions are likely to be more efficient. One also has to take into account that in some countries export restrictions apply to encryption algorithms but not to hash functions. Finally note that block ciphers may exhibit some weaknesses that are only important if they are used in a hashing mode. Well-known examples are the weak keys and the complementation property of the DES [49, 93]. Other attacks like differential [9] and linear [77] attacks are only academic if the DES is used for encryption, but might pose serious problems if extensions could be developed which are applicable to hash functions.

The encryption operation E will be written as $Y =$

$E(K, X)$. Here X denotes the plaintext, Y the ciphertext, and K the key. The size of the plaintext and ciphertext or the block length will be denoted with r, while the key size will be denoted with k. For the DES, $r = 64$ and $k = 56$ [39]. The rate of a hash function based on a block cipher is defined as the number of encryptions to process r plaintext bits.

A distinction will be made between the case $n = r$ and $n = 2r$. This is motivated by the fact that most block ciphers have a block length of only 64 bits, and therefore an MDC with a result twice the block length is necessary to obtain a CRHF. Other proposals are based on a block cipher with a large key and on a block cipher with a fixed key.

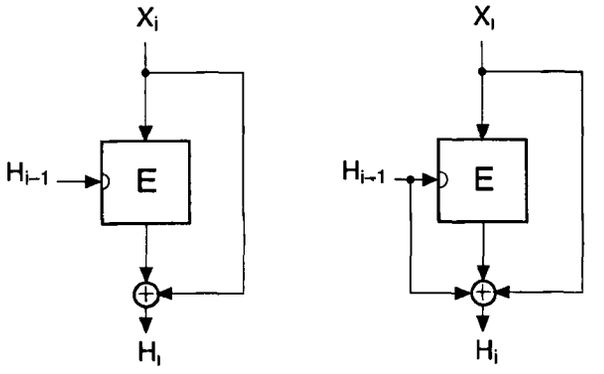

Fig. 5 - The round function of the scheme by Matyas et al. (left) and of the scheme by Preneel et al. and Miyaguchi et al. (right).

5.1.1. Size of hashcode equal to the block length

From Definition 2 it follows that in this case the hash function can only be collision resistant if the block length r is at least 128 bits. Many schemes have been proposed in this class, but the first secure schemes were only proposed after several years. Recently the author has suggested a synthetic approach: we have studied all 64 possible schemes that use exclusive ors and with an internal memory of only one block [102, 103]. As a result, it was shown that 12 secure schemes exist, but up to a linear transformation of the variables, they correspond essentially to 2 schemes. The first is the 1985 scheme by S. Matyas, C. Meyer, and J. Oseas [78]:

$$f = E^{\oplus}\left(s\left(H_{i-1}\right),\ X_i\right)$$

(here $s()$ is a mapping from the ciphertext space to the key space and $E^{\oplus}(K, X) = E(K, X) \oplus X$); the second is the variant that was proposed by B. Preneel, R. Govaerts, and J. Vandewalle in 1989 [98] and by S. Miyaguchi, M. Iwata, and K. Ohta [89] for N-hash and later for any block cipher [58]:

$$f = E^{\oplus}\left(s\left(H_{i-1}\right),\ X_i\right) \oplus H_{i-1}$$

Fig. 5 shows both schemes. The first scheme is contained in ISO/IEC 10118 Part 2, the international standard specifying hash functions based on block ciphers [57]. The 12 variants have slightly different properties related to weak keys, the complementation property, and differential attacks [102, 103]. The strength of these schemes is based on the feedforward of the plaintext, which makes the round function hard to invert. The dual of the first scheme, namely,

$$f = E^{\oplus}\left(X_i,\ H_{i-1}\right)$$

is attributed to D. Davies in [129], and to C. Meyer in [30]. D. Davies has confirmed in a personal communication to the author that he did not propose the scheme. Nevertheless, this scheme is widely known as the Davies-Meyer scheme (e.g., [88, 107]). It was also shown by the author that the security level of these hash functions is limited by $\min(k, r)$, even if the size of the internal variables is equal to $\max(k, r)$.

5.1.2. Size of hashcode equal to twice the block length

This class of hash functions has been proposed to construct a collision resistant hash function based on a block cipher with a block length of 64 bits like the DES. A series of proposals attempted to double the size of the hashcode by iterating a OWHF; all succumbed to a "divide and conquer" attack. Another proposal that was broken by the author [101] is the scheme by Zheng, Matsumoto, and Imai [133]. The Algorithmic Research Digital Fingerprint Function (ARDFP) [59] was broken by I. Damgård and L. Knudsen [28]; a weaker attack was discovered independently by the author [102]. A new version of the ARDFP with three parallel operations instead of two was presented at Crypto'93 [10]; the scheme is apparently more secure, but it is less elegant since it uses additional arithmetic operations.

R. Merkle has designed an interesting class of schemes [83]. He has given a security "proof" under the assumption that the DES has sufficient random behavior. However, the rate of the most efficient proposal equals about 3.6.

Two more efficient schemes called MDC-2 and MDC-4 were proposed by B. Brachtl et al. [12]; these schemes are also known as the Meyer-Schilling hash functions, after the two coauthors who published them at Securicom'88 [86]. For the time being a security proof is lacking. MDC-2 can be described as follows (Fig. 6):

$$T1_i = E^{\oplus}\left(H1_{i-1}, X_i\right) = LT1_i \,\|\, RT1_i$$

$$T2_i = E^{\oplus}\left(H2_{i-1},\ X_i\right) = LT2_i \,\|\, RT2_i$$

$$H1_i = LT1_i \,\|\, RT2_i$$

$$H2_i = LT2_i \,\|\, RT1_i$$

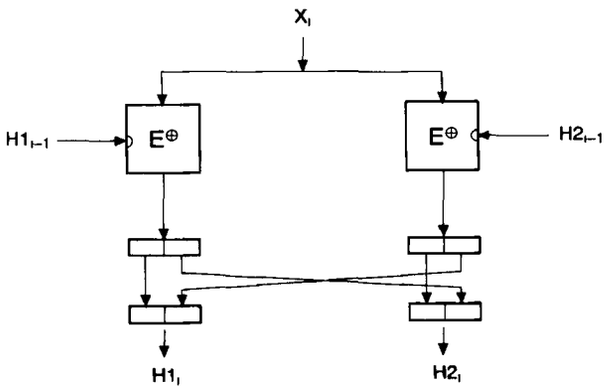

Fig. 6 - The round function of the MDC-2 hash function.

Here $H1_0$ and $H2_0$ are initialized with IV_1 and IV_2 respectively, and the hashcode is equal to $H1_t \parallel H2_t$. In order to protect these schemes against attacks based on (semi-)weak keys [93] the second and third key bits are fixed to 10 and 01 for the first and second encryption. MDC-2 is the second hash function that is specified in ISO/IEC 10118 Part 2 [57]. The best known attacks to find a collision and a second preimage require 2^{55} and 2^{83} encryptions respectively. One iteration of MDC-4 consists of the concatenation of two MDC-2 steps, where the plaintexts in the second step are equal to $H2_{i-1}$ and $H1_{i-1}$. The rate of MDC-4 is equal to 4. Finding a preimage for MDC-4 requires 2^{109} encryptions, but finding a collision is not harder than in the case of MDC-2. It should be noted that the security level of these hash functions might not be sufficient within five to ten years.

Subsequently many attempts were made to improve the efficiency of these proposals. The analysis of all these schemes is rather involved. Moreover, it can be expected that very efficient schemes are generally more vulnerable. Therefore, it is currently not recommended to use any of these hash functions. Examples of schemes in this class that failed are the scheme suggested in [97], for which the first weakness was identified in [71], and that was finally broken by the author in [102]; the scheme in [107] that was broken in [22]; the scheme in [106], for which serious weaknesses were found in [69]; the scheme in [14] for which serious weaknesses were found in [50, 71, 69]; the scheme in [50], which was broken in [68]. In [69, 70] attacks on a wider class of schemes are developed. It is shown that for this class finding a preimage requires at most 4×2^r encryptions, but for some schemes the best known attack also requires the storage of $2^r 2r$ bit values.

In addition, these schemes are vulnerable to weaknesses based on the underlying block cipher. D. Coppersmith has shown that fixed points corresponding to the weak keys of the DES are fatal for the schemes in [14, 106]. Both schemes are also vulnerable to an attack based on the complementation property [102]. For the scheme based on LOKI [14], it was already known that it could be broken based on weaknesses of LOKI [7, 9, 34]. These results suggest that countermeasures should be taken to avoid the weaknesses in the block ciphers (e.g., by fixing certain bits), rather than to design hash functions that are immune to these weaknesses.

5.1.3. Size of key equal to twice the block length

Some block ciphers have been proposed for which the key size is approximately twice the block length. Examples in this class are FEAL-*NX* [90] (a FEAL version with a 128 bit key) and IDEA [72]. Triple-DES with 2 keys has a key size of 112 bits and a block length of 64 bits and could therefore also be considered to belong to this class.

Size of hashcode equal to the block length. A scheme in this class was proposed by R. Merkle in [81]. It can also be classified as "non-invertible chaining":

$$f = E\left(H_{i-1} \parallel X_i, IV\right)$$

An alternative scheme was suggested in [71]:

$$f = E\left(H_{i-1} \parallel X_i, H_{i-1}\right)$$

These constructions can only yield a CRHF if the block length is larger than 128 bits (R. Merkle suggested 100 bits in 1979), and if the key size sufficiently large. For smaller block lengths, a OWHF can be obtained. The security depends strongly on the key scheduling of the cipher.

Size of hashcode equal to twice the block length. In order to obtain a CRHF based on a 64 bit block cipher, a different construction is required. The first two schemes in this class were recently proposed by X. Lai and J. Massey [71]. Both try to extend the Davies-Meyer scheme. One scheme is called "Tandem Davies-Meyer", and has the following description:

$$T_i = E\left(H2_{i-1} \parallel X_i, H1_{i-1}\right)$$

$$H1_i = T_i \oplus H1_{i-1}$$

$$H2_i = E\left(X_i \parallel T_i, H2_{i-1}\right) \oplus H2_{i-1}$$

The second scheme is called "Abreast Davies-Meyer":

$$H1_i = E\left(H2_{i-1} \parallel X_i, H1_{i-1}\right) \oplus H1_{i-1}$$

$$H2_i = E\left(H2_{i-1} \parallel X_i, \overline{H2_{i-1}}\right) \oplus H2_{i-1}$$

Both schemes have rate equal to 2, and are claimed to

be ideally secure, or finding a second preimage takes 2^{2r} operations and finding a collision takes 2^r operations.

5.1.4. Schemes with a fixed key

All previous schemes (except for the ARDFP schemes of section 5.1.2.) modify the key of the block cipher for every iteration. The key scheduling process is generally slower than the encryption. Moreover many attacks exploit the fact that the key can be manipulated (e.g., attacks based on weak keys). Finally, this allows to construct a hash function based on any one-way function with small dimensions.

In [100] the author proposes such a scheme with the advantage that a trade-off is possible between security level and speed. The more efficient schemes with a security level of more than 60 bits have a rate equal slightly higher than 4 and need an internal memory of about $3 \cdot 64$ bits. The size of the final hashcode can be reduced by applying a stronger but slower scheme to the final result. The design principles in this paper could be exploited to increase the security level of other hash functions like MDC-2.

5.2. *Hash functions based on modular arithmetic*

These hash functions are designed to use the modular arithmetic hardware that is required to produce digital signatures. Their security is partially based on the hardness of certain number theoretic problems. Moreover, these schemes are easily scalable. The disadvantage is that the algebraic structure makes them vulnerable to several attacks, e.g., fixed points of modular exponentiation (trivial examples are 0 and 1), multiplicative attacks, and attacks with small numbers, for which no modular reduction occurs. These attacks can be thwarted by introducing redundancy.

Several schemes with a small modulus (about 32 bits) designed by R. Jueneman (e.g., [62, 63, 64]) have been broken by D. Coppersmith. A second class of schemes uses a large modulus (the size of the modulus n is typically 512 bits or more). Here the operands are mostly elements of the ring corresponding to an RSA modulus. This poses the following practical problem: the person who has generated the modulus knows its factorization, and therefore he has a potential advantage over the other users of the hash function. The solution is to ask a trusted third party to generate the modulus or to compute the modulus with a secure multi-party computation (in that case one cannot use the modulus of the user). Alternatively, one can design the hash function in such a way that the advantage is limited.

The most efficient schemes are based on modular squaring. Moreover some theoretical results suggest that inverting a modular squaring without knowledge of the factorization of the modulus is a difficult problem. Again one can study all possible schemes that use a single squaring and exclusive ors, and that require an internal memory of only one block. Several schemes of this type have been evaluated [46, 95]. The same approach as in section 5.1.1. can be applied [102, 103]. It shows that the optimal scheme is the one proposed in [95]:

$$f = \left(X_i \oplus H_{i-1} \right)^2 \bmod N \oplus X_i$$

However, most existing proposals use the well known Cipher Block Chaining mode ([40], section 6). In order to avoid the vulnerabilities (one can go backwards easily), additional redundancy is added to the message. The first proposal was to fix the 64 most significant bits to 0 [30]. It was however shown in [46, 65] that this is not secure. In a new proposal, which appeared in several standards (e.g., the informative annex D of CCITT-X.509 [18]) the redundancy was dispersed. D. Coppersmith showed however that one can construct two messages such that the corresponding hashcodes are a multiple of each other [21]. If the hash function is combined with a multiplicative signature scheme like RSA [110], one can exploit this attack to forge signatures. Consequently, new methods for adding redundancy were proposed within ISO/IEC JTC1/SC27 and in [66], but they are still under study. It is expected that one of these methods will be included in Part 4 of ISO/IEC 10118.

B. den Boer [34] has found collisions for the round function of the squaring scheme by I. Damgård [27]. It was shown in [101] that the scheme by Y. Zheng, T. Matsumoto, and H. Imai [133] is vulnerable to the attack described in [46].

Stronger schemes have been proposed that require more operations. Examples are the use of two squaring operations [46]:

$$f = \left(H_{i-1} \oplus \left(X_i \right)^2 \right)^2 \bmod N$$

and the replacement of the squaring by a higher exponent (3 or $2^{16} + 1$) in the previous schemes. This allows to simplify the redundancy [46].

One can conclude that it would be desirable to find a secure redundancy scheme for a hash function based on modular squaring, and to replace the CBC mode by a more secure mode. If a slower scheme is acceptable, the exponent can be increased.

This class of hash functions also includes several provably secure schemes. I. Damgård [26] has suggested constructions for which finding a collision is provably equivalent to factoring an RSA modulus or finding a discrete logarithm modulo a large prime. The construction of J. K. Gibson [44] yields a collision resistant function based on the discrete logarithm modulo a composite. Both the factoring and the discrete logarithm problem are believed to be difficult number theoretic problems. The disadvantage of these schemes is that they are not very efficient.

5.3. *Hash functions based on a knapsack*

The knapsack problem was used in 1978 by R. Merkle and M. Hellman to construct the first public-key encryption system [80]. However, almost all public-key schemes based on the knapsack problem have been broken [13, 37], which has given the knapsack a bad reputation. It is an open problem whether the knapsack problem is only hard in the worst case, while the average instance is easy. If this would be true, the knapsack problem would be useless for cryptography. The problem is so attractive because both hardware and software implementations are very fast compared to schemes based on number theoretic problems.

In the case of additive knapsacks, several constructions have been suggested and broken (e.g., P. Camion and J. Patarin have shown in [16, 96] that a second preimage can be constructed for the scheme by I. Damgård [27]). Other results can be found in [47, 52]. It is for the time being an open problem whether a random knapsack with 1024 512 bit integers is hard to solve. New results of A. Joux and L. Granboulan presented at Eurocrypt'94 suggest a negative answer: lattice reduction algorithms can thus be used to break the additive knapsacks that have been proposed for hash functions. Also, one has the problem of trapdoors: the person who chooses the knapsack can easily generate it such that he knows collisions.

The first multiplicative knapsack proposed by J. Bosset [11] was broken by P. Camion [15]. A new scheme by G. Zémor is also based on the hardness of finding "short" factorizations in certain groups [132]. For the suggested parameters it can be shown that two messages with the same hashcode will differ in at least 215 bits. It remains an open problem whether it is easy to find factorizations of a "reasonable size".

5.4. *Dedicated hash functions*

In this section some dedicated hash functions will be discussed, i.e., algorithms that were especially designed for hashing operations.

MD2 [67] is a hash function that was published by R. Rivest in 1990. The algorithm is software oriented yet not very fast in software. Reduced versions of MD2 (i.e., with fewer rounds) were shown to be vulnerable [102].

A faster algorithm by the same designer is MD4 [111, 112]. Attacks on reduced versions of MD4 have been developed by R. Merkle, and by B. den Boer and A. Bosselaers [33]. This resulted in a strengthened version of MD4, namely MD5 [113]. It was however shown by B. den Boer and A. Bosselaers [35] that the round function of MD5 is not collision resistant. This does not yield a direct attack, but it raises some doubts about the security: one of the design goals, namely a collision resistant round function is not satisfied (see also Theorem 2). A second improved variant of MD4, the Secure Hash Algorithm (SHA), was developed by NSA and published by NIST [42]. The size of the hashcode is increased from 128 to 160 bits and the message words are not simply permuted but encoded with a cyclic code, which provides additional security. In spite of this, NIST has recently announced that a security flaw was discovered; this flaw can be avoided by a small modification to the standard. Another improved version of MD4 called RIPEMD was developed in the framework of the EEC-RACE project RIPE (Race Integrity Primitives Evaluation) [109]. Both SHA and RIPEMD are currently under consideration for standardization within ISO/IEC JTC1/SC27 (Part 3 of ISO/IEC 10118). HAVAL was proposed by Y. Zheng, J. Pieprzyk, and J. Seberry at Auscrypt'92 [134]; it is a collection of extensions of MD5.

N-hash is a hash function with $N = 8$ rounds designed by S. Miyaguchi, M. Iwata, and K. Ohta based on the same principles as FEAL [89, 91]. B. den Boer has found collisions for the round function [34], and E. Biham and A. Shamir have shown that a differential attack applies if N is equal to 3, 6, 9, or 12 [6, 9]. An extended version of N-hash appeared in [92] and in a Japanese contribution to ISO [58]: the roles of X_i and H_{i-1} can be interchanged, and the number of rounds is lower bounded by 4 (for a OWHF) and by 8 (for a CRHF). It was shown in [8, 102] that interchanging the values reduces the security: finding a collision is trivial if N is a multiple of 3.

FFT-Hash I and II are MDC's suggested by C.P. Schnorr [116, 117] based on the Fast Fourier Transform principle. The first version was broken independently by J. Daemen, A. Bosselaers, R. Govaerts, and J. Vandewalle [24] and by T. Baritaud, H. Gilbert, and M. Girault [4]. The second version was broken three weeks after its publication by S. Vaudenay [126]. Recently C.P. Schnorr and S. Vaudenay have proposed two new hash functions based on the same principles [118].

R. Merkle suggested in 1989 a software oriented one-way hash function called Snefru [84]. It is based on large random substitution tables (2 kbyte per pass). E. Biham and A. Shamir have shown in [6, 9] that Snefru with a small number of passes is vulnerable to differential attacks. Consequently it is recommended to use 8 passes or more, possibly combined with an increased size of the hashcode. However, these measures increase the size of the substitution tables and decrease the performance.

The scheme by I. Damgård [27] based on a cellular automaton was broken by J. Daemen, J. Vandewalle, and R. Govaerts in [23]. In the same paper these authors have proposed Cellhash, a new hash function based on a cellular automaton [23]. Later an improved version called Subhash was developed [25]. Both schemes are hardware oriented.

6. AN OVERVIEW OF MAC PROPOSALS

The general model for an iterated MAC is similar as

the model for an MDC. The main difference is that the round function f and in some cases the initial value IV depend on the secret key K.

In contrast with the variety of MDC proposals, very few algorithms exist. This can perhaps be explained by the fact that the existing standards are still widely accepted. The ANSI standard [3] specifies that the resulting MAC contains 32 bits. Clearly a result of 32 bits can be sufficient if additional protection is present against random attacks (section 4.3.2.) and if a birthday attack (section 4.3.2.) is not applicable, which is certainly the case in the wholesale banking environment. In other applications, this cannot be guaranteed. Therefore, certain authors recommend also for a MAC a result of 128 bits [63, 64].

The most widespread methods to compute a MAC are the Cipher Block Chaining (CBC) and Cipher Feed-Back (CFB) mode of the DES [3, 41, 53, 55, 85]:

$$\text{CBC}: f = E\left(K,\ H_{i-1} \oplus X_i\right) \quad \text{and}$$

$$\text{CFB}: f = E\left(K,\ H_{i-1}\right) \oplus X_i$$

The descriptions and standards differ because some of them select one of the two modes, suggest other padding schemes, or leave open the number of output bits that is used for the MAC. In the case of CFB it is important to encrypt the final result once more, to avoid a linear dependence of the MAC on the last plaintext block.

For the DES, an attack based on exhaustive key search (section 4.3.2.), differential attacks [9], and linear attacks [77] can be thwarted by encrypting only the last block with triple-DES; at the same time this can block the following chosen plaintext attack [34] (it will be described for the case of CBC): let H and H' be the CBC-MAC corresponding to key K and plaintext X and X' respectively. The attacker appends a block Y to X and obtains with a chosen plaintext attack the new MAC, which will be denoted with G. It is then clear that the MAC of the concatenation of X' and $Y' = Y \oplus H \oplus H'$ will also be equal to G. An alternative way to block this attack is to encrypt the result with a key derived from K. Both extensions are explained in an Annex to ISO/IEC 9797.

If both authenticity and secrecy are protected using the same block cipher, the keys for both operations have to be different [61, 85, 102].

A new mode to compute a MAC was suggested by the author [102, 103]:

$$f = E\left(K,\ X_i \oplus H_{i-1}\right) \oplus X_i$$

It has the advantage that the round function is harder to invert.

The Message Authentication Algorithm (MAA) is a dedicated MAC. It was published in 1983 by D. Davies and D. Clayden in response to a request of the UK Bankers Automated Clearing Services (BACS)[29, 31]. In 1987 it became a part of the ISO 8731 banking standard [53]. The algorithm is software oriented and has a 32-bit result, which makes it unsuitable for some applications.

A new non-iterative MAC based on stream ciphers was proposed recently by X. Lai, R. Rueppel, and J. Woollven [73]. Further study is necessary to assess its security.

Several authors have proposed to transform an MDC into a MAC by inserting a secret key (e.g., [125]). An analysis of these schemes shows that inserting the key at the beginning or at the end is generally not sufficient. It is recommended to insert the key in both places. However, the security can certainly be improved if the round function is made key dependent too.

The DSA algorithm (Decimal Shift and Add, not to be confused with the Digital Signature Algorithm published by NIST [43]) was designed in 1980 by Sievi of the German Zentralstelle für das Chiffrierwesen. It is used as a message authenticator for banking applications in Germany [32]. Weaknesses of this algorithm have been identified in [48, 102]. The scheme by F. Cohen [19] and its improvement by Y. Huang and F. Cohen [51] proved susceptible to an adaptive chosen message attack [99]. Attacks were also developed [102] on the weaker versions of this algorithm that are implemented in the ASP integrity toolkit [20]. Several MAC algorithms exist that have not been published, such as the S.W.I.F.T. authenticator and Dataseal [74].

Finally it should be noted that if one is willing to exchange a very long key, one should consider the unconditionally secure schemes discussed in section 4.1.

7. PERFORMANCE OF HASH FUNCTIONS

In order to compare the performance of software implementations of hash functions, an overview has been compiled in Table 1. All timings were performed on a 16 MHz IBM PS/2 Model 80 with a 80386 processor. The implementations were written by A. Bosselaers. Most of them use additional memory to improve the speed. The C-code was compiled with a 32 bit compiler in protected mode. The table has been completed with the speed of the DES, a modular squaring, and a modular exponentiation. For the last two operations a 512 bit modulus was chosen, and no use was made of the Chinese remainder theorem to speed up the computations. From these figures it can be derived that MDC-2 will run at about 100 kbit/s. Some algorithms like Snefru and SHA would perform relatively better on a RISC processor, where the complete internal state can be stored in the registers. On this type of processor, SHA is only about 15% slower than MD5.

Table 1 - *Performance of several hash functions on an IBM PS/2 (16 MHz 80386).*

Type	Hash Function	C Language (kbit/s)	Assembly Language (kbit/s)
MAC	MAA		2750
MDC	MD2	78	78
	MD4	2669	6273
	MD5	1849	4401
	SHA	710	1370
	RIPEMD	1334	3104
	N-hash	266	477
	FFT-hash I	212	304
	Snefru-8	270	270
Block Cipher	DES (+ key schedule)	130	200
	DES (fixed key)	512	660
Modular Arithmetic	Squaring	50	273
	Exponentiation ($2^{16}+1$)	1.8	14

8. CONCLUSIONS

The importance of hash functions for protecting the authenticity of information has been shown. In addition, hash functions are becoming an important basic tool to solve other security problems. The design of cryptographic hash functions that are both secure and efficient seems to be a difficult problem. For the time being only a limited number of provably secure constructions exists, that are rather slow or require a large number of key bits. Some theoretical results are available to support practical constructions, but most of our knowledge on practical schemes is originating from trial and error procedures. Therefore, it is important that new proposals are evaluated thoroughly by several independent researchers and that they are not implemented too quickly. Moreover implementations should be modular such that upgrading of the algorithm is feasible. The choice between different algorithms will also depend on the required performance. In the past standardization has played an important role, and since new standards are appearing, it is expected that the influence of standards will increase.

Acknowledgments

The author gratefully acknowledges the information received from Donald Davies, Bert den Boer, Lars Knudsen, Ben Smeets, and Doug Stinson.

Manuscript received on February 15, 1994

REFERENCES

[1] A. V. Aho, J. E. Hopcroft, J. D. Ullman: *The design and analysis of computer algorithms*. Addison-Wesley, 1974.

[2] *American national standard for Data Encryption Algorithm (DEA)*. X3.92-1981, ANSI, New York.

[3] *American national standard for financial institution message authentication (Wholesale)*. X9.9-1986 (Revised), ANSI, New York.

[4] T. Baritaud, H. Gilbert, M. Girault: *FFT hashing is not collision-free*. Advances in: Cryptology, Proc. Eurocrypt '92, LNCS 658, (R. A. Rueppel, Ed.), Springer-Verlag, 1993, p. 35-44.

[5] J. Bierbrauer, T. Johansson, G. Kabatianskii, B. Smeets: *On families of hash functions via geometric codes and concatenation*. Advances in: Cryptology, Proc. Crypto'93, LNCS 773, (D. Stinson, Ed.), Springer-Verlag, 1994, p. 331-342.

[6] E. Biham, A. Shamir: *Differential cryptanalysis of Feal and N-hash. Advances in Cryptology*. Proc. Eurocrypt'91, LNCS 547, (D. W. Davies, Ed.), Springer-Verlag, 1991, p. 1-16.

[7] E. Biham, A. Shamir: *Differential cryptanalysis of Snefru, Khafre, REDOC-II, LOKI, and Lucifer*. Advances in Cryptology, Proc. Crypto'91, LNCS 576, (J. Feigenbaum, Ed.), Springer-Verlag, 1992, p. 156-171.

[8] E. Biham: *On the applicability of differential cryptanalysis to hash functions. E.I.S.S. Workshop on cryptographic hash function. Oberwolfach (D)*. March 25-27, 1992.

[9] E. Biham, A. Shamir: *Differential Cryptanalysis of the data encryption Standard*. Springer-Verlag, 1993.

[10] E. Biham, B. Chor, A. Fiat: *A suggestion for an improved digital fingerprint function*. Presented at the Rump Session of Crypto'93.

[11] J. Bosset: *Contre les risques d'altération, un système de certification des informations*. "01 Informatique", No. 107, February 1977.

[12] B. O. Brachtl, D. Coppersmith, M. M. Hyden, S. M. Matyas, C. H. Meyer, J. Oseas, S. Pilpel, M. Schilling: *Data authentication using modification detection codes based on a public one way encryption function*. U.S. Patent Number 4,908,861, March 13, 1990.

[13] E. F. Brickell, A. M. Odlyzko: *Cryptanalysis: a survey of recent results*. In: *Contemporary Cryptology: The science of information integrity*. (G. J. Simmons, Ed.), IEEE Press, 1991, p. 501-540.

[14] L. Brown, J. Pieprzyk, J. Seberry: *LOKI - a cryptographic primitive for authentication and secrecy applications*. Advances in Cryptology, Proc. Auscrypt '90, LNCS 453, (J. Seberry, J. Pieprzyk, Eds.), Springer-Verlag, 1990, p. 229-236.

[15] P. Camion: *Can a fast signature scheme without secret be secure?* Proc. 2-nd International Conference on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes, LNCS 228, A. Poli, Ed., Springer-Verlag, 1986, p. 215-241.

[16] P. Camion, J. Patarin: *The knapsack hash function proposed at Crypto'89 can be broken*. Advances in Cryptology, Proc. Eurocrypt '91, LNCS 547, (D. W. Davies, Ed.), Springer-Verlag, 1991, p. 39-53.

[17] J. L. Carter, M .N. Wegman: *Universal classes of hash functions*. "Journal of Computer and System Sciences", Vol. 18, 1979, p. 143-154.

[18] *The Directory - Authentication framework*. C.C.I.T.T. Recommendation X.509, 1988, (same as IS 9594-8, 1989).

[19] F. Cohen: *A cryptographic checksum for integrity protection*. "Computers & Security", Vol. 6, 1987, p. 505-510.

[20] F. Cohen: *The ASP integrity toolkit*. Version 3.5. ASP Press, Pittsburgh (PA), 1991.

[21] D. Coppersmith: *Analysis of ISO/CCITT Document X.509* Annex D. IBM T. J. Watson Center, Yorktown Heights, N.Y., 10598, Internal Memo, June 11, 1989, (also ISO/IEC JTC1/SC20/WG2/N160).

[22] D. Coppersmith: *Two broken hash functions*. IBM T. J. Watson Center, Yorktown Heights, N.Y., 10598, Research Report RC 18397, October 6, 1992.

[23] J. Daemen, R. Govaerts, J. Vandewalle: *A framework for the design of one-way hash functions including cryptanalysis of Damgård's one-way function based on a cellular automaton*. Advances in Cryptology, Proc. Asiacrypt'91, LNCS 739, (H. Imai, R. L. Rivest, T. Matsumoto, Eds.), Springer-Verlag, 1993, p. 82-96.

[24] J. Daemen, A. Bosselaers, R. Govaerts, J. Vandewalle: *Collisions for Schnorr's FFT-hash*. Advances in Cryptology, Proc. Asiacrypt'91, LNCS 739, (H. Imai, R. L. Rivest, T. Matsumoto, Eds.), Springer-Verlag, 1993, p. 477-480.

[25] J. Daemen, R. Govaerts, J. Vandewalle: *A hardware design model for cryptographic algorithms*. Computer Security - ESORICS 92, Proc. Second European Symposium on Research in Computer Security, LNCS 648, (Y. Deswarte, G. Eizenberg, J. J. Quisquater, Eds.), Springer-Verlag, 1992, p. 419-434.

[26] I. B. Damgård: *Collision free hash functions and public key signature schemes*. Advances in Cryptology, Proc. Eurocrypt'87, LNCS 304, (D. Chaum, W. L. Price, Eds.), Springer-Verlag, 1988, p. 203-216.

[27] I. B. Damgård: *A design principle for hash functions*. Advances in Cryptology, Proc. Crypto'89, LNCS 435, (G. Brassard, Ed.), Springer-Verlag, 1990, p. 416-427.

[28] I. B. Damgård, L. R. Knudsen: *The breaking of the AR hash function*. Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765, (T. Helleseth, Ed.), Springer-Verlag, 1994, p. 286-292.

[29] D. Davies: *A message authenticator algorithm suitable for a mainframe computer*. Advances in Cryptology, Proc. Crypto'84, LNCS 196, (G. B. Blakley, D. Chaum, Eds.), Springer-Verlag, 1985, p 393-400.

[30] D. Davies, W. L. Price: *Digital signatures, an update*. Proc. 5th International Conference on Computer Communication, October 1984, p. 845-849.

[31] D. Davies, D. O. Clayden: *The message authenticator algorithm (MAA) and its implementation*. NPL Report DITC 109/88, February 1988.

[32] D. Davies, W. L. Price: *Security for computer networks: an introduction to data security in teleprocessing and electronic funds transfer (2nd edition)*. Wiley & Sons, 1989.

[33] B. den Boer, A. Bosselaers: *An attack on the last two rounds of MD4*. Advances in Cryptology, Proc. Crypto'91, LNCS 576, (J. Feigenbaum, Ed.), Springer-Verlag, 1992, p 194-203.

[34] B. den Boer: *Personal communication*.

[35] B. den Boer, A. Bosselaers: *Collisions for the compression function of MD5*. Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765, (T. Helleseth, Ed.), Springer-Verlag, 1994, p. 293-304.

[36] B. den Boer: *A simple and key-economical unconditional authentication scheme*. "Journal of Computer Security", Vol. 2, No. 1, 1993, p. 65-71.

[37] Y. Desmedt: *What happened with knapsack cryptographic schemes?* In: Performance Limits in Communication, Theory and Practice. (J. K. Skwirzynski, Ed.), Kluwer, 1988, p. 113-134.

[38] W. Diffie M. E. Hellman: *New directions in cryptography*. "IEEE Trans. on Information Theory", Vol. IT 22, No. 6, 1976, p. 644-654.

[39] *Data Encryption Standard*. Federal Information Processing Standard (FIPS), Publication 46, National Bureau of Standards, U.S. Department of Commerce, Washington D.C., January 1977.

[40] *DES Modes of Operation*. Federal Information Processing Standard (FIPS), Publication 81, National Bureau of Standards, US Department of Commerce, Washington D.C., December 1980.

[41] *Computer Data Authentication*. Federal Information Processing Standard (FIPS), Publication 131, National Bureau of Standards, US Department of Commerce, Washington D.C., May 1985.

[42] *Secure Hash Standard*. Federal Information Processing Standard (FIPS), Publication 180, National Institute of Standards and Technology, US Department of Commerce, Washington D.C., May 1993.

[43] *Digital Signature Standard*. Federal Information Processing Standard (FIPS), Draft, National Institute of Standards and Technology, US Department of Commerce, Washington D.C., May 1994.

[44] J. K. Gibson: *Discrete logarithm hash function that is collision free and one way*. "IEE Proceedings-E", Vol. 138, No. 6, November 1991, p. 407-410.

[45] E. Gilbert, F. MacWilliams, N. Sloane: *Codes which detect deception*. "Bell System Technical Journal", Vol. 53, No. 3, 1974, p. 405-424.

[46] M. Girault: *Hash-functions using modulo-n operations*. Advances in Cryptology, Proc. Eurocrypt'87, LNCS 304, (D. Chaum, W. L. Price, Eds.), Springer-Verlag, 1988, p. 217-226.

[47] Ph. Godlewski, P. Camion: *Manipulations and errors, detection and localization*. Advances in Cryptology, Proc. Eurocrypt'88, LNCS 330, (C. G. Günther, Ed.), Springer-Verlag, 1988, p. 97-106.

[48] F. Heider, D. Kraus, M. Welschenbach: *Some preliminary remarks on the Decimal Shift and Add algorithm (DSA)*. Abstracts Eurocrypt'86, May 20-22, 1986, Linkoping, Sweden, p. 1. 2. (Full paper available from the authors.)

[49] M. Hellman, R. Merkle, R. Schroeppel, L. Washington, W. Diffie, S. Pohlig, P. Schweitzer: *Results of an initial attempt to cryptanalyze the NBS Data Encryption Standard*. Information Systems Lab., Dept. of Electrical Eng., Stanford Univ., 1976.

[50] W. Hohl, X. Lai, Th. Meier, C. Waldvogel. *Security of iterated hash functions based on block ciphers*. Advances in Cryptology, Proc. Crypto'93, LNCS 773, (D. Stinson, Ed.), Springer-Verlag, 1994, p. 379-390.

[51] Y. J. Huang, F. Cohen: *Some weak points of one fast cryptographic checksum algorithm and its improvement*. "Computers & Security", Vol. 7, 1988, p. 503-505.

[52] R. Impagliazzo, M. Naor: *Efficient cryptographic schemes provably as secure as subset sum*. Proc. 30th IEEE Symposium on Foundations of Computer Science, 1989, p. 236-241.

[53] *Banking - Approved algorithms for message authentication, Part 1, DEA*. IS 8731-1, ISO, 1987. *Part 2, Message Authentication Algorithm (MAA)*. IS 8731-2, ISO, 1987.

[54] *Information technology - Security techniques - Digital signature scheme giving message recovery* IS 9796, ISO/IEC, 1991.

[55] *Information technology - Data cryptographic techniques - Data integrity mechanisms using a cryptographic check function employing a block cipher algorithm*. IS 9797, ISO/IEC, 1993

[56] *Information technology - Security techniques - Modes of operation of an n-bit block cipher algorithm*. IS 10116, ISO/IEC, 1991.

[57] *Information technology - Security techniques - Hash-functions, Part 1: General and Part 2: Hash-functions using an n-bit block cipher algorithm*. IS 10118, ISO/IEC, 1994.

[58] *Hash functions using a pseudo random algorithm.* ISO/IEC JTC1/SC27/WG2N98, Japanese contribution, 1991.

[59] *AR Fingerprint Function*, ISO/IEC JTC1/SC27/WG2N179, working document, 1992.

[60] T. Johansson, G. Kabatianskii, B. Smeets: *On the relation between A-codes and codes correcting independent errors.* Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765, (T. Helleseth, Ed.), Springer-Verlag, 1994, p. 1-11.

[61] R. R. Jueneman, S. M. Matyas, C. H. Meyer: *Message authentication with Manipulation Detection Codes.* Proc. 1983 IEEE Symposium on Security and Privacy, IEEE Computer Society Press, 1983, p. 33-54.

[62] R. R. Jueneman, S. M. Matyas, C. H. Meyer: *Message authentication.* "IEEE Communications Mag.", Vol. 23, No. 9, 1985, p. 29-40.

[63] R. R. Jueneman: *A high speed manipulation detection code.* Advances in Cryptology, Proc. Crypto'86, LNCS 263, (A. M. Odlyzko, Ed.), Springer-Verlag, 1987, p. 327-347.

[64] R. R. Jueneman: *Electronic document authentication.* "IEEE Network Mag.", Vol. 1, No. 2, 1987, p. 17-23.

[65] A. Jung: *Implementing the RSA cryptosystem.* "Computers & Security", Vol. 6, 1987, p. 342-350.

[66] A. Jung: *The strength of the ISO/CCITT hash function.* Arbeitspapiere der GMD, No. 492, December 1990.

[67] B. S. Kaliski: *The MD2 Message-Digest algorithm.* Request for Comments (RFC) 1319, Internet Activities Board, Internet Privacy Task Force, April 1992.

[68] L. R. Knudsen, X. Lai: *Attacks on double block length hash functions.* Proceedings of the Cambridge Workshop on Algorithms, LNCS, Springer-Verlag, to appear.

[69] L. R. Knudsen, X. Lai: *New attacks on a class of hash functions including the parallel DM.* Preprint.

[70] L. R. Knudsen, X. Lai: *Target attacks on all double block length hash functions of hash rate 1.* Preprint.

[71] X. Lai, J. L. Massey: *Hash functions based on block ciphers. Advances in Cryptology.* Proc. Eurocrypt'92, LNCS 658, (R. A. Rueppel, Ed.), Springer-Verlag, 1993, p. 55-70.

[72] X. Lai: *On the design and security of block ciphers.* "ETH Series in Information Processing", Vol. 1, (J. Massey, Ed.), Hartung-Gorre Verlag, Konstanz, 1992.

[73] X. Lai, R. A. Rueppel, J. Woollven: *A fast cryptographic checksum algorithm based on stream ciphers.* Advances in Cryptology, Proc. Asiacrypt'92, LNCS 718, (J. Seberry, Y. Zheng, Eds.), Springer-Verlag, 1993, p. 339-348.

[74] C. Linden H. Block: *Sealing electronic money in Sweden.* "Computers & Security", Vol. 1, No. 3, 1982, p. 226.

[75] M. Luby, C. Rackoff: *How to construct pseudorandom permutations from pseudorandom functions.* "SIAM Journal on Computing", Vol 17, No. 2, April 1988, p. 373-386.

[76] J. L. Massey: *An introduction to contemporary cryptology.* In: *Contemporary Cryptology: The science of information integrity.* (G. J. Simmons, Ed.), IEEE, p. 3-39.

[77] M. Matsui: *Linear cryptanalysis method for DES cipher.* Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765, (T. Helleseth, Ed.), Springer-Verlag, 1994, p. 386-397.

[78] S. M. Matyas, C. H. Meyer, J. Oseas: *Generating strong one-way functions with cryptographic algorithm.* "IBM Techn. Disclosure Bull.", Vol. 27, No. 10A, 1985, p. 5658-5659.

[79] K. Mehlhorn, U. Vishkin: *Randomized and deterministic simulations of PRAMs by parallel machines with restricted granularity of parallel memories.* "Acta Informatica", Vol. 21, 1984, p 339-374.

[80] R. Merkle, M. Hellman: *Hiding information and signatures in trapdoor knapsacks.* "IEEE Trans. on Information Theory", Vol. IT-24 No. 5, 1978, p. 525-530.

[81] R. Merkle: *Secrecy, authentication, and public key systems.* UMI Research Press, 1979.

[82] R. Merkle: *A certified digital signature.* Advances in Cryptology, Proc. Crypto'89, LNCS 435, G. Brassard, (Ed., Springer-Verlag), 1990, p. 218-238.

[83] R. Merkle: *One way hash functions and DES.* Advances in Cryptology, Proc. Crypto'89, LNCS 435, (G. Brassard, Ed.), Springer-Verlag, 1990, p. 428-446.

[84] R. Merkle: *A fast software one-way hash function.* "Journal of Cryptology", Vol. 3 No. 1, 1990, p. 43-58.

[85] C. H. Meyer, S. M. Matyas: *Cryptography: a new dimension in data security.* Wiley & Sons, 1982.

[86] C. H. Meyer, M. Schilling: *Secure program load with manipulation detection code.* Proc. Securicom 1988, p. 111-130.

[87] C. Mitchell: *Multi-destination secure electronic mail.* "The Computer Journal", Vol. 32, No. 1, 1989, p. 13-15.

[88] C. Mitchell, F. Piper, P. Wild: *Digital signatures.* In: *Contemporary Cryptology: The science of information integrity.* (G. J. Simmons, Ed.), IEEE Press, 1991, p. 325-378.

[89] S. Miyaguchi, M. Iwata, K. Ohta: *New 128 bit hash function.* Proc. 4th International Joint Workshop on Computer Communications, Tokyo, Japan, July 13-15, 1989, p. 279-288.

[90] S. Miyaguchi: *The FEAL cipher family.* Advances in Cryptology, Proc. Crypto'90, LNCS 537, (S. Vanstone, Ed.), Springer-Verlag, 1991, p. 627-638.

[91] S. Miyaguchi, K. Ohta, M. Iwata: *128 bit hash function (N-hash).* Proc. Securicom 1990, p. 127-137.

[92] S. Miyaguchi, K. Ohta, M. Iwata: *128 bit hash function (N-hash).* "NTT Review", Vol. 2, No. 6, 1990, p. 128-132.

[93] J. H. Moore, G. J. Simmons: *Cycle structure of the DES for keys having palindromic (or antipalindromic) sequences of round keys.* "IEEE Trans. on Software Engineering", Vol. 13, 1987, p. 262-273.

[94] M. Naor, M. Yung: *Universal one-way hash functions and their cryptographic applications.* Proc. 21st ACM Symposium on the Theory of Computing, 1990, p. 387-394.

[95] J. C. Pailles, M. Girault: *The security processor CRIPT.* 4th IFIP SEC, Monte-Carlo, December 1986, p. 127-139.

[96] J. Patarin: *How to find and avoid collisions for the knapsack hash function.* Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765, (T. Helleseth, Ed.), Springer-Verlag, 1994, p. 305-317.

[97] B. Preneel, A. Bosselaers, R. Govaerts, J. Vandewalle: *Collision free hash functions based on blockcipher algorithms.* Proc. 1989 International Carnahan Conference on Security Technology, p. 203-210.

[98] B. Preneel, R. Govaerts, J.Vandewalle: *Cryptographically secure hash functions: an overview.* ESAT Internal Report, K. U. Leuven, 1989.

[99] B. Preneel, A. Bosselaers, R. Govaerts, J. Vandewalle: *Cryptanalysis of a fast cryptographic checksum algorithm.* "Computers & Security", Vol. 9, 1990, p. 257-262.

[100] B. Preneel, R. Govaerts, J. Vandewalle: *On the power of memory in the design of collision resistant hash functions.* Advances in Cryptology, Proc. Auscrypt'92, LNCS 718, (J. Seberry, Y. Zheng, Eds.), Springer-Verlag, 1993, p. 105-121.

[101] B. Preneel, R. Govaerts, J. Vandewalle: *An attack on two hash functions by Zheng, Matsumoto, and Imai.* Advances in Cryptology, Proc. Auscrypt'92, LNCS 718, (J. Seberry, Y. Zheng, Eds.), Springer-Verlag, 1993, p. 535-538.

[102] B. Preneel: *Analysis and design of cryptographic hash functions.* Doctoral Dissertation, Katholieke Universiteit Leuven, 1993. See also: *Cryptographic Hash Functions.* Kluwer Academic Publishers, 1994.

[103] B. Preneel, R. Govaerts, J. Vandewalle: *Hash functions based on block ciphers: a synthetic approach.* Advances in Cryptology, Proc. Crypto'93, LNCS 773, (D. Stinson, Ed.), Springer-Verlag, 1994, p. 368-378.

[104] B. Preneel, R. Govaerts, J. Vandewalle: *Differential cryptanalysis of hash functions based on block ciphers.* Proc. 1st ACM Conference on Computer and Communications Security, ACM, 1993, p. 183-188.

[105] J. J. Quisquater, J. P. Delescaille: *How easy is collision search?. Application to DES.* Advances in Cryptology, Proc. Eurocrypt'89, LNCS 434, (J. J. Quisquater, J. Vandewalle, Eds.), Springer-Verlag, 1990, p 429-434.

[106] J. J. Quisquater, M. Girault: *2n-bit hash-functions using n-bit symmetric block cipher algorithms*. Abstracts Eurocrypt'89, April 10-13, 1989, Houthalen, Belgium.

[107] J. J. Quisquater, M. Girault: *2n-bit hash-functions using n-bit symmetric block cipher algorithms*. Advances in Cryptology, Proc. Eurocrypt '89, LNCS 434, (J. J. Quisquater, J. Vandewalle, Eds.), Springer-Verlag, 1990, p. 102-109.

[108] M. O. Rabin: *Digitalized signatures. in foundations of secure computation*. (R. Lipton, R. DeMillo, Eds.), Academic Press, New York, 1978, p. 155-166.

[109] *Race Integrity Primitives Evaluation (RIPE): final report*. RACE 1040, 1993.

[110] R. L. Rivest, A. Shamir, L. Adleman: *A method for obtaining digital signatures and public-key cryptosystems*. "Communications ACM", Vol. 21, February 1978, p. 120-126.

[111] R. L. Rivest: *The MD4 message digest algorithm*. Advances in Cryptology, Proc. Crypto'90, LNCS 537, (S. Vanstone, Ed.), Springer-Verlag, 1991, p. 303-311.

[112] R. L. Rivest: *The MD4 message-digest algorithm*. Request for Comments (RFC) 1320, Internet Activities Board, Internet Privacy Task Force, April 1992.

[113] R. L. Rivest: *The MD5 message-digest algorithm*. Request for Comments (RFC) 1321, Internet Activities Board, Internet Privacy Task Force, April 1992.

[114] J. Rompel: *One-way functions are necessary and sufficient for secure signatures*. Proc. 22nd ACM Symposium on the Theory of Computing, 1990, p. 387-394.

[115] R. A. Rueppel: *Stream ciphers*. In: *Contemporary Cryptology: The science of information integrity*. (G. J. Simmons, Ed.), IEEE Press, 1991, p. 65-134.

[116] C. P. Schnorr: *An efficient cryptographic hash function*. Presented at the Rump Session of Crypto'91.

[117] C. P. Schnorr: *FFT-Hash II, efficient cryptographic hashing*. Advances in Cryptology, Proc. Eurocrypt'92, LNCS 658, (R. A. Rueppel, Ed.), Springer-Verlag, 1993, p. 45-54.

[118] C. P Schnorr, S. Vaudenay: *Parallel FFT-Hashing*. Proceedings of the Cambridge Workshop on Algorithms, LNCS, Springer-Verlag, to appear.

[119] C. E. Shannon: *Communication theory of secrecy systems*. "Bell System Technical Journal", Vol. 28, 1949, p. 656-715.

[120] G. J. Simmons: *A natural taxonomy for digital information authentication schemes*. Advances in Cryptology, Proc. Crypto'87, LNCS 293, (C. Pomerance, Ed.), Springer-Verlag, 1988, p. 269-288

[121] G. J. Simmons: *A survey of information authentication. In Contemporary Cryptology: The science of information integrity*. (G. J. Simmons, Ed.), IEEE Press, 1991, p. 381-419.

[122] D. R. Stinson: *The combinatorics of authentication and secrecy codes*. "Journal of Cryptology", Vol. 2, No. 1, 1990, p. 23-49.

[123] D. R. Stinson: *Combinatorial characterizations of authentication codes*. "Designs, Codes and Cryptography", Vol. 2, 1992, p. 175-187. See also: *Advances in cryptology*, Proc. Crypto'91, LNCS 576, (J. Feigenbaum, Ed.), Springer-Verlag, 1992, p. 62-73.

[124] D. R. Stinson: *Universal hashing and authentication codes*. "IEEE Trans. on Information Theory", to appear. See also: *Advances in cryptology*, Proc. Crypto'91, LNCS 576, (J. Feigenbaum, Ed.), Springer-Verlag, 1992, p. 74-85.

[125] G. Tsudik: *Message authentication with one-way hash functions*. "Computer Communications Review", Vol. 22, No. 5, 1992, p. 29-38.

[126] S. Vaudenay: *FFT-hash-II is not yet collision-free*. Advances in Cryptology, Proc. Crypto'92, LNCS 740, (E.F. Brichell, Ed.), Springer-Verlag, 1993, p. 587-593.

[127] K. Vedder: *Security aspects of mobile communications*. State of the Art and Evolution of Computer Security and Industrial Cryptography, LNCS 741, (B. Preneel, R. Govaerts, J. Vandewalle, Eds.), Springer-Verlag, 1993, p. 189-206.

[128] M. N. Wegman, J. L. Carter: *New hash functions and their use in authentication and set equality*. "Journal of Computer and System Sciences", Vol. 22, 1981, p. 265-279.

[129] R. S. Winternitz: *Producing a one-way hash function from DES*. Advances in Cryptology, Proc. Crypto'83, (D. Chaum, Ed.), Plenum Press, New York, 1984, p. 203-207.

[130] A. C. Yao: *Theory and applications of trapdoor functions*. Proc. 23rd IEEE Symposium on Foundations of Computer Science, 1982, p. 80-91.

[131] G. Yuval: *How to swindle Rabin. Cryptologia*. Vol. 3, 1979, p. 187-189.

[132] G. Zémor: *Hash functions and graphs with large girths*. Advances in Cryptology, Proc. Eurocrypt'91, LNCS 547, (D. W. Davies, Ed.), Springer-Verlag, 1991, p. 508-511.

[133] Y. Zheng, T. Matsumoto, H. Imai: *Duality between two cryptographic primitives*. Proc. 8th International Conference on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, LNCS 508, (S. Sakata, Ed.), Springer-Verlag, 1991, p. 379-390.

[134] Y. Zheng, J. Pieprzyk, J. Seberry: *HAVAL - a one-way hashing algorithm with variable length output*. Advances in Cryptology, Proc. Auscrypt '92, LNCS 718, (J. Seberry, Y. Zheng, Eds.), Springer-Verlag, 1993, p. 83-104.