IST-2002-507932

ECRYPT

European Network of Excellence in Cryptology

Network of Excellence

Information Society Technologies

D.STVL.3
Ongoing Research Areas in Symmetric Cryptography

Due date of deliverable: 31. January 2005
Actual submission date: 31. January 2005

Start date of project: 1 February 2004 Duration: 4 years

Lead contractor: Institut National de Recherche en Informatique et en Automatique (INRIA)

Revision 2.5

Project co-funded by the European Commission within the 6th Framework Programme		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission services)	
RE	Restricted to a group specified by the consortium (including the Commission services)	
CO	Confidential, only for members of the consortium (including the Commission services)	

Ongoing Research Areas in Symmetric Cryptography

Editor
Anne Canteaut (INRIA)

Contributors
Daniel Augot (INRIA), Alex Biryukov (KUL), An Braeken (KUL),
Carlos Cid (RHUL), Hans Dobbertin (RUB), Håkan Englund (LUND),
Henri Gilbert (FTRD), Louis Granboulan (ENS), Helena Handschuh (G+),
Martin Hell (LUND), Thomas Johansson (LUND), Alexander Maximov (LUND),
Matthew Parker (UiB), Thomas Pornin (CRY), Bart Preneel (KUL),
Matt Robshaw (RHUL), Michael Ward (MC)

31. January 2005
Revision 2.5

The work described in this report has in part been supported by the Commission of the European Communities through the IST program under contract IST-2002-507932. The information in this document is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

Contents

Introduction

Basic cryptographic algorithms split into two families: symmetric algorithms, otherwise known as secret-key algorithms, which normally require a key to be shared and simultaneously kept secret within a restricted group, and public-key algorithms where the private key is almost never shared. From outside, this may give the impression that symmetric techniques become obsolete after the invention of public-key cryptography in the mid 1970's. However, symmetric techniques are still widely used because they are the only ones that can achieve some major functionalities as high-speed or low-cost encryption, fast authentication, and efficient hashing. Today, we find symmetric algorithms in GSM mobile phones, in credit cards, in WLAN connections, and symmetric cryptology is a very active research area.

There is a strong need for further research in this area. On the one hand, new industrial needs are arising with the development of new application environments. For instance, the demand for low-cost primitives dedicated to low-power devices is pressing. On the other hand, progress in cryptanalysis may threaten the security of some existing and widely used algorithms. A better understanding of recent attacks is then necessary for the evaluation of existing primitives and for designing new and more secure ones.

This document gives a brief summary of some of the research trends in symmetric cryptography. Four major aspects have been identified and will be the focus of future work within the *Symmetric Techniques Virtual Lab* in ECRYPT:

- A need for lightweight algorithms (especially for low-cost stream ciphers), dedicated to hardware environments where the available resources are heavily restricted, arises from industry. A dedicated ECRYPT workshop will be held on that topic in July of 2005;
- The new attacks presented at Crypto 2004 on different commonly used hash functions must be further investigated. Even if the SHA-family is still considered secure, the very recent results raise some concerns about its security. The investigation and the development of new general design principles for hash functions (and for MAC algorithms) is a major challenge;
- The recent development of algebraic attacks which may threaten both stream and block ciphers is another important breakthrough. A better understanding of these techniques requires further works on several topics, such as the development and the study of algorithms for solving algebraic systems of multivariate equations and the definition of new design criteria related to these attacks.
- The development of new cryptanalytic techniques, such as algebraic attacks, has important consequences on the properties required for the elementary functions used in a symmetric cipher. Therefore, there is a need for a clarification of all design criteria which must be prescribed for a given application. The development of tools in order to help the designers on this particular topic is being encouraged with ECRYPT.

The present document is organized as follows. In Section 1, we review the status of work with regards to different types of cryptographic algorithms. In Section 2 we consider a new and active area of research concentrating on algebraic cryptanalysis. Then we turn our attention

to construction and provable properties of symmetric cryptography before finally closing with some remarks on the major industrial needs in the area of symmetric cryptography.

1 The Status of Symmetric Primitives

Here we review recent progress and open problems concerning different types of symmetric primitives (block ciphers, stream ciphers, hash functions and message authentication codes). One recent advance has been in the cryptanalysis of hash functions and in Sections 1.3 and 1.4 we investigate these new cryptanalytic results and consider their impact on the design of secure hash functions and MAC algorithms. Finally, in Section 2 we focus on algorithms for solving algebraic systems, which lie at the core of the recently proposed algebraic attacks against block and stream ciphers.

1.1 Block ciphers

Block ciphers and stream ciphers are the two main classes of primitives encountered in symmetric cryptology. A block cipher can be described as a keyed pseudo-random permutation of the $\{0,1\}^n$ set of n-bit blocks, whereas a stream cipher can be described as a keyed pseudo-random sequence over a finite alphabet (e.g. $\{0,1\}$). The most usual block lengths for existing block ciphers are $n = 64$ and 128 bits.

Block ciphers are typically slower than stream ciphers (20-40 cycles/byte) and require more gates (5000-100,000), though this gap in performance is narrowing. They form a very flexible building block, that can be used in various modes of operation for confidentiality, message or entity authentication, one-way functions, and hash functions. Block ciphers can even be efficiently converted to a stream cipher, if used in an appropriate mode of operation (such as OFB), whereas the converse is not true. Historically, block ciphers have been more prominent than stream ciphers in open standards (DES, Triple-DES, AES), which may explain their popularity. They are used in many cryptographic applications such as home banking, e-mail, authentication, key distribution and in recent standards for encryption in mobile telephony, in hard disk encryption, and so forth. Stream ciphers are preferred for selected applications with high performance or low power requirements.

In the mid-1970's, the block cipher standard DES (Data Encryption Standard) was published by the US NBS (National Bureau of Standards, now NIST, National Institute for Standards and Technology) [19]. DES has been the de facto world standard for encryption until the mid-1990's though in recent years the short key length of DES (56 bits) had undermined its security. In critical applications DES was often replaced by Triple-DES (threefold iteration of DES). In addition, certain applications required a block length larger than 64 bits (both DES and Triple-DES operate on 64-bit blocks). Following an open competition, the Belgian proposal *Rijndael* by Rijmen and Daemen, was selected as the AES (Advanced Encryption Standard) [18] to succeed DES. More than half of existing security products currently use DES or variants of DES but many products will shift to AES and a large part of the confidentiality of mass market applications of the cryptology will, in the future, be based on the security of AES. Outside from DES, Triple-DES and AES, several other recently proposed block ciphers are also used in numerous security products, for instance IDEA (an algorithm previously used in the PGP file encryption software), *RC5* (an algorithm used in many S/MIME protected email products), *MISTY1* and its variant *KASUMI* (which was adopted encryption and message authentication algorithm for the UMTS third generation mobile system), and numerous block cipher proposals have been evaluated as part of the

European project NESSIE.

Studies made during the 25 years of existence of DES have led to important theoretical advances in the public knowledge on the design of block ciphers. The discovery of *differential and linear cryptanalysis* techniques [28, 8] in the early-1990's represent (together with pre-computation techniques such as Hellman's Time-memory trade-off [21]) the most significant advances in the analysis of DES and more generally of iterated block ciphers. Consequently resistance to these attacks has become one of the main criteria in the analysis of the strength of block ciphers. Some recently proposed designs, e.g. MISTY [29] and KASUMI (whose nested structure exploits upper bounds of differential and linear transition probabilities established by Nyberg and Knudsen [33], or constructions based upon the so-called decorrelation theory by Vaudenay [40], offer provable resistance against basic forms of differential and linear cryptanalysis.

Several cryptanalytic methods other than differential and linear cryptanalysis have been discovered: higher order differential attacks, truncated differential attacks, interpolation attacks, integral (saturation) attacks, impossible differential, boomerang, and rectangle attacks can be more effective than usual differential techniques. Other attacks such as chi-square, partitioning, and stochastic cryptanalysis, as well as attacks against key schedules, such as sliding attacks and related key attacks can offer other avenues for the cryptanalyst. Although formal proofs of security against these various classes attacks have not been systematically developed for existing block ciphers, their existence is generally taken into account by the designers of block cipher proposals, and an algorithm such as AES can be reasonably conjectured to resist these attacks techniques (most of which are essentially statistical in nature). While the only assertion one has for now is that there exists no feasible shortcut attack on AES, it has been observed that the AES uses several algebraic structures, it cannot be entirely precluded that further use of advanced algebraic techniques such as the use of Gröbner basis computations, probabilistic interpolation, and quadratic approximations might not establish weaknesses in AES [17, 32].

Outside from the study of various categories of attacks and of design methods to resist these attacks, cryptologic research on block ciphers has been strongly influenced by the development of unconditional security proof techniques which allows us to partially validate one specific level of a block cipher construction or perhaps a mode of operation of a block cipher. This security paradigm was proposed by Luby and Rackoff in 1988 [27] and later developed by Patarin, Maurer, Rogaway, Bellare, Vaudenay and others. On one level, a cryptographic construction is modeled as a pseudo random function (or permutation) generator, and this is compared with an ideal (uniformly drawn) function or permutation generator with the same input and output sizes. Pseudorandomness results allow us to partially validate block cipher features such as the so-called Feistel structure of the DES construction, or to validate modes of operation of block cipher such as the CBC MAC mode. The use of such techniques will likely become more systematic in validating the structure of block ciphers or their modes of operation.

1.1.1 Ongoing research directions

Some current research areas include the following.

Cryptanalysis of AES and similar block ciphers. The AES algorithm is a simple and elegant design and it is secure against attacks known to date; there are even some strong heuristic arguments that differential and linear cryptanalysis do not apply. A first line of research could be to further validate via a security proof that AES is secure against differential and linear attacks and improved variants thereof, perhaps taking into account the difference between probabilities over all keys and security for a particular key. The security of AES could also be validated by studying in more depth the basic AES structure (SPN network), and by trying to establish its soundness by further investigating pseudo-randomness and super-pseudo-randomness of generic constructions following the AES approach.

A second line of research should be to investigate and develop new attacks that exploit the algebraic structures present within the AES. While the AES is very elegant mathematically, it is clear that this opens new lines of research for cryptanalysis, which require a longer term effort. In this respect, a cryptographic algorithm is very different from other algorithms in computer science: a "normal" algorithm that works correctly now, will also work correctly in five years, and can only be improved. The security of a cryptographic algorithm with fixed parameters such as AES can only degrade over time because the state of the art in cryptanalysis develops. It is impossible at this stage to indicate which types of attacks will be successful against the AES, but we can make a few educated guesses. A first strategy could be to extend the rather sophisticated methods (combining genetic algorithms with statistical attacks) developed to attack MD4 and similar hash functions to block ciphers. Another recently proposed completely new idea is based on the use of systems of quadratic equations which might be used to recover the key. For the time being, this approach has not been proved to be effective. However, fundamental research is required to investigate the applicability of this new mathematical technique as well as other algebraic attacks, such as probabilistic interpolation attacks.

New constructions and building blocks. New block ciphers that may offer specific advantages over the AES (such as lower gate count, higher performance, very fast key setup, very large block length, or enhancements in terms of provable security) need to be studied and designed. An important example of an "alternative" block cipher to the AES is KASUMI, which is being deployed in third generation phones, mainly for its low gate count, but it is clear that other applications will need improved block ciphers as well. In this context, it is important to explore block ciphers that have a structure completely different from DES and AES. This will also require new approaches to cryptanalysis, similar to the new approaches now being studied for AES.

Among the basic elementary building blocks used in block cipher constructions, only the S-boxes design and the overall structure (Feistel scheme, Misty scheme, etc.) have been extensively analyzed. Other building blocks such as: the linear part of S/P networks, the key schedule, and the use of uniform rather than hybrid round structures have been much less investigated until now.

1.1.2 Open problems for block ciphers

Some open problems in the area of block ciphers include the following.

- Can a practical and efficient block cipher be constructed whose security can be directly and provably related to the intractability of a well identified and well studied mathematical problem?
- Are there alternative construction strategies? Block ciphers are pseudorandom permutations and generally result from the iteration of a one-to-one round function. Pseudorandom n-bit to m-bit functions based upon the iteration of not one to one round functions might also represent useful primitives: such functions could be directly used for the purposes of authentication or key distribution, and modes of operation allowing to encrypt data using such a primitive could also be easily defined. However, such constructions have not been well studied. Most constructions proposed until now proved to be extremely weak, due to the existence of collision attacks and/or "ciphertext only" attacks, and it would be useful to know whether simple and efficient constructions avoiding such attacks can be found.
- How do we estimate an optimal (in terms of security) number of rounds for an iterative cipher?
- Are there (applicable) attacks that are independent of the number of rounds, or are polynomial in the number of rounds?
- Can we refine criteria on the properties expected from the linear (diffusion) part of block ciphers with a substitution/permutation structure? These have been much less studied than criteria governing the selection of S-boxes. For instance, it is easy to determine stable subspaces of the linear part of a S/P block cipher, but the cryptanalytic consequences of the existence of stable subspaces are not well known.
- Can we state the optimal properties for S-boxes? We still do not know if there exist differentially 2-uniform bijective S-boxes with an even number of bits. We do not know how many exist with an odd number of bits. The same questions might apply for linear approximations. Algebraic properties such as large algebraic degree, no low degree approximation, and no multivariate quadratic approximation might also need to be taken into account (see Section 3). It is still hard to determine when higher differential attacks apply. Should we try to design with all these aspects in mind?
- Are there new (and more powerful) attacks that use the data adaptively?
- Is it possible to develop block ciphers that are inherently more secure against certain side channel attacks? Perhaps this can be done by using secret sharing-type techniques and one-way functions inherently within the design. This may lead to completely new designs of block ciphers, that can be much faster than existing ones in environments where side channel attacks are applicable. Implementation dependent attacks and performance concerns can be improved by enhancing the cooperation between cryptographers and the engineers that use block ciphers.
- Should we salt or tweak block ciphers, that is, add a public input for randomization? This may result in simpler and more efficient modes, at the cost of more powerful attacks against the basic primitive. This is an interesting trade-off to consider, which may bring substantial improvements.
- It is still an open problem whether Hellman's time-memory tradeoff [21] is optimal.

1.2 Stream ciphers

While block ciphers are generally used to encrypt a block of characters of a plaintext message using a fixed encryption transformation, a stream cipher encrypts individual characters of the plaintext using an encryption transformation that varies with time. We often refer to any stream cipher producing one output bit on each clock as a *classical stream cipher design*. However other stream ciphers are word-oriented and may encrypt the plaintext as bytes or larger units of data.

Typically we consider a binary additive stream cipher in which the keystream, the plaintext, and the ciphertext are sequences of binary digits. The output sequence of the keystream generator, $z_1, z_2, \ldots$ is added bitwise to the plaintext sequence $m_1, m_2, \ldots$, producing the ciphertext $c_1, c_2, \ldots$. The keystream generator is initialized through a secret key K, and hence, each key K will correspond to an output sequence. Since the key is shared between the transmitter and the receiver, the receiver can decrypt by adding the output of the keystream generator to the ciphertext and obtain the message sequence, see Figure 1. This kind of

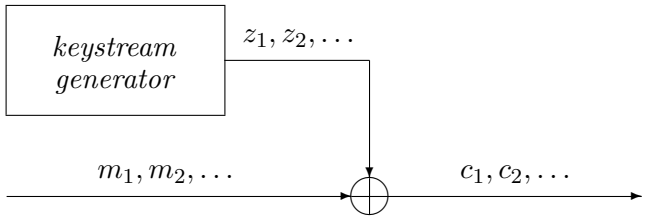

Figure 1: A binary additive stream cipher.

stream cipher is known as a synchronous stream cipher. While there is a second type of stream cipher—the self-synchronising stream cipher—discussions at the ECRYPT *State of the Art of Stream Ciphers* workshop suggested that there was little real demand for this second type of keystream generator.

The design goal for a stream cipher is to produce a secure keystream where we are typically concerned about two types of attacks:

- *Key recovery attacks*: The cryptanalyst tries to recover the secret key K.
- *Distinguishing attack*: The cryptanalyst tries to determine whether any arbitrarily selected key stream $z_1, z_2, \ldots, z_N$ has been generated by a given stream cipher or whether it is a truly random sequence. If we can build a distinguisher, i.e. a box that implements some algorithm, to correctly answer the above question with high probability, then we have a distinguishing attack.

It is clear that a distinguishing attack is weaker than a key recovery attack. Whereas a key recovery attack allows the attacker to get access to any possible plaintext information he or she wants, the distinguishing attack can give only some limited amount of information to the attacker. For example, if the plaintext message is one out of two possible, the distinguishing attack can tell the attacker which of the two was transmitted.

Today, there is an extensive theoretical knowledge on stream ciphers and on various design principles for stream ciphers. Often the basic building block of stream cipher design is the *Linear Feedback Shift Register* and as a consequence much stream cipher design work has focused on the ideas of modifying, combining, and disrupting LFSR sequences so as to derive secure keystream generators. There are however some other prominent ciphers that do not use LFSRs, the obvious example being RC4.

LFSR-based designs. Many stream ciphers are built around the *Linear Feedback Shift Register*. Within this class of ciphers there are a variety of design approaches.

A *combination generator* is a key stream generator for stream cipher applications. The idea of the combiner generator is to destroy the inherent linearity in LFSRs by using several LFSRs in parallel. The outputs from these n parallel LFSRs u_1, ..., u_n are combined by a nonlinear Boolean function, denoted by $f(\cdot)$ and called a combining function. The output from the nonlinear function is the keystream and the output symbol at time instant t is denoted by z_t, this symbol is calculated as

$$z_t = f(u_t^1, u_t^2, \ldots, u_t^n) \ ,$$

where u_t^i denotes the output bit from LFSR i at time instant t.

It is possible to consider the constituent sequences u_1, ..., u_n as being formed from successive stages of a single LFSR. In this case the combining function $f(\cdot)$ is known as a filter function and the corresponding stream cipher as a *filter generator*. In both the case of the combination and the filter function, however, it is possible to set out certain desirable properties of the function $f(\cdot)$ so as to (hopefully) derive secure keystream generation. However, as new attacks are developed, it is likely that new design criteria may need to be added.

The combination and filter generators are very popular designs, but the consistent alignment of internal registers as the output is generated might make the job somewhat easier for the cryptanalyst. One way to try and thwart such attacks is to use what is termed *clock control*. Again the stream cipher would be based around LFSRs, but instead of the subcomponents being clocked at the same time, the decision to update a particular register, or the decision as to how far to move that register at any given instance, is dependent on some other component of the cipher. Such ciphers are referred to as clock control ciphers and there are many different designs in use today.

Table driven stream ciphers. Another major class of stream cipher design is that of the table driven cipher. The classic example is RC4 which has a massive state space which is slowly—but continually—evolving. While some weaknesses in the output function of RC4 have been noted, table-driven stream ciphers can offer significant performance advantages though with some potentially large implementation cost in hardware. Their design is often such that they have little in common with LFSR-based design and so, as a result, are often immune to classical LFSR-based analysis. However they can become susceptible to dedicated attacks.

1.2.1 Typical stream cipher analysis

Just as there are a few different families of stream cipher designs, it is possible to group together the most important types of stream cipher analysis. Since LFSRs are used widely in stream cipher design, it is perhaps unsurprising that analysis exploiting the algebraic properties of the shift register is very popular. Consequently the use of linear complexity, the Berlekamp-Massey algorithm, the linear complexity profile, and other advanced but related topics in the analysis of stream ciphers is well-known. There is a large collection of results on the properties of the final sequences derived from some ensemble or combination of constituent LFSR components.

Divide and conquer attacks. A very generic set of attacks are referred to as divide-and-conquer attacks. These rely on the fact the the keystream generator is built out of several, rather weak, components. As an example, suppose that we have a nonlinear combiner generator consisting of n different LFSRs and that these LFSRs have lengths L_1, L_2, ..., and L_n. Then the total number of different possible initialization values of these LFSRs is $\prod_{i=1}^{n}(2^{L_i}-1)$. However if we assume that there is some weakness in the generation process so that the properties of some individual component register leaks into the keystream produced (the usual example is that there exists some *correlation* between the keystream and the output of one of the LFSRs) then one can potentially break the keystream generator one component at a time. Thus, under a known keystream attack and under the assumption that we have sufficiently many keystream bits, we might be able to try to identify the correct initial state of each LFSR in turn. If so, we might be able to find the initial states of all the LFSRs in at most $\sum_{i=1}^{n}(2^{L_i}-1)$ trials which is much less than $\prod_{i=1}^{n}(2^{L_i}-1)$ we might have expected. While the exact property exploited to identify the component LFSR might vary from cipher to cipher, there are a variety of design principles that might be employed to protect the cipher against a range of divide-and-conquer attacks.

Correlation attacks. One way to launch a divide-and-conquer attack is to exploit what is called the correlation between an output sequence and one of the constituent components. Certainly basic versions of LFSR-based stream ciphers are vulnerable to *correlation attacks*. These techniques were introduced by Siegenthaler [39] and in the original correlation attack, the initial state of the target LFSR was recovered by an exhaustive search: the value of the correlation enables to distinguish the correct initial state from a wrong one since the sequence generated by a wrong initial state is assumed to be statistically independent of the keystream. Thereafter, fast correlation attacks were introduced by Meier and Staffelbach in 1988 [30, 31]. They avoided the need to examine all possible initializations of the target LFSR by using efficient error-correcting techniques. But, they required the knowledge of a longer segment of the keystream. In practice, the most efficient fast correlation attacks are able to recover the initial state of a target LFSR of length 60 for an error-probability $p = 0.4$ in a few hours on a PC with around 10^6 bits of keystream.

Algebraic attacks. A recently developed—and powerful—type of analysis has been introduced in [16]. The basic idea behind the *algebraic attack* is very simple. First, the cryptanalyst sets up a system of equations including key bits and output bits. Second, the cryptanalyst

solves this system to recover key or keystream information. Solving a system of linear equations is easy using, for instance, Gaussian elimination. However a good cipher always contains a non-linear part, so the equations will be non-linear, that is of degree greater than one. If the system of equations is very over-defined then the equation set can still be solved using techniques such as linearization, or other methods such as Gröbner bases. However, since the complexity of solving such equations grows exponentially with the degree of the equations, the cryptanalysis is keen to identify low degree equations relating bits of the output and the internal components of the cipher. A variety of techniques have been proposed to help the cryptanalyst but their effectiveness tends to be somewhat cipher specific. In 2003 a significant improvement was proposed and the fast algebraic attack was introduced [15]. The idea was to reduce the degree in the equations using an additional pre-computation step. This step was later improved in [2]. It is noteworthy that there are some important limitations to algebraic attacks. However, generally speaking, they have been very effective in the analysis of several stream ciphers to date. This will be discussed more in Section 2.

1.2.2 Research directions and open problems

Recent progress in research related to algebraic attacks has given us new design criteria for stream ciphers. To add to past conditions related to the non-linearity and correlation-immunity of combining or filter functions we can add properties that aim to thwart algebraic attacks. As the state-of-the art progresses more conditions will presumably be added.

One interesting consideration for stream ciphers is their future desirability. At the ECRYPT *State of the Art of Stream Ciphers* workshop in October 2004 [34], Adi Shamir expanded on some thoughts originally presented at the 2004 RSA Security Conference. These were concerned with the future need for stream ciphers with, it seems, block ciphers being perfectly adequate for use in all but a few niche areas. These niche areas were identified as:

- Exceptional encryption performance in software, where the luxury of additional hardware is not available to speed up encryption.
- Any reasonable kind of encryption performance in hardware environments where the available resources such as gate count or power might be heavily restricted. The extreme example of this is provided by simple RFID tags.

Interestingly it is unclear whether any current stream cipher proposals particularly satisfy these two requirements. Therefore one area of open research that is being encouraged within ECRYPT is the development of stream ciphers for these two environments. New research, perhaps focusing on different underlying components such as T-functions instead of LFSRs, might help point the way to new primitives suited to demanding applications. It is intended that these will help to satisfy the demands for lightweight cryptographic primitives suitable for deployment in a broad range of applications.

In tandem with the search for lightweight stream ciphers, work within ECRYPT is emphasizing the need for lightweight algorithms in general. This will be the focus of a dedicated workshop that is to be hosted by the Graz University of Technology in July of 2005.

1.3 Hash functions

Hash functions, also known as message digests, are important cryptographic primitives. The hash of a message can be compared with the fingerprint of a person. An important application of hash functions are digital signature schemes, where instead of a signing the message itself a short hash value representing that message is signed. The selection of a secure hash function is therefore necessary to create a secure digital signature scheme. Here, security means a high level of collision resistance. We assume that the reader is familiar with the notion of a hash function and its basic properties.

During 2004 there was considerable progress in the cryptanalysis of hash functions, to be more precise, in attacking the collision resistance of dedicated hash functions. At Crypto'04 three results on this topic were presented that drew a lot of attention: Biham and Chen presented a new cryptanalytic method, the neutral bit technique (see [6]) which they applied to find near-collisions of SHA-0 and collisions for significantly reduced versions of SHA-1. Joux, Carribault, Jalby and Lemuet applied this technique to the full SHA-0 and succeeded in finding collisions, which they presented at the rump session of Crypto'04 (see [23]). In the same rump session Wang, Lai, Feng and Yu also presented collisions (see [42]) for the functions MD4, MD5, HAVAL-128 and RIPEMD, which they found using another new technique.

In this section we will describe some background and details about these two new kinds of attacks. We will begin with some general framework, describing some common aspects of the two attack methods and their main differences, before in the following subsections we will describe some details of these attacks.

Notation. We will denote the message blocks by X, X' and the single words in these blocks by X_i, i.e. we have $X = (X_0, \ldots, X_{k-1})$ where in most cases $k = 16$. The values resulting from the message expansion which are used as inputs in the step operation are denoted by W_i. By $X_i \lll s$ we denote the rotation (cyclic shift) of X_i by s bits.

As in the dedicated hash functions considered in this context usually only one register is changed in each step, we can use a notation in which it is not necessary to distinguish which of the registers actually used in an implementation is changed in a certain step. Therefore we simply denote the (new) value of the register changed in step i by R_i. For example the step operation of SHA-0 and SHA-1 then can be described as follows

$$R_i = (R_{i-1} \lll 5) + (R_{i-5} \ggg 2) + \phi_i(R_{i-2}, R_{i-3} \ggg 2, R_{i-4} \ggg 2) + K_i + W_i$$

where the (seemingly) additional rotations come from the fact that in each step additionally one register is rotated by two bits.

1.3.1 General framework

Both techniques can be divided into two main parts. In the first part the general "attack strategy", a *difference pattern*, is chosen or determined. In the second part, which requires usually a lot of time-consuming computations, the actual collisions, which conform to this difference pattern, are determined.

Difference patterns. In a collision attack we are looking for two messages X and X' which produce the same hash value. Therefore we have to correlate the computations that are done when computing the hash value of X and the computations for the hash value of X'. A *difference pattern* is a sequence of differences, where each difference corresponds to one step in these computations and is defined as a difference of a value from the computation for X and the corresponding value from the computation for X'.

We have to distinguish between *input differences*, which means differences in the message words, or rather in the values W_i after the message expansion, and *output differences*, that is, differences appearing in the register values R_i after applying the step operations. We say that a certain message *conforms* to a certain difference pattern (consisting of an input and an output pattern), if processing this message and the message modified by the given input pattern results in the given output pattern.

Another important distinction is that between *modular differences*, that is, differences with respect to integer addition usually modulo 2^n (where n is the register size in bits), and $\oplus$-*differences*. This is also the most obvious difference between the two presented attacks. Biham and Chen, based on the attack of Chabaud and Joux, talk only about $\oplus$-differences, whereas Wang et al. mainly use modular differences for their attack and talk about $\oplus$-differences only where necessary. But it is not easy to tell what is the more promising approach. Using $\oplus$-differences is easier if you use a linearized function, because then you can apply many techniques from linear algebra or coding theory for example, but the problem is that you have to transfer everything back to the original function afterwards. In contrast, modular differences can be applied to the original function more easily but you cannot avoid also looking at $\oplus$-differences in addition to handle for example the bitwise defined functions used in the step operation.

1.3.2 The neutral bit technique.

The neutral bit technique by Biham and Chen is an improvement of the method used by Chabaud and Joux to attack SHA-0 in [13]. Therefore we will first sketch the ideas of their attack.

The Chabaud/Joux Attack on SHA-0. Chabaud and Joux use an approach with $\oplus$-differences. But as it is nearly impossible to analyze the $\oplus$-difference behaviour directly in the original step operation, they use an $\oplus$-linear approximation of the step operation, which can be constructed by substituting all nonlinear parts (i.e. the modular additions and the nonlinear, bitwise defined functions) by $\oplus$-additions. Then for this linearized function it is easy to find difference patterns which lead to a collision.
Their idea to actually find collisions for the original function is to look for messages which have the same difference propagation in the original function as in the linearized function, i.e. applying the computed input difference pattern to this message results in the same output difference pattern as in the case of the linearized function. Clearly, this cannot be true for every message, but it is possible to deduce conditions from the difference patterns which describe for which actual register values the difference propagation is the same.

Chabaud and Joux used some refined randomized search to find actual collisions: They

start, by repeatedly choosing random values for X_0 and computing the first step until all the conditions for R_0 are fulfilled. Then they do the same with X_1, the second step and R_1 and so on up to X_{14}, the 15-th step and R_{14}. This can be done step by step, as the values $R_0, \ldots, R_{i-1}$ are not influenced by X_i for $i \leq 15$.
After having found this (first 15 words of a) message conforming to the first 15 steps, they only choose random values for X_{15}. This does not change the output difference pattern for the first 15 steps, but produces a nearly random behaviour for the remaining steps. Thus mainly the probability for fulfilling the conditions for these *remaining steps* is of importance for the overall complexity of this attack. Of course, one can construct at most 2^{32} different messages by choosing only X_{15} and hence, after a certain number of (unsuccessful) tries for X_{15} one has to start from the beginning again by choosing new (random) values for $X_0, \ldots, X_{14}$.

In [13] Chabaud and Joux describe a difference pattern which is fulfilled (in this sense) with a probability of 2^{-61}, that means their attack has a complexity of about 2^{61}.

Improvements by Biham and Chen. In [6] Biham and Chen improved this approach, by looking for what they call *neutral bits*. Their idea is to increase this range of steps for which you try to assure in advance (before the main part of the randomized search) that the randomly chosen messages conform to the difference pattern. Clearly, if you look at more than 15 steps, it is not possible anymore (as before) to change some message word arbitrarily without having to fear that the output difference pattern has changed in these steps. But this is, where the *neutral bits* come into play:

Suppose we start with a message conforming to the given difference pattern up to some step r. Then, a bit of the message is called *neutral*, if inverting it does not prevent the message from conforming to the difference pattern up to step r. A *pair of bits* is called neutral, if this is true for each of these bits and also if both are inverted simultaneously. Analogously, a *set of bits* is called neutral if this holds for every subset of bits and it is called *2-neutral* if each pair of bits from this set is neutral. The maximum number of neutral bits for a given message and step r is denoted by $k(r)$.

Biham and Chen observed the following: If you have a 2-neutral set of bits, then after inverting any subset of these bits the message still conforms to the difference pattern up to step r with a probability of about $1/8$. This means, starting from one initial message which conforms to the difference pattern up to step r, you can produce about $2^{k(r)-3}$ messages which also conform up to step r.

The number of producible message can even be increased by not only using neutral bits but also *simultaneous-neutral* sets of bits. A set of bits is called simultaneous-neutral, if the single bits of this set are not neutral, but inverting all the bits of the set simultaneously does not prevent the message from conforming to the differential pattern up to step r. Thus, each simultaneous-neutral set of bits can be viewed and used as a single neutral bit of a message, probably increasing the number $k(r)$.

To apply this method successfully, two things are required:

- deciding up to which step r the message has to conform to the given difference pattern
- finding messages with large 2-neutral sets of bits for a given message efficiently

For the first question you have to consider the probability $P(r)$ that a randomly chosen message conforms to the given difference pattern in the steps following step r. This probability can be approximated very well from the conditions on the register values and r should be chosen such that the number of producible messages $2^{k(r)-3}$ is about $1/P(r)$. Then there is some non-negligible chance to find a collision by testing all the possible messages.

For actually finding large sets of neutral bits, Biham and Chen give a description how to reduce this problem to finding maximal cliques in a graph. Although this is in general a NP-hard problem, in the cases which are needed here this seems to work fine. Then to actually find messages which have large 2-neutral sets they suggest to perform some kind of local search. They start with one message and compute the corresponding set of 2-neutral bits. Then they test for some of the messages that can be produced by changing some certain subsets of these bits (according to another observation they made) which of these new messages have a larger 2-neutral set of bits and then take one of these messages as the new base message. By repeatedly doing this process they can maximize (locally) the size of the 2-neutral set of bits.

In [6] Biham and Chen present collisions for an extended 82-step SHA-0 which were found using the technique described. Additionally in [7] applications of this method to reduced version of SHA-1 are presented which result in collisions for up to 43 steps and the conclusion that collisions for the *last* 53 steps should also be possible. Joux et al. (see [23]) applied this technique to find actual collision for the original (80 step) SHA-0, by combining 4 such differential patterns, constructed as described above, to produce a collision with two messages consisting of 4 message blocks each.

1.3.3 The attacks of Wang *et al.*

Most of the details given in this section have not been published up to now and stem from [41]. The attacks by Wang et al. differ from the method described above in one main fact, which is that they mainly use modular differences instead of the $\oplus$-differences. This also means, that they do not use a linearized approximation of the compression function but work directly on the original step operation.

The recently published collisions produced by these attacks (see [42]) are all collisions for hash functions which use, as message expansion, a roundwise permutation in contrast to the recursive message expansion which is applied in the SHA-functions. This means that each of the message words is applied exactly once per round as one of the W_i. (The l-th *round* of the compression function which uses message blocks of k words consists of the steps $(l-1)k, \ldots, lk-1$)

Finding the difference pattern. Similar as in the Chabaud/Joux attack Wang et al. start by looking for a difference pattern, but in their attack the search for an appropriate difference pattern is again divided into two separate parts: finding a useful input difference pattern to have a "nice" differential behaviour in some part (e.g. in the last round), and then find an appropriate output difference pattern for the remaining steps.

For example, in the MD4-attack, the input pattern is chosen such that randomly chosen messages conform to the difference pattern *in the last* (i.e. third) *round* with a probability of 1/4. This can be done by looking at the step operation and choosing the input differences

such that they cancel each other after only a few steps. For example, the step operation of the last round of MD4 can be described by the following equation (for step i):

$$R_i \;=\; (R_{i-4} + \;(R_{i-1} \oplus R_{i-2} \oplus R_{i-3}) + W_i + K_i) \lll s_i$$

Thus, if we induce a (modular) difference of 2^{16} into X_{12} which is used as W_{35} in step 35, we can see that in this step the value in the brackets produces also a difference of 2^{16} (if we suppose that in the steps before there have been zero output differences in the R_i). Then by the rotation by $s_{35} = 15$ bits, this modular difference is rotated to either a difference of 2^{31} or $2^{31} + 1$, depending on one of the carry bits. Hence, with a probability of $1/2$ (depending on the actual values of the registers) the modular difference in R_{36} is 2^{31}. The advantage of using this special modular difference is that it implies also an $\oplus$-difference of 2^{31} in R_{35}. Thus in the next step

$$R_{36} \;=\; (R_{32} + \;(R_{35} \oplus R_{34} \oplus R_{33}) + W_{36} + K_{36}) \lll 3$$

it follows that the $\oplus$-operation $R_{35} \oplus R_{34} \oplus R_{33}$ results in a difference of again 2^{31}. By choosing a difference of $2^{31} + 2^{28}$ for $X_2 = W_{36}$ we then get a difference of 2^{28} in the brackets (the "2^{31}"s cancel as we compute modulo 2^{32}) which is again rotated to a difference of 2^{31} in R_{36} with a probability of $1/2$. Similar considerations can be done for the following steps to produce zero differences. The complete difference propagation up to the collision in step 41 is illustrated in Figure 2.

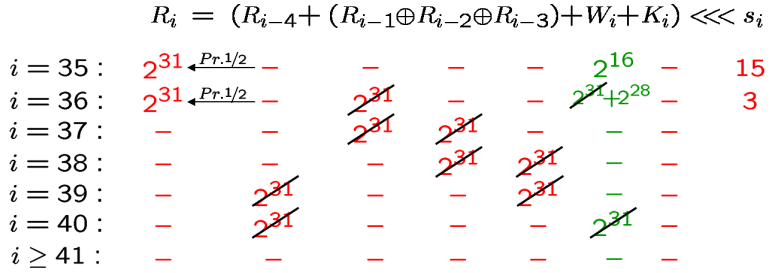

Figure 2: Difference propagation in last round of MD4.

By this consideration the complete input difference pattern is determined. To determine the complete difference pattern it remains to find an output pattern for the first two rounds which can be fulfilled given this input pattern. Wang et al. do this similarly to what we just described by simply considering the step operation and the modular differences in the registers. But the distinction now is that for this part there is no freedom in the choice of the differences for the W_i anymore.

The only freedom of choice for the attacker comes from the fact that the relation between modular differences and $\oplus$-differences is not one-to-one: A modular difference of 2^k may, for example, result in an $\oplus$-difference of $2^{k+l} + 2^{k+l-1} + \ldots + 2^k$ with arbitrary values of $l \in \{0, \ldots, 31 - k\}$, depending on the actual register values, where small values for l are more probable than large values. Thus by imposing conditions on these register values it is possible to influence the $\oplus$-differences and thus the differences coming from the bitwise defined functions in the step operation.

Using such techniques Wang et al. found the differential patterns together with a set of conditions on the register values (similar to those in the Chabaud/ Joux attack) which were used to find the actual collisions.

Basic and advanced modifications. To actually find messages conforming to this differential patterns, Wang et al. do what they call *basic and advanced modifications*. This means they start with some arbitrary message and determine up to which step t the message conforms to the differential pattern. Then depending on the step t they do either a basic or an advanced modification of this message to assure that the failing condition now is fulfilled.

For the first round (step $0 \leq t \leq 15$) such a *basic modification* simply means to adjust the bits in the register R_t such that the conditions are fulfilled and to compute the message word X_t which is necessary to produce this register value from the transformed equation of the step operation (again for the example of MD4):

$$X_t = (R_t \ggg s_t) - R_{t-4} - \phi_t(R_{t-1}, R_{t-2}, R_{t-3}) - K_t$$

For later rounds ($t \geq 16$) the necessary *advanced modification* is a little bit more sophisticated. The general idea is, as before for the basic modification, to look for a message bit which can be used to change the incorrect register bit. So, for example to correct the i-th bit in R_{16}, one could just invert the $(i-3)$-th bit of X_0, as can be seen from the description of step 16:

$$R_{16} \;=\; (R_{12} + \;\phi_{16}(R_{15}, R_{14}, R_{13}) + X_0 + K_{15}) \lll 3$$

But simply changing one bit in X_0 would cause a lot of changes in the register values following the first application of X_0, probably causing that many already fulfilled conditions would become false again. Thus the idea for an advanced modification is to invert this bit indirectly and thereby cause as few changes as possible. For example, to change the $(i-3)$-th bit as required above, one could change the i-th bit of R_0:

$$X_0 = (R_0 \ggg 3) - R_{-4} - \phi_0(R_{-1}, R_{-2}, R_{-3}) - K_0$$

To avoid further changes in other registers, one also has to adjust the message blocks X_1, X_2, X_3, X_4 as they are used in the following steps which are also influenced by the change in R_0:

$$X_t = (R_t \ggg s_t) - R_{t-4} - \phi_t(R_{t-1}, R_{t-2}, R_{t-3}) - K_t, \quad t = 1, 2, 3, 4$$

Of course, this might also cause some conditions to fail now, but the probability that this happens is much smaller, because the conditions include only register values and at least in $R_0, \ldots, R_{15}$ only one bit was changed by this advanced modification.

Another advantage of this advanced modifications is that there are many possibilities to perform them. Hence, if one way causes some other condition to fail, there are other ways one can try to correct one condition without loosing other conditions in return.

Wang et al. successfully applied this technique to break two hash functions, whose compression functions consists of three rounds, namely MD4 and HAVAL-128. From looking at the methods used it seems that functions with about three rounds can be broken by this method in general, while functions with more than three rounds can only be broken if there are special weaknesses which can be exploited.

For example they also found collisions for the RIPEMD-0 (the original RIPEMD from [14]) which consists of two parallel strings of three rounds each, i.e. of six rounds altogether. The weakness here is, that the two strings of three rounds are nearly identical in the design such that it was possible to find one differential pattern for three rounds which can be applied simultaneously to both strings.

The most interesting collisions presented by Wang et al. in [42] are the collisions for MD5 for which a little bit more effort was required, as MD5 consists of four rounds:

Wang's attack on MD5. The general idea is to use multi-block messages (similar to what Joux et al. did to produce the SHA-0 collisions in [23]), i.e. messages for which the compression function has to be invoked more than once. In the case of the MD5 attack the differential pattern for the first application of the compression function leads to a difference vector of

$$(2^{31}, 2^{31} - 2^{25}, 2^{31} - 2^{25}, 2^{31} - 2^{25}).$$

The differential pattern for the second application of the compression function starts with these differences and leads to the following differences:

$$(2^{31}, 2^{31} + 2^{25}, 2^{31} + 2^{25}, 2^{31} + 2^{25})$$

Thus in the final computation step (which adds again the initial register values to the current ones) these differences cancel such that there is a collision after these two applications of the compression function.

The special weakness (compare also [12] on this) exploited in this attack is that it is possible to induce a output difference of 2^{31} by choosing some input differences and then this output difference is propagated from step to step with probability 1 in the third round and with probability 1/2 per step in a large part of the fourth round. Hence, it is possible to find an input difference pattern which leads to an output difference pattern in round 3 and 4 which is fulfilled with high probability. Thus it is possible to attack even this four round hash function with the method described earlier.

1.3.4 Research directions

So far we have described the research perspectives closely related to the state of the art for the, w.r.t. practical applications most significant, class of MD4-type hash functions. The analysis of other hash functions as Whirlpool and Tiger remains also a very important challenge.

Of course there are also fundamental questions for which answers are completely elusive today, like how to design a fast and *provable* secure hash function. The process underlying the design and analysis of hash functions today is more of trial-and-error character. Thus investigation and development of new general principles similar to, for instance, the MD-strengthening would be of great interest.

1.4 MAC algorithms

MAC algorithms compute a short string as a complex function of a message and a secret key. In a communications setting, the sender will append the MAC value to the message.

The recipient shares a secret key with the sender. On receipt of the message, he recomputes the MAC value using the shared key and verifies that it is the same as the MAC value sent along with the message. If the MAC value is correct, he can be convinced that the message originated from the particular sender and that it has not been tampered with during the transmission. Indeed, if an opponent modifies the message, the MAC value will no longer be correct. Moreover, the opponent does not know the secret key, so he is not able to predict how the MAC value should be modified.

The main security properties of a MAC algorithm is that one should not be able to forge MAC values, that is, to predict values on new messages without knowing the secret key. A second requirement is that it should be computationally infeasible to recover the MAC key by exhaustive search, since an exhaustive key search allows for arbitrary forgeries.

1.4.1 Block cipher based MAC algorithms

The most popular MAC algorithms are the variants of CBC-MAC [1] which are based on a block cipher; in the past this has been mostly DES or triple-DES and currently AES is becoming more popular. Since the mid 1990s, constructions based on hash functions such as HMAC have been introduced on the Internet [3].

There exist several security proofs for CBC-MAC and variants (Bellare, Krawczyk and Rogaway [5], Petrank and Rackoff [36], Vaudenay, Maurer, Black and Rogaway [10]). Most of these proofs reduce the security of CBC-MAC to the assumption that the underlying block cipher is a pseudo-random function. Moreover, the best advantage an attacker has to break the system that can be shown in this case is on the order of $q^2 \cdot m^2/2^n$, with q the number of chosen texts, m the number of blocks in each message, and n the block length of the block cipher.

If CBC-MAC is used with a pseudo-random function, the best known attack by Preneel and van Oorschot [37] has advantage $q^2 \cdot m/2^n$. Recently, Rogaway has pointed out some small flaws in the old proofs and has presented a new security proof starting from the assumption that the underlying block cipher is a pseudo-random permutation. He obtains an advantage $q^2 \cdot m/2^n$. If CBC-MAC is used with a pseudo-random permutation (as this is done in practice), the best known attack by Preneel and van Oorschot [37] has advantage $q^2/2^n$.

This leads to the following open problems:

1. Try to close the gaps between the best known attack and the security bound; it seems likely that in both cases this can be achieved by tightening the proof and getting rid of a factor of m.

2. Try to unify the existing proof methodologies for CBC-MAC and variants.

3. Try to refine the model for the security proofs by distinguishing between known and chosen texts and MAC verifications as is typically done in papers presenting attacks on MAC schemes.

4. CBC-MAC has the disadvantage that it does not allow for parallelism, unlike PMAC [11]. For PMAC we might ask: Can the gap between proofs and bounds for

PMAC be closed easily? Can this construction be further simplified (see also Rogaway, Asiacrypt 2004)?

5. Can we develop better attacks and proofs for the security against key recovery attacks for constructions that double the key length such as MacDES [25] and the ANSI retail MAC?

6. Can we beat the birthday bound? There are only two MAC constructions known that beat the birthday bound: RMAC [22] (which needs a stronger security assumption on the block cipher, i.e. that the block cipher needs to be resistant to related-key attacks) and XOR-MAC [4]. Do other constructions exist that are more efficient than XOR-MAC, yet require weaker assumptions than RMAC?

1.4.2 Hash function based MAC algorithms

The security of HMAC, EHMAC and ENMAC [35] is based on a set of non-standard assumptions, such as pseudo-randomness properties in the presence of secret initialization vectors (IVs) and collision-resistance or weak-collision-resistance with secret IVs. These assumptions should be studied for reduced-round versions of popular hash algorithms such as MD5, SHA-1 and RIPEMD-160. Also, collisions and near-collisions have been found on several hash functions at Crypto 2004.

1. For how many rounds of these functions can one break the HMAC construction?

2. Do near-collisions endanger the HMAC construction at all? Are more efficient primitives such as EHMAC or ENMAC at risk?

1.4.3 Universal hash function based MAC algorithms

Universal hash functions known today are either moderately efficient (in between HMAC-SHA-1 or HMAC-MD5) with a rather short key, or extremely efficient (UMAC [9]) with a rather long key.

1. Can we improve the trade-off, that is, develop constructions that are extremely fast in software yet have modest keys (say less than 64 bytes)?

1.4.4 Authenticated encryption schemes

An authenticated encryption scheme is a symmetric-key mechanism in which both the privacy and the authenticity of a message are protected. The standard admitted solution is a two-pass scheme where one encrypts the data using a symmetric encryption algorithm and checks the message for authenticity using a MAC algorithm. Both algorithms use their own key. The generic composition paradigm is to encrypt-then-authenticate, but certain schemes may also prove secure if composed the opposite way [26]. More efficient schemes such as one-pass schemes do also exist. They provide simultaneous encryption and authentication and include IAPM [24], OCB [38], XCBC [20], but they all make use of independent random

masking data. Other variants define schemes for which headers and specific data need not be encrypted. These are called authenticated-encryption schemes with associated data. Still other schemes exist which associate authenticity with encryption based on stream ciphers.

1. Under which conditions are security proofs available for schemes which authenticate-then-encrypt or encrypt-and-authenticate?
2. Are there any one-pass AE schemes which do not require independent random masking data? Is there an alternative approach?
3. Can we develop security proofs for recently proposed AE primitives based on stream ciphers?

References

[1] ANSI X9.19. Financial institution retail message authentication. American Bankers Association, 1986.

[2] F. Armknecht. Improving fast algebraic attacks. In *Fast Software Encryption - FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 65–82. Springer-Verlag, 2004.

[3] M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In *Advances in Cryptology - CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag, 1996.

[4] M. Bellare, R. Guérin, and P. Rogaway. XOR MACs: New Methods for Message Authentication Using Finite Pseudorandom Functions. In *Advances in Cryptology - CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 15–28. Springer-Verlag, 1995.

[5] M. Bellare, J. Kilian, and P. Rogaway. The security of cipher block chaining. In *Advances in Cryptology - CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 341–358. Springer-Verlag, 1994.

[6] E. Biham and R. Chen. Near-Collisions of SHA-0. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 290–305. Springer-Verlag, 2004.

[7] E. Biham and R. Chen. Near-Collisions of SHA-0 and SHA-1. In *Selected Areas in Cryptography - SAC 2004*, 2004. `http://www.cs.technion.ac.il/~biham/`.

[8] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. In *Advances in Cryptology - CRYPTO'90*, volume 537 of *Lecture Notes in Computer Science*, pages 2–21. Springer-Verlag, 1991.

[9] J. Black, S. Halevi, H. Krawczyk, T. Krovetz, and P. Rogaway. UMAC: Fast and Secure Message Authentication. In *Advances in Cryptology - CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 216–233. Springer-Verlag, 1999.

[10] J. Black and P. Rogaway. CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions. In *Advances in Cryptology - CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 197–215. Springer-Verlag, 2000.

[11] J. Black and P. Rogaway. A Block-Cipher Mode of Operation for Parallelizable Message Authentication. In *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 384–397. Springer-Verlag, 2002.

[12] B. den Boer and A. Bosselaers. Collisions for the Compression Function of MD5. In *Advances in Cryptology - EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, page 293. Springer-Verlag, 1993.

[13] F. Chabaud and A. Joux. Differential Collisions in SHA-0. In *Advances in Cryptology - CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 56–71. Springer-Verlag, 1998.

[14] RIPE Consortium. *Ripe Integrity Primitives – Final report of RACE Integrity Primitives Evaluation (R1040)*, volume 1007 of *Lecture Notes in Computer Science*. Springer-Verlag, 1995.

[15] N. Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 176–194. Springer-Verlag, 2003.

[16] N. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 345–359. Springer-Verlag, 2003.

[17] N. T. Courtois and J. Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *Advances in Cryptology - Asiacrypt'02*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer-Verlag, 2002.

[18] FIPS 197. Advanced Encryption Standard. Federal Information Processing Standards Publication 197, 2001. U.S. Department of Commerce/N.I.S.T.

[19] FIPS 46-3. Data Encryption Standard. Federal Information Processing Standards Publication 46-3, 1999.

[20] V. D. Gligor and P. Donescu. Integrity-Aware PCBC Encryption Schemes. In *Security Protocols Workshop*, volume 1796 of *Lecture Notes in Computer Science*, pages 153–171. Springer-Verlag, 1999.

[21] M. E. Hellman. A cryptanalytic time memory trade-off. *IEEE Transactions on Information Theory*, (26):401–406, 1980.

[22] E. Jaulmes, A. Joux, and F. Valette. On the Security of Randomized CBC-MAC Beyond the Birthday Paradox Limit: A New Construction. In *Fast Software Encryption - FSE 2002*, volume 2365 of *Lecture Notes in Computer Science*, pages 237–251. Springer-Verlag, 2002.

[23] A. Joux, P. Carribault, W. Jalby, and C. Lemuet. Collisions in SHA-0. Presented at the rump session of CRYPTO 2004, August 2004.

[24] C. S. Jutla. Encryption Modes with Almost Free Message Integrity. In *Advances in Cryptology - EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 529–544. Springer-Verlag, 2001.

[25] L. Knudsen and B. Preneel. MacDES: MAC algorithm based on DES. *Electronics Letters*, 34(9):871–873, 1998.

[26] H. Krawczyk. The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?). In *Advances in Cryptology - CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 310–331. Springer-Verlag, 2001.

[27] M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom function. *SIAM Journal on Computing*, 17(2), 1988.

[28] M. Matsui. Linear cryptanalysis method for DES cipher. In *Advances in Cryptology - EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*. Springer-Verlag, 1994.

[29] M. Matsui. New Block Encryption Algorithm MISTY. In *Fast Software Encryption - FSE'97*, Lecture Notes in Computer Science, pages 54–68. Springer-Verlag, 1997.

[30] W. Meier and O. Staffelbach. Fast correlation attacks on stream ciphers. In *Advances in Cryptology - EUROCRYPT'88*, volume 330 of *Lecture Notes in Computer Science*, pages 301–314. Springer-Verlag, 1988.

[31] W. Meier and O. Staffelbach. Fast correlation attack on certain stream ciphers. *J. Cryptology*, pages 159–176, 1989.

[32] S. Murphy and M. J. B. Robshaw. Essential algebraic structure within the aes. In *Advances in Cryptology - CRYPTO'02*, Lecture Notes in Computer Science, pages 17–38. Springer-Verlag, 2002.

[33] K. Nyberg and L.R. Knudsen. Provable security against a differential attack. *Journal of Cryptology*, 8(1):27–37, 1995.

[34] ECRYPT Network of Excellence, editor. *SASC Workshop Record*, 2004. Available via `www.isg.rhul.ac.uk/research/projects/ecrypt/stvl/sasc.html`.

[35] S. Patel. An Efficient MAC for Short Messages. In *Selected Areas in Cryptography - SAC 2002*, volume 2595 of *Lecture Notes in Computer Science*, pages 353–368. Springer-Verlag, 2002.

[36] E. Petrank and C. Rackoff. CBC MAC for Real-Time Data Sources. *Journal of Cryptology*, 13(3):315–338, 2000.

[37] B. Preneel and P. C. van Oorschot. On the Security of Iterated Message Authentication Codes. *IEEE Transactions on Information Theory*, 45(1):188–199, 1999.

[38] P. Rogaway, M. Bellare, and J. Black. OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. *ACM Trans. Information System and Security*, 6(3):365–403, 2003.

[39] T. Siegenthaler. Decrypting a class of stream ciphers using ciphertext only. *IEEE Transactions on Computers*, C-34(1):81–84, 1985.

[40] S. Vaudenay. Provable security for block ciphers by decorrelation. In *Proceedings of STACS '98*, number 1371 in Lecture Notes in Computer Science, pages 249–275. Springer-Verlag, 1998.

[41] X. Wang, X. Lai. Private Communication, November 2004.

[42] X. Wang, X. Lai, D. Feng and H. Yu. Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD. Presented at the rump session of CRYPTO 2004, August 2004. (`http://eprint.iacr.org/2004/199`).

2 Algebraic attacks on symmetric primitives

The recent development of algebraic attacks can be considered an important breakthrough in the analysis of symmetric primitives, since they apply to both block and stream ciphers. The basic principle of these techniques goes back to Shannon's work: they consist in expressing the whole cryptosystem as a large system of multivariate algebraic equations (typically over $\mathbf{F}_2$), which can be solved to recover the secret key. Efficient algorithms for solving such algebraic systems are therefore the essential ingredients of algebraic attacks and have recently started receiving special attention from the cryptographic community.

In this section we discuss the basic principles of algebraic attacks on block and stream ciphers. We give a brief overview of the construction of such attacks and the main algorithms for solving algebraic systems. We conclude with recent results on the complexity of some of these algorithms and future research directions.

2.1 Algebraic attacks

Algebraic attacks represent a new approach to cryptanalysis. In contrast to conventional methods of cryptanalysis, these new techniques are primarily algebraic rather than statistical; they exploit the intrinsic algebraic structure of the cipher. More specifically, the attacker expresses the encryption transformation as a large set of multivariate polynomial equations, and subsequently attempts to solve the system to recover the encryption key. Algebraic attacks are in principle applicable to both block ciphers and stream ciphers.

Block ciphers. While in theory most modern block ciphers can be fully described by a system of multivariate polynomials over a finite field, for the majority of the cases such systems prove to be just too complex for any practical purpose. Yet there are a number of recently proposed ciphers that present a highly algebraic structure and could therefore be more vulnerable to algebraic attacks [4]. Of particular interest is the case of the AES. Courtois and Pieprzyk described in [12] how to express the AES encryption operation as a large, sparse, overdefined system of multivariate quadratic equations over $\mathbf{F}_2$. Based on an alternative representation of the cipher, a simpler system of equations over $\mathbf{F}_{256}$ was presented in [19]. These two systems exploit the fact that the AES S-Box is based on the inverse mapping over $\mathbf{F}_{256}$, and has therefore a very simple algebraic description. Although some *ad hoc* methods have been proposed for solving these systems, currently it is not known whether they can provide an efficient way to recover the secret key.

Stream ciphers. Generally speaking, algebraic attacks have been (in theory) quite effective in the analysis of several LFSR-based stream ciphers [9]. The attack exploits the fact that each new bit of the key stream gives a new equation on the key bits. By collecting a large number of bits from the key stream, one can construct a system of equations that can be solved using one of the methods discussed below.

2.2 Techniques for solving polynomial systems

Solving multivariate polynomial systems is a typical problem studied in Algebraic Geometry and Commutative Algebra. In this section, we focus on the main algorithms for solving algebraic systems, in the context of cryptology. Our discussion will go from the simplest to the most efficient algorithms, that is from the linearization principle to F_4 and F_5, through XL and Buchberger algorithms, although this does not respect the chronological order of discovery of these algorithms. We conclude by discussing some recent results on the relationship between these algorithms.

The problem. Let k be a field and $f_1, \ldots, f_m$ be polynomials in n variables with coefficients in k, i.e. $f_i \in k[X_1, \ldots, X_n]$, for $i = 1, \ldots, m$. Let K be an algebraic extension of k. The problem is to find $(x_1, \ldots, x_n) \in K^n$ such that $f_i(x_1, \ldots, x_n) = 0$, for $i = 1, \ldots, m$. Note that the problem may have no solution (inconsistency of the equations), a finite number of solutions, or an infinite number of solutions (when the system is underdefined and K is the algebraic closure of k).

This problem is most often studied in the context of abstract algebra. More precisely, let $I \subseteq k[X_1, \ldots, X_n]$ be the *ideal* generated by $f_1, \ldots, f_m$ and

$$V_K(I) = \{(x_1, \ldots, x_n) \in K^n; \quad f_i(x_1, \ldots, x_n) = 0, \text{ for } i = 1 \ldots m\}$$

be the *variety* over K associated to I. The problem is then to find $V_K(I)$.

When k is a finite field of order q, one can always add to the existing set of equations the so-called field equations $X_i^q = X_i$, for $i = 1 \ldots n$, and obtain $m + n$ equations. For most cryptographic applications, the case of interest is when $k = K = \mathbf{F}_2$. In this case, the field equations are $X_i^2 = X_i$. This preprocessing step has the following consequences: the space of solutions is 0-dimensional (or empty), including at "infinity", and the ideal becomes radical (i.e. the solutions are of multiplicity one). In the following discussion, we will consider that the systems have been prepared this way, when q is not too large.

2.2.1 Linearization

The method of *linearization* is a well-known technique for solving large systems of multivariate polynomial equations. In this method, one considers all monomials in the system as independent variables and tries to solve the system using linear algebra techniques. More precisely, let A be the set of multi-indices $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbf{N}^n$, which represent the exponents of the monomials of $k[X_1, \ldots, X_n]$. Then any polynomial f can be written as $f = \sum_{\alpha \in A} c_\alpha X^\alpha$, where the sum involves only a finite number of monomials $X^\alpha = X_1^{\alpha_1} \cdots X_n^{\alpha_n}$. Using this notation, we can write the following matrix M_{L}:

$$\begin{array}{c} \\ f_1 \\ \vdots \\ f_m \end{array} \begin{array}{c} \begin{array}{ccc} \ldots & X^\alpha & \ldots \end{array} \\ \begin{pmatrix} \ldots & c_\alpha^1 & \ldots \\ & & \\ \ldots & c_\alpha^j & \ldots \end{pmatrix} \end{array} = M_{\mathrm{L}},$$

where $f_i = \sum_\alpha c_\alpha^i X^\alpha$. Note that the columns of the matrix can be arranged in different ways, depending on the order chosen to sort the multi-indices α.

To apply linearization, one now considers each (non-constant) monomial X^α as an indeterminate and attempts to solve the corresponding system of linear equations using linear algebra techniques.

The effectiveness of the method clearly depends of the number of linearly independent polynomials in the system. For example, in the case of boolean functions, the total number of monomials of degree less than or equal to 2 (excluding the constant) is $\binom{n}{2} + n$. Thus if the system consists of m polynomials of degree 2, it can be solved if the matrix M_{L} has this rank. Note that the method also tolerates a smaller rank: it is possible to perform an exhaustive search on the affine space of solutions when the dimension of the kernel of the matrix is not too large.

Concerning the complexity, we observe that the cost of the linear algebra operations is $O(N^3)$, N being the size of the matrix M_{L}. We may theoretically write $O(N^\omega)$, ω being the exponent of linear algebra, and sometimes even optimistically use $\omega \approx 2 + \epsilon$ in the case of sparse matrices.

Linearization has been considered in the cryptanalysis of LFSR-based, filtered, stream ciphers. As stated before, each new bit of the key stream gives rise to a new equation on the key bits, and by using a large number of bits from the key stream, one should have in theory enough equations to directly apply linearization. Note however that no practical attack has been reported to have been implemented using linearization, and the problem of estimating the rank of the linearized system is still unsolved (even if experimental results on attacking reduced versions of Toyocrypt point out that the number of linear dependencies is limited in these cases).

2.2.2 The XL algorithm and variants

In order to apply the linearization method, the number of *linearly independent* equations in the system needs to be approximately the same as the number of terms in the system. When this is not the case, a number of techniques have been proposed that attempt to generate enough LI equations. The most publicized is the XL algorithm (standing for *eXtended Linearization*), which was introduced in [10]. The XL algorithm aims at introducing new rows to the matrix M_{L}, by multiplication of the original equations by monomials of prescribed degree. More specifically, the following matrix M_{XL} is constructed:

$$\begin{array}{c} \\ \vdots \\ X^\beta f_1 \\ \vdots \\ X^{\beta'} f_m \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} \cdots & X^\alpha & \cdots \end{array} \\ \begin{pmatrix} & \vdots & \\ \cdots & c_\alpha^{1,\beta} & \cdots \\ & \vdots & \\ \cdots & c_\alpha^{j,\beta'} & \cdots \\ & \vdots & \end{pmatrix} \end{array} = M_{\mathrm{XL}},$$

where the set of the rows is constructed from all products $X^\beta f_j = \sum_\alpha c_\alpha^{j,\beta} X^\alpha$, where β and f_j are such that $\deg(X^\beta f_j) \leq D$, D being a parameter of the algorithm. The hope is that at

least one univariate equation (say in X_1) will appear after the Gaussian elimination on M_{XL}. This equation should be easily solved over the finite field, the found values substituted in the equations, and the process repeated for the other indeterminates X_i, $i \geq 2$. One expects that after a few iterations, the algorithm will yield a solution for the system.

To estimate the complexity of the XL algorithm, the problem is to find D such that XL succeeds with parameter D. Since the number of monomials of total degree $\leq D$ in $k[X_1, \ldots, X_n]$ is equal to $\binom{n+D}{D}$, there is an exponential dependance on D.

Since the introduction of the XL method, a number of variants have been proposed attempting to exploit some specific properties of the polynomial system [11]. Of particular relevance for the analysis of block ciphers is the method proposed in [12]. The *XSL* method is based on the XL algorithm, but uses the sparsity and specific structure of the equations; instead of multiplying the equations by all monomials of degree $\leq D-2$ (supposing that the original equations were quadratic), in the XSL algorithm the equations are multiplied only by "carefully selected monomials" [12]. This has the intention to create less new terms when generating the new equations.

XSL is an *ad hoc* method, and estimating its complexity is not a simple task. Currently it is not known whether it can be used to efficiently solve the systems of equations derived from the AES.

2.2.3 Gröbner bases algorithms

Gröbner bases algorithms are perhaps the best known technique for solving polynomial systems. These algorithms return a basis for the ideal derived from the set of equations, which can then be used to obtain the solutions of the system. The most accessible historical reference is [6], while the book [13] presents a gentle introduction to the topic together with the basics of algebraic geometry (it does not however included the more recent algorithms F_4 and F_5).

We now give a definition of a Gröbner basis of an ideal. Let $\preceq$ be a monomial order, i.e. a total order on the set of monomials X^α, $\alpha \in \mathbf{N}^n$, which is compatible with multiplication. Then the set of terms $c_\alpha X^\alpha$ of a polynomial $f = \sum_\alpha c_\alpha X^\alpha \in k[X_1, \ldots, X_n]$ can be ordered with respect to $\preceq$, and the notion of leading term $\mathrm{LT}(f)$, leading monomial $\mathrm{LM}(f)$ and leading coefficient $\mathrm{LC}(f)$ of the polynomial f are all well defined.

Let $I \subseteq k[X_1, \ldots, X_n]$ be an ideal and $\mathrm{LM}(I) = \{\mathrm{LM}(f)\ ; f \in I\}$ the set of leading monomials of polynomials in I. A Gröbner basis of the ideal I is a set $G = \{g_1, \ldots, g_l\} \subset I$ such that:

$$\mathrm{LM}(I) = \bigcup_{i=1}^{l} \mathrm{LM}(g_i).\{X^\alpha,\ \alpha \in \mathbf{N}^n\}.$$

In other words, G is a Gröbner basis of I if the leading term of any polynomial in I is divisible by the leading term of some polynomial of G. One can show that every non-empty ideal $I \subseteq k[X_1, \ldots, X_n]$ has a Gröbner basis (which however is not unique). It is to be stressed that the notion of Gröbner basis is a mathematical one, independently of any algorithm computing Gröbner bases.

There is also the notion of a *Gröbner basis of degree D* of an ideal I (denoted by G_D),

which has the property that the leading monomial of every polynomial in I of degree $\leq D$ is divisible by the leading monomial of a polynomial of G_D. It can be shown that there exists D large enough such that G_D is a Gröbner basis of I.

Gröbner bases algorithms are powerful tools for solving systems of polynomial equations. In most cases, when the Gröbner basis is found, the solution is also found. For most cryptographic applications, we will have a system with unique solution, say $(a_1, \ldots, a_n) \in \mathbf{F}_2^n$, and the ideal is radical. Then the *reduced* Gröbner basis of I is $\{X_1 - a_1, \ldots, X_n - a_n\}$.

The Buchberger algorithm

The Buchberger algorithm is the classical algorithm for computing the Gröbner basis of an ideal I. It works based on a generalization of the Euclidean division of polynomials in one variable to the multivariate case. More precisely, given a monomial order, there exists an algorithm `division`$(f, f_1, \ldots, f_l) = (g_1, \ldots, g_l, r)$ with the following properties: $f = f_1 g_1 + \ldots f_l g_l + r$, and no leading monomial of the g_i divides r. Then a Gröbner basis of an ideal generated by $f_1, \ldots, f_l$ can be computed by the following algorithm (Buchberger algorithm):

Initialize: $G = \{f_1, \ldots, f_l\}$

Loop

1. Combine every pair f_i, f_j by cancelling leading terms, to get $S(f_i, f_j)$ (the S-polynomials);
2. Compute the remainders of the $S(f_i, f_j)$ by G;
3. Augment G with the non-zero remainders.

Until all remainders are zero.

Return G.

One can show that this algorithm terminates and computes a Gröbner basis of the ideal generated by $f_1, \ldots, f_l$. It is a fact that most S-polynomials generated in step 1 will reduce to zero, and therefore many useless computations leading to zero remainder are performed. The algorithm can be modified to include *Buchberger's criteria* [5], which are a priori conditions on the pairs (f_i, f_j) to detect the ones whose S-polynomial will have a remainder equal to zero, and therefore discard them from steps 1, 2 of the algorithm. While a great proportion of pairs will be discarded by the criteria, still many S-polynomials constructed will reduce to zero, as experienced in reported implementations.

The complexity of the Buchberger algorithm is closely related to the total degree of the intermediate polynomials that are generated during the running of algorithm. In the worst case, it is known to run in double exponential time. Regarding implementation, there are number of optimizations that can be made to improve the performance of the algorithm. For example, the main loop of the algorithm can be sliced into loops of finer grain, and instead of combining every possible pair, pairs can be successively selected with respect to some strategy, and steps 2 and 3 can be performed with this selection. For instance, the most recent pairs from G can be chosen; alternatively, the pairs of smallest total degree may also be chosen.

The F_4 and F_5 algorithms

The F_4 algorithm [15] can be roughly sketched as a matricial version of the Buchberger algorithm. To introduce the idea, we first depict the Euclidean division for univariate polynomials $f = f_d X^d + \cdots + f_1 X + f_0$ and $g = g_{d'} X^{d'} + \cdots + g_1 X + g_0$, with $d' \leq d$, as a matrix reduction algorithm. Consider the following:

$$\begin{array}{l} \\ f \\ X^{d-d'}g \\ X^{d-d'-1}g \\ X^{d-d'-2}g \\ \vdots \\ g \end{array} \begin{array}{c} \begin{array}{cccccccc} X^d & X^{d-1} & & & & & & X^0 \end{array} \\ \begin{pmatrix} f_d & f_{d-1} & \cdots & & & & & f_0 \\ g_{d'} & g_{d'-1} & \cdots & g_0 & & & & \\ 0 & g_{d'} & g_{d'-1} & \cdots & g_0 & & & \\ 0 & 0 & g_{d'} & g_{d'-1} & \cdots & g_0 & & \\ \vdots & & & \ddots & & & \ddots & \\ 0 & 0 & 0 & 0 & g_{d'} & g_{d'-1} & \cdots & g_0 \end{pmatrix} \end{array} \tag{1}$$

Then successive reductions of the first row by the remaining rows (row echelon reduction by elementary row operations) give the remainder of f by g. Similarly the multivariate division algorithm can be written in a matrix fashion.

At each iteration of the F_4 algorithm, corresponding to each iteration in the Buchberger algorithm, and subject to the selection strategy, the two parts f_i and f_j of the selected pairs (f_1, f_j) are written in a global matrix M_{F_4}. Now the crucial point of the algorithm F_4 it to write, at a given step of the algorithm, all the considered f_i and f_j into this global matrix, together with all already known polynomials of the current basis G, multiplied by fitting monomials, in the same way as the polynomial g is shifted in the matrix in (1).

Then, in a single step corresponding to one iteration in the Buchberger algorithm, a huge matrix reduction operation (computation of the row echelon form) is done on the matrix M_{F_4}. In contrast to the Buchberger algorithm, where each remainder is computed separately and sequentially, this global reduction operates all reductions of all polynomials by all multiples of polynomials in the current basis G. It turns out that, properly implemented, this is a big win [15, 21]. Additionally, the algorithm can also benefit from any optimization technique from linear algebra algorithmics which can be applied here.

The F_5 algorithm [16] is an optimized version of the F_4 algorithm, with an optimal F_5 criterion, generalizing Buchberger's criteria, for discarding *any* pair which will reduce to zero in the remaining computations. That is, the rank of the constructed matrix M_{F_5} is equal to the number of its rows (property of full rank).

Relationship between these algorithms

Recent research has shown that some of the algorithms introduced above are related. In fact, let M_∞ denote the Macaulay matrix with an infinite number of rows and columns, defined as

$$\begin{array}{c} \\ \vdots \\ X^\beta f_i \\ \vdots \\ X^{\beta'} f_j \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} \dots & X^\alpha & \dots \end{array} \\ \begin{pmatrix} & \vdots & \\ \dots & c_\alpha^{1,\beta} & \dots \\ & \vdots & \\ \dots & c_\alpha^{j,\beta'} & \dots \\ & \vdots & \end{pmatrix} \end{array} = M_\infty,$$

for all monomials X^β, $X^{\beta'}$, of unbound degree. The M_{XL} matrix of the XL algorithm in degree D is therefore just a finite submatrix of the Macaulay matrix, corresponding to all monomials of degree less than or equal to D. Performing a Gaussian elimination on the Macaulay matrix is equivalent to running the Buchberger algorithm [18]. This fact is closely related to the behaviour of the XL algorithm, and it is shown in [1] that the XL algorithm terminates for a degree D if and only if it terminates in degree D for the lexicographical ordering.

Concerning F_4, we can see that the matrix

$$\begin{array}{c} \\ \vdots \\ X^\beta f_i \\ \vdots \\ X^{\beta'} f_j \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} \dots & X^\alpha & \dots \end{array} \\ \begin{pmatrix} & \vdots & \\ \dots & c_\alpha^{1,\beta} & \dots \\ & \vdots & \\ \dots & c_\alpha^{j,\beta'} & \dots \\ & \vdots & \end{pmatrix} \end{array} = M_{F_4}$$

is constructed only from pairs (f_i, f_j) originating from the previous iterations of the algorithm, and which are not discarded by the Buchberger criteria. This shows that M_{F_4} is a very small submatrix of the matrix M_{XL} constructed by XL. Using an XL description as an F_4 algorithm, it is proven [1] that *a slightly modified XL computes a Gröbner basis.*

We note that things are pushed further in the same vein, when one considers the F_5 algorithm, which constructs a matrix M_{F_5} with even less rows than M_{F_4}. In [1], an example is given for the case of 130 equations in 128 variables, where the number of rows in the matrix M_{XL} will be more than 10 thousands times the number of rows in the matrix M_{F_5}.

2.3 Complexity bounds

We now state some results on the complexity of the algorithms introduced above, taken from [14, 22], which focus on the XL algorithm, and from [3], which focus on F_5. In both cases, the concepts of Hilbert Theory are behind the scenes, and it is important to understand it to properly analyze these algorithms. In the case of XL, the point is to find the degree D such that an univariate equation in the last variable X_1 can be found after the Gaussian elimination process.

For a "generic system" of quadratic polynomials, we have the following complexity result: when the number of polynomials is $m = n + c$, then the minimum degree for XL to succeed is

$$D \geq \frac{n}{\sqrt{c-1}+1}.$$

Since the number of monomials is $\binom{n}{D}$ in the boolean case, and $\binom{n+D}{D}$ in the general case, this theorem implies that XL has an exponential complexity.

For F_5, the highest degree which appears in the algorithm is when the number of rows is equal to the number of columns (because of the property of full rank). This reasoning is exact since the F_5 criterion eliminates all rows which reduce to zero. In [3], we are presented with the following result for quadratic polynomials over $\mathbf{F}_2$: for n equations in n variables (without counting the field equations), the degree for which F_5 stops is approximately $D \approx 0.09n$, the approximation being valid even for small n. This also implies exponential complexity for F_5.

For generic systems (over $\mathbf{F}_2$) the results are the following [3]: when m grows linearly with n, the size of the matrix is exponential in n, and the complexity of F_5 is exponential; when n/m tends to zero, F_5 is subexponential; and when m grows as Nn^2, F_5 has polynomial complexity, with exponent depending on N.

Application to cryptology

The results above indicate that, being of exponential nature, the algorithms introduced earlier should be of very limited use in cryptology, given the large sizes involved. It is known however that F_5 was used successfully for solving the HFE challenge I [17]. In fact, this experiment has also been reproduced with an independent implementation of F_4 [21], and it now takes a few hours to break HFE challenge I with affordable Magma software [20] on a workstation.

The reason is that the theorems above hold for *generic* systems, which are basically systems with no particular properties. In the case on finite fields, random systems take the role of generic systems. In mathematical terms, they are *semi-regular* sequences, which are conjectured to be the "generic" case. It turns out however that the HFE systems of equations are not random-like [8, 17], and it appears that F_5 (and also F_4 as reported in [21]) is sensitive to this fact, i.e. it is a distinguisher for HFE systems. More precisely, the degree for which the matrix M_{F_5} has large enough rank is much lower than the generic bound. From the implementation of XL made in [1], it seems that the XL algorithm is not sensitive to this fact.

For the case of block ciphers, although no practical attack has ever been reported, it appears they also give rise to very structured systems. Table 1 is an extension (from [2]) of the table given in [4], where systems of quadratic equations have been constructed for various ciphers; in this case the *expected* degree reached by the F_5 algorithm and the size of the matrices have been added. We can see that the expected degrees are quite large and that the matrices should be in principle too large to be tractable.

One should bear in mind however that the sizes given in Table 1 are the ones that would be reached if these systems were generic. It may be well that the systems are non generic, in which case F_5 may succeed with a lower D and smaller matrices. With the current state of knowledge, only practical experiments could tell how these systems behave.

Cipher	Variables	Linear equations	Quadratic equations	D	Matrix size
Khazad	6464	1664	6000	379	2^{2076}
Misty1	3856	2008	1848	179	2^{1040}
Kasumi	4264	2264	2000	193	2^{1129}
Camelia-128	3584	1920	4304	78	2^{538}
Rijndael-128	3296	1696	4600	69	2^{479}
Serpent-128	16640	8320	9360	703	2^{4196}

Table 1: The degree D for the systems of equations constructed in [4]

2.4 Research Directions

Algebraic attacks have received a lot of attention of the cryptographic community in the last few years. Although they have been considered against the HFE cryptosystem as well as a number of LFSR-based stream ciphers, there has not been too much progress in assessing whether they can be effective against block ciphers. The main reason is that the size of systems arising from block ciphers are completely out of reach for the current computational power. While for most methods of cryptanalysis it is quite straightforward to perform experiments on reduced versions of the cipher to understand how the attack might perform, this has not been the case for algebraic attacks on block ciphers. One possible direction to follow is the introduction of toy examples of symmetric ciphers, in order to test the effectiveness of the main algorithms in solving the systems of algebraic equations. However it is not an easy task to design small versions that can replicate the main cryptographic and algebraic properties of the cipher. Currently there is some work being performed on small scale variants of the AES [7], and the hope is that this can provide a preliminary insight into the behaviour of algebraic cryptanalysis on the AES.

Additionally, there is much that can be done on the algorithmic side as well. Computer algebra is a well established subject, and some of the algorithms presented earlier have been known and extensively studied for a number of years now. For example, Gröbner bases algorithms are known to be a powerful tool for solving systems of polynomial equations. They are however general-purpose algorithms, which are used to deal with a number of problems arising in algebraic geometry (including computing the solutions of a system). They may well prove to be an overkill when considered in the context of cryptography. The systems arising from symmetric ciphers are very structured and with special characteristics: they are usually sparse, with unique solution over a finite field, structured in blocks of similar format (rounds), etc. Perhaps the most promising approach would be the development of *dedicated* methods for specific block ciphers. These could be built upon known techniques from computer algebra, but aiming to exploit the special properties of a particular system. This theoretical approach together with experiments with small versions of the ciphers can hopefully shed some light on how effective algebraic attacks can be against symmetric ciphers.

References

[1] G. Ars, J.-C. Faugère, H. Imai, M. Kawazoe, and M. Sugita. Comparison between XL and Gröbner basis algorithms. In *Advances in Cryptology - ASIACRYPT 2004*, volume

3329 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.

[2] M. Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie*. PhD thesis, Université Paris VI, 2004.

[3] M. Bardet, J.-C. Faugère, and B. Salvy. Complexity of Gröbner basis computation for semi-regular overdetermined sequences over F_2 with solutions in F_2. Technical Report 5049, INRIA, December 2003. `http://www.inria.fr/rrrt/rr-5049.html`.

[4] A. Biryukov and C. de Cannière. Block ciphers and systems of quadratic eqautions. In *Fast Software Encryption - FSE 2003*, volume 2887 of *Lecture Notes in Computer Science*. Springer Verlag, 2003.

[5] B. Buchberger. A criterion for detecting unnecessary reductions in the construction of Groëbner basis. In *Symbolic and Algebraic Computation, EUROSAM '79, An International Symposium on Symbolic and Algebraic Computation*, volume 72 of *Lectures Notes in Computer Science*. Springer Verlag, 1979.

[6] B. Buchberger. Gröebner bases: an algorithmic method in polynomial ideal theory. In *Multidimensional Systems Theory*. D. Reidel Publishing Company, 1985,.

[7] C. Cid, S. Murphy, and M. Robshaw. Small Scale Variants of the AES. In *Fast Software Encryption - FSE 2005*, Lecture Notes in Computer Science. Springer Verlag, 2005.

[8] N. T. Courtois. The security of hidden field equations (HFE). In *Progress in Cryptology - CT-RSA 2001: The Cryptographers' Track at RSA Conference 2001*, volume 2020 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001.

[9] N. T. Courtois. Higher order correlation attacks, XL algorithm and cryptanalysis of Toyocrypt. In *Information Security and Cryptology - ICISC 2002: 5th International Conference*, volume 2587 of *Lecture Notes in Computer Science*. Springer-Verlag, 2003.

[10] N. T. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Advances in Cryptology - EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*. Springer-Verlag, 2000.

[11] N. T. Courtois and J. Patarin. About the XL algorithm over GF(2). In *Topics in Cryptology - CT-RSA 2003: The Cryptographers' Track at the RSA Conference 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 141–157, 2003.

[12] N. T. Courtois and J. Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *Advances in Cryptology - Asiacrypt'02*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer-Verlag, 2002.

[13] D. Cox, J. Little, and D. O'Shea. *Ideals, varieties, and algorithms*. Undergraduate Texts in Mathematics. Springer-Verlag, second edition, 1997. An introduction to computational algebraic geometry and commutative algebra.

[14] C. Diem. The XL algorithm and a conjecture from commutative algebra. In *Advances in Cryptology - ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.

[15] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *Journal of Pure and Applied Algebra*, 1999.

[16] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*. ACM, 2002.

[17] J.-C. Faugère and A. Joux. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using gröbner bases. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*. Springer-Verlag, 2003.

[18] D. Lazard. Gröbner-bases, Gaussian elimination and resolution of systems of algebraic equations. In *Proceedings of the European Computer Algebra Conference on Computer Algebra*, volume 162 of *Lectures Notes in Computer Science*. Springer-Verlag, 1983.

[19] S. Murphy and M. Robshaw. Essential Algebraic Structure within the AES. In *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 1–16. Springer-Verlag, 2002.

[20] University of Sydney. The Magma computational algebra system. `http://magma.maths.usyd.edu.au`.

[21] A. Steel. Allan Steel's Gröbner basis timings page, 2004. `http://magma.maths.usyd.edu.au/users/allan/gb/`.

[22] B.-Y. Yang and J.-M. Chen. Theoretical analysis of XL over small fields. In *Information Security and Privacy: 9th Australasian Conference, ACISP 2004*, volume 3108 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.

3 Design of Symmetric primitives

Many attacks on conventional cryptographic algorithms are related to some properties of the Boolean functions describing the system. The formalization of well-known attacks such as linear or differential cryptanalysis on block ciphers, correlation attacks against LFSR-based stream ciphers... have led to the definitions of some relevant quantities related to Boolean functions. These quantities measure the resistance of a cryptosystem to classical attacks. For instance, it has been proved that the use of highly nonlinear substitution functions in a block cipher guarantees a high resistance to linear cryptanalysis. The resistance to differential attacks can also be quantified by some properties of the substitution function. For stream ciphers, it is well-known that the use of a correlation-immune Boolean function in combining generators avoids correlation attacks. The designers of symmetric ciphers now provide evidence that their ciphers cannot be broken by these classical attacks.

However, many problems remain open and there is no exact recipe for choosing the composing functions of a symmetric algorithm. The first reason is that the well-known criteria on Boolean functions are obviously necessary but not sufficient security conditions. Some new cryptanalytic techniques have recently been proposed. They have introduced new requirements which must be investigated. The major recent breakthrough in this area is certainly the development of algebraic attacks against both block and stream ciphers. It has led to the definition of a new criterion, called algebraic immunity. Many issues related to this new parameter must be studied. For instance, there is a need for a fast algorithm which determines the algebraic immunity of a function with many variables. It is also necessary to determine the relationships between algebraic immunity and the other criteria; at the moment, it remains unclear whether some previously known criteria are compliant with a high algebraic immunity. Another major problem which is still completely open is to find some general constructions for functions which guarantee an optimal algebraic immunity.

Besides these new criteria, there is still a need for constructions of functions which provide an optimal resistance to classical attacks. Even the optimal values for some classical quantities, such as the nonlinearity, are still unknown. Moreover, there are often some trade-offs between all criteria, and it is very important to determine which of them are really relevant in a particular context, and then to give some general constructions dedicated to a given application. Suitable constructions of Boolean functions and S-boxes must also include some implementation constraints.

Finally, optimizing the resistance to some classical attacks usually leads to the choice of functions which possess very particular structures. For instance, all known S-boxes which satisfy maximal resistance to both differential and linear attacks are linearly equivalent to some power functions over a finite field. The intuition is that such a strong algebraic property may introduce a weakness which could be exploited by another attack. Many other specific structural properties appear in the study of optimal objects. But, it is not clear if these properties are really relevant since no associated attack has been found so far. It is then essential to try investigate these structural properties from a cryptanalytic point of view and to determine whether they induce a weakness or not.

3.1 Boolean functions for stream ciphers

LFSR-based stream ciphers are very popular because LFSRs are appropriate for low-cost hardware implementations, produce sequences with good statistical properties and can be easily analyzed. For the two most well-known families, combining generators and filter generators, the security relies primarily on the Boolean function used either for combining several LFSRs or for filtering the state of a single register. In such systems, the Boolean function is actually the only component which breaks the linearity inherent to the LFSRs. In practical applications, some additional techniques are usually used in order to make attacks on these ciphers more difficult. The most commonly used method consists in clocking the involved LFSRs in an irregular way, as in LILI-128, A5/1, ... But, the weaknesses of the regularly clocked system may nevertheless be exploited for the irregularly clocked generator. For instance, a distinguisher for the simple underlying generator can be used in a divide-and-conquer attack which performs an exhaustive search on the clocking subsystem only. For these reasons, it is often recommended that even more complex LFSR-based ciphers follow the basic design criteria required for the simple underlying generator.

3.1.1 Filtering functions

In a filter generator, it is well-known that the inherent Boolean function must satisfy the following 3 properties. It must be *balanced*, i.e., its output must be uniformly distributed. It must possess a *high algebraic degree* in order to guarantee a high linear complexity. It must have a *high nonlinearity*, i.e., it must lie as far as possible from all functions of degree 1. Otherwise the existence of a good affine approximation can be exploited in a fast correlation attack. In this case, the keystream is seen as a noisy version of the output of an LFSR with known feedback polynomial, and the initial state of the register can be recovered by applying some decoding algorithms. When an n-variable filtering function is used, the error-probability that the attacker has to correct corresponds to the nonlinearity of the function divided by 2^n.

Constructing filtering functions which satisfy the previous three criteria is not a completely solved problem, even though it has been investigated for several years. Actually, the highest possible nonlinearity for a balanced Boolean function of n variables is still unknown for $n \geq 8$. There are two major results in this field: the first one is a general recursive construction due to Dobbertin [30] which consists in modifying a normal bent function in order to obtain a balanced and highly nonlinear function. By definition, a normal bent function f of n variables is constant on an affine subspace V and balanced on all other cosets of V. Therefore, replacing the restriction of f to V by a highly nonlinear function g of $n/2$ variables leads to a balanced function with nonlinearity $2^{n-1} - 2^{n/2} + \mathcal{NL}(g)$. This construction can be iterated if $n/2$ is even, and it provides a lower bound on the best achievable nonlinearity for a balanced function. It is worth noticing that bent functions are essential ingredients in this construction. It is therefore necessary to go further in the study and in the construction of bent functions (even if they may have no direct cryptographic applications because they are not balanced) if we want to find many highly nonlinear balanced functions with good cryptographic properties. For instance, the simple Maiorana-McFarland family of bent functions provide many functions, but they seem to be inherently vulnerable to algebraic attacks. A more specific construction of highly nonlinear balanced functions of n variables has been proposed in [65] for some values of n. These functions have been obtained by computer from the well-known Paterson-

Wiedemann's functions. The proposed algorithm leads to balanced functions of $n = (15 + 2k)$ variables having nonlinearity strictly greater than $2^{n-1} - 2^{\frac{n-1}{2}}$ (which is the highest nonlinearity achievable by quadratic functions).

The design of good filtering then presents both theoretical aspects (such as the determination of the best achievable nonlinearity for a balanced function) and practical issues. Actually, a function of more than 15 variables can be used in practice only if it can be represented in a compact form (e.g. with a sparse algebraic normal form). Therefore, it seems very important to focus on the cryptographic properties of Boolean functions which can be implemented with a few gates. Symmetric functions (i.e., Boolean functions whose output only depends on the Hamming weight of the input) are good candidates in terms of implementation complexity because they are the only functions having a known implementation with a number of gates which is linear in the number of input variables [74]. But, they have generally bad cryptographic properties as shown in [55, 73]. Some generalizations such as rotation symmetric functions [36, 71] seem to be better from a cryptographic point of view, but further work needs to be done in this direction, including both cryptographic and implementation aspects.

3.1.2 Combining functions

When a Boolean function is used for combining the outputs of several LFSRs, it must obviously be balanced and have a high algebraic degree for the same reasons as for a filtering function. Two particular requirements are specific to the case of combining functions. They must have a *high order of correlation-immunity* (also called order of resiliency when the function is balanced). The correlation-immunity order is the highest number t such that the output of the function remains balanced when any t input variables are fixed. The cryptographic significance of this parameter is as follows: for a t-th order correlation-immune function, $(t + 1)$ is the minimal number of LFSRs that must be considered together in a correlation attack. In other words, a classical correlation attack (as defined by Siegenthaler) requires an exhaustive search for the initial states of at least $(t + 1)$ LFSRs. The second important requirement for the combining function relates to the Hamming distance between the combining function and the affine functions of $(t + 1)$ variables. Indeed, in a (fast) correlation attack, the attacker exploits the existence of a correlation between the running-key and the output of a smaller combination generator, comprising k of the LFSRs combined by a Boolean function g of k variables. If the combining function is t-th order correlation-immune, the minimal value for k is $(t+1)$ and the highest correlation is obtained when g is the $(t+1)$-variable function closest to the combining function in terms of Hamming distance. Then, it can be proved [12, 75] that the $(t + 1)$-variable function g which maximizes the correlation is an affine function. The error-probability that the attacker has to correct in a fast correlation then corresponds to the lowest Hamming distance between the n-variable combining function and the affine functions depending on exactly t input variables. For this reason, the nonlinearity of the combining function provides an upper bound on the error-probability: the use of a highly nonlinear combining function prevents fast correlation attacks. But, this is not a necessary condition since the combining function must only be far from all affine functions depending on few variables, and not from all affine functions.

One of the main problems in designing good combining functions is that there exist some trade-offs between the previously mentioned criteria. Most notably, the degree of a balanced

and t-th order correlation-immune function of n variables is at most $(n-t-1)$. Its nonlinearity and its distance to affine functions of $(t+1)$ variables is also upper-bounded because the Walsh coefficients of a t-th order correlation-immune and balanced function are divisible by 2^{t+2} [66]. The divisibility is even higher when the degree of the function is not maximal, i.e., when it is strictly less than $(n-t-1)$ [14, 17]. Some constructions of balanced t-th order correlation-immune functions of n variables have been proposed (e.g. [66, 72, 54, 67, 63]...), but only for particular values of n and t. Only a few practical constructions are available to the designers of combination generators. As noted in [67], the construction of optimal correlation-immune and balanced functions of 8, 9 and 10 variables is still open. Moreover, it seems that most proposed constructions are weak regarding algebraic attacks. For instance, the 3-resilient function of 10 variables presented in [66] has been used as a filtering function in LILI-128, but it does not guarantee an optimal resistance to algebraic attacks because of the existence of many relations of degree 4 between its output and its inputs.

3.1.3 Algebraic immunity of Boolean functions

As pointed out by Courtois and Meier [24], all LFSR-based stream ciphers are vulnerable to algebraic attacks if there exist relations of low degree between the output and the inputs of the associated Boolean function f. Such relations correspond to low degree multiples of f, i.e., to relations $g(x)f(x) = h(x)$ for some g where h has a low degree. But it was proved in [35, 56] that, in the case of algebraic attacks over $\mathbf{F}_2$, the existence of any such relation is equivalent to the existence of a low degree function in the annihilator ideal of f or of $(1+f)$. Indeed, if $g(x)f(x) = h(x)$ with $\deg(h) \leq d$, we obtain, by multiplying this equation by $f(x)$, that

$$g(x)\,[f(x)]^2 = h(x)f(x) = g(x)f(x) = h(x)\ ,$$

leading to $h(x)\,[1+f(x)] = 0$.

Suppose that the keystream bit at time t, s_t, is obtained by applying f to the current internal state of the generator, $s_t = f(x_t)$. Then, the algebraic attack exploits the following relations:

- if $s_t = 1$, any function g of degree at most d in the annihilator ideal of f, $AN(f) = \{g \mid g(x)f(x) = 0,\ \forall x\}$ leads to $g(x_t) = 0$;
- if $s_t = 0$, any function g' of degree at most d in $AN(1+f)$ leads to $g'(x_t) = 0$.

If the transition function $x_t \mapsto x_{t+1}$ is linear, the internal state of the generator at time t is a linear function of the initial state: $x_t = L_t(x_0)$. The previous relations then provide a system of equations of degree d depending on ℓ variables where ℓ is the number of bits of initial state. This system can be solved with one of the general techniques discussed in Section 2. The simplest one, called *linearization*, consists in identifying the system with a linear system of $\sum_{i=1}^{d} \binom{n}{i}$ variables, where each product of i bits of the initial state ($1 \leq i \leq d$) is seen as a new variable. The entire initial state is then recovered by a Gaussian reduction (or by more sophisticated techniques) whose time complexity is roughly

$$\left(\sum_{i=1}^{d} \binom{n}{i}\right)^{\omega} \simeq \ell^{\omega d}\ ,$$

where ω is the exponent of the matrix inversion algorithm, i.e., $\omega \simeq 2.37$ [21]. This rough estimate for the time complexity will be used later for computing the suitable parameters for the Boolean functions used in LFSR-based stream ciphers. But, it should be noted that algebraic attacks have usually been applied against filter generators, and that it is not clear whether the complexity may not be lower for combining generators because of the particular structure of the algebraic system (the internal state of LFSR i at time t only depends on the initial state of this LFSR and not on all bits of the initial states).

The cryptographic relevance of the algebraic immunity. The relevant parameter in the context of algebraic attack, called the *algebraic immunity* of the Boolean function, $AI(f)$, is the lowest degree achieved by a function in $AN(f) \cup AN(1+f)$. A simple combinatorial argument implies that, for any Boolean function f of n variables, $AI(f) \leq \lceil n/2 \rceil$. We deduce from this bound that if an n-variables function is used in the generator, the complexity of the algebraic attack will be at most $\ell^{\frac{\omega n}{2}}$, where ℓ is the size of the internal state. This value must be higher than the complexity of an exhaustive search for the key. We here suppose that the size of the internal state is minimal with respect to key-size k, i.e. that $\ell = 2k$ (it is known that the size of the internal state must be at least twice the key size in order to resist time-memory trade-off attacks). Therefore, we must have $(2k)^{\frac{\omega n}{2}} \geq 2^k$, i.e.,

$$n \geq 0.84 \left\lceil \frac{k}{1+\log_2(k)} \right\rceil \ .$$

For instance, a filter generator with a 128-bit key and a 256-bit LFSR must use a filtering function of at least 16 variables. Note that the recommended number of variables is probably higher than the previous bound because more efficient techniques can be used for solving the algebraic system (see Section 2).

Properties of the annihilator ideal of a Boolean function. The set $AN(f)$ of all annihilating functions of f is obviously an ideal in the ring of all Boolean functions, and it is generated by $(1+f)$. It consists of the $2^{2^n - wt(f)}$ functions of n variables which vanish on the support of f, i.e., on all x such that $f(x) = 1$, where $wt(f)$ denotes the size of the support of f. It is important to note that the number of functions with a given degree in $AN(f) \cup AN(1+f)$ is less important from a cryptanalytic point of view than the algebraic immunity: the number of such annihilating functions only influences the number of keystream bits required for the attack, and not the time-complexity (except maybe for some refinements such as fast algebraic attacks).

The number of functions of degree at most d in $AN(f)$ is equal to 2^κ where κ is the dimension of the kernel of the matrix obtained by restricting the Reed-Muller code of length 2^n and order d to the support of f. In other words, the rows of this matrix correspond to the evaluations of the monomials of degree at most d on $\{x, f(x) = 1\}$. Since this matrix has $\sum_{i=0}^{d} \binom{n}{i}$ rows and $wt(f)$ columns, its kernel is non-trivial when

$$\sum_{i=0}^{d} \binom{n}{i} > wt(f) \ .$$

Similarly, $AN(1+f)$ contains some functions of degree d or less if

$$\sum_{i=0}^{d} \binom{n}{i} > 2^n - wt(f) \ .$$

This shows, as pointed out in [29], that the algebraic immunity of an n-variable function is related to its Hamming weight. Most notably, for odd n, only balanced functions can have optimal algebraic immunity. For even n, the Hamming weight of a function with optimal algebraic immunity must satisfy

$$\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i} \leq wt(f) \leq \sum_{i=0}^{\frac{n}{2}} \binom{n}{i}$$

Algebraic immunity of 5-variable balanced functions. For 5-variable functions, it is possible to compute the algebraic immunity of all Boolean functions using the classification due to Berlekamp and Welch (because algebraic immunity is invariant under composition by a linear permutation). We here focus on balanced functions because they are the only ones that may have optimal algebraic immunity for n odd. First, we compute the minimum degree of $AN(f)$ for all $601,080,390$ balanced functions of 5 variables:

$\min \deg(AN(f))$	1	2	3
nb. of balanced f	62	403,315,208	197,765,120
proportion of balanced f	10^{-7}	0.671	0.329

The same results hold for the algebraic immunity since we observe that the minimum degree in $AN(f)$ is always the same as the minimum degree in $AN(1+f)$ for balanced functions of 5 variables. We have also computed the exact dimension of $AN(f)$.

$\dim(AN(f))$	0	1	2	3	4	5
nb. of balanced f	197,765,120	345,283,456	56,801,920	1,213,960	15,872	62
proportion	0.329	0.574	0.094	0.002	$2 \cdot 10^{-5}$	10^{-7}

It is worth noticing that both $AN(f)$ and $AN(1+f)$ have the same dimension except for one function f up to linear equivalence (i.e., to 555,520 functions), for which $\dim(AN(f)) = 2$ and $\dim(AN(1+f)) = 1$ (and for its complement). Similar simulations can be performed as far as the functions of n variables are classified into equivalence classes under composition by a linear permutation (see e.g. [7]). But, such a classification only exist for $n = 6$ and for cubic functions up to 8 variables.

Algebraic immunity of random balanced functions. Even if some well-known constructions of cryptographic Boolean functions have been proved to have a low algebraic immunity, probabilistic arguments tend to show that the proportion of balanced functions with low algebraic immunity is very small. It has been proved in [56] that the probability that a balanced function of n variables has algebraic immunity less than $0.22n$ tends to zero when n tends to infinity. An upper bound on the probability that a balanced function has an annihilator of degree less than d is also given. This bound involves a part of the weight enumerator

of $RM(d,n)$ and any new information on its complete weight distribution can clearly improve the result. However, this bound does not say anything on the average value of the algebraic immunity or on the proportion of balanced functions with optimal algebraic immunity.

The proportion of balanced functions with optimal algebraic immunity obviously corresponds to the probability that a subset of 2^{n-1} columns of the Reed-Muller code of length 2^n and of order $\lceil n/2 \rceil$ has maximal rank. If we assumed that the generator matrices of the Reed-Muller codes behave like random matrices, we would deduce that the probability that a balanced function has optimal algebraic immunity is (almost) constant. More precisely, it would be deduced for n even, that the probability that $AN(f)$ has minimal degree $n/2$ is almost 1 and, for n odd, that the probability that $AN(f)$ has minimal degree $\frac{n+1}{2}$ (resp. $\frac{n-1}{2}$) is 0.289 (resp. 0.711). One can first observe a difference with the results obtained for $n = 5$, which is not very surprising because $RM(2,5)$ does not behave like a random code (its weight distribution is clearly not close to the distribution expected for a random code with similar parameters). Moreover, simulations tend to show that the situation differs very much from the expected one. Actually, the proportion of balanced functions of n variables with optimal algebraic immunity seems to decrease when n increases, and the average value of the algebraic immunity appears to decrease with n.

3.1.4 Algebraic immunity and other cryptographic criteria

Besides the Hamming weight of the function, its nonlinearity is also related to its algebraic immunity [29]. It can be proved that, for any linear function φ, the algebraic immunity of $f + \varphi$ is at most $AI(f) + 1$. Therefore, any function f of n variables with algebraic immunity at least d satisfies

$$\mathcal{NL}(f) \geq \sum_{i=0}^{d-2} \binom{n}{i} .$$

It follows that any function with optimal algebraic immunity has a high nonlinearity, more precisely

$$\mathcal{NL}(f) \geq \begin{cases} 2^{n-1} - \binom{n}{\frac{n-1}{2}} & \text{if } n \text{ is odd} \\ 2^{n-1} - \frac{1}{2}\binom{n}{\frac{n}{2}} - \binom{n}{\frac{n}{2}-1} & \text{if } n \text{ is even} \end{cases}$$

A high nonlinearity and a high algebraic immunity are then compatible criteria. Another important consequence is that the nonlinearity of a function may be sufficient criteria to decide whether it has low algebraic immunity (but the converse is not true).

Another cryptographic property that implies that a function does not have a maximal algebraic immunity is the notion of *normality*. A function is said to be k-normal (resp. k-weakly normal) if there exists an affine subspace of dimension k on which the function is constant (resp. affine). Since the minimum weight codewords of $RM(r,n)$ are those whose support is an affine subspace of dimension $n - r$, we deduce that any k-normal function f of n variables has algebraic immunity at most $n-k$. Similarly, any k-weakly normal function has algebraic immunity at most $n - k + 1$. Non-normal (and non-weakly normal) functions such as the functions exhibited in [11] may be good candidates if we want to construct functions with optimal nonlinearity.

The existence of links between algebraic immunity and other cryptographic criteria remains unknown. For instance, the relation between the distance of a function to all low-degree

functions (i.e., its distance to $R(d,n)$) and its algebraic immunity is still unclear. Correlation-immunity does not seem to be a priori incompatible with optimal algebraic immunity: there exists a 1-resilient function of 5 variables with optimal algebraic immunity. However, the link with all known criteria must be investigated further.

Algebraic immunity of known constructions. Some bounds have been established on the algebraic immunity of the cryptographic functions obtained by applying some classical constructions. A first construction, called the Maiorana-McFarland family, consists in deriving an n-variable function f by concatenating 2^{n-k} affine functions of k variables. These small functions correspond to the restrictions of f to all cosets of a given subspace of dimension k. This construction is quite popular because it may lead to resilient functions and to highly nonlinear functions. It is proved in [56] that any function in $AN(f)$ corresponds to the concatenation of annihilators of its restrictions. Therefore, the algebraic immunity of f cannot exceed $(n-k+1)$. But, the existence of functions with optimal algebraic immunity in the Maiorana-McFarland family is still open when $k < \lfloor \frac{n}{2} \rfloor$.

More generally, the algebraic immunity of a function can be derived from the algebraic immunities of its restrictions to a given hyperplane and to its complement [29]. For instance, if

$$f(x_1,\ldots,x_n) = (1+x_n)f_1(x_1,\ldots,x_{n-1}) + x_n f_2(x_1,\ldots,x_{n-1}) \ ,$$

we have:

- if $AI(f_1) \neq AI(f_2)$, then $AI(f) = \min(AI(f_1), AI(f_2)) + 1$;
- if $AI(f_1) = AI(f_2)$, then $AI(f) \in \{AI(f_1), AI(f_1)+1\}$.

Therefore, it is obvious how to construct a function of $2t$ variables with optimal algebraic immunity from two functions of $(2t-1)$ variables with respective algebraic immunities equal to t and to $(t-1)$. But, constructing a function of $(2t+1)$ variables with optimal algebraic immunity from two functions of $2t$ variables is much more difficult since both restrictions must have optimal algebraic immunity and they must also satisfy some additional conditions.

Some lower and upper bounds on the algebraic immunities of the functions constructed in [63] are also known [29, 15]. This construction can lead for instance to a 7-resilient function f of 14 variables with $AI(f) = 5$. But, at the moment, none of the classical constructions is known to provide functions with optimal algebraic immunity. The definition of an infinite family of Boolean functions with optimal algebraic immunity and with other good cryptographic properties (such as high nonlinearity) is still an open problem and must be investigated.

Computing the algebraic immunity of a Boolean function. The basic algorithm for computing the algebraic immunity of an n-variable function consists in performing a Gaussian elimination on the generator matrix of the punctured $RM(\lfloor \frac{n-1}{2} \rfloor, n)$ restricted to the support of f. This matrix has $k(\lfloor \frac{n-1}{2} \rfloor, n) = \sum_{i=0}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n}{i}$ rows and $wt(f)$ columns. Therefore, the algorithm requires $k^2(\lfloor \frac{n-1}{2} \rfloor, n) wt(f)$ operations, which is close to 2^{3n-3} when f is balanced. As noted in [56], the complexity can be significantly reduced if we only want to check whether

a function has annihilators of small degree d, since we do not need to consider all positions in the support of f. Indeed, considering a number of columns which is only slightly higher that the code dimension $k(d, n)$ is usually sufficient for proving that a function does not admit any annihilator of degree d.

A technique for reducing the size of the matrix over which the Gaussian elimination is performed is presented in [56]. The idea is that the elements in the support of f with low Hamming weight provide simple equations that can be removed from the matrix by a substitution step. However, due to the lack of simulation results, it is very hard to evaluate the time complexity of the substitution step in practice.

Gröbner bases algorithms such as F5 provide other techniques for computing the size of the annihilator ideal. But they need to be compared with the basic techniques in this particular context.

3.1.5 Resistance to fast algebraic attacks and other criteria

At CRYPTO 2003, Courtois presented some important improvements on algebraic attacks, called *fast algebraic attacks* [22]. The refinement first relies on the existence of some low degree relations between the bits of the initial state and not only one but several consecutive keystream bits. In other word, the attacker wants to find some low degree relations g between the inputs and outputs of the function

$$\begin{array}{rccl} F_m\colon & \mathbf{F}_2^{\ell} & \to & \mathbf{F}_2^{m} \\ & x & \mapsto & ((f(x), f(L(x)), \ldots, f(L^m(x)) \end{array}$$

where L is the linear transition function for the internal state. This function is very similar to the so-called *augmented function* defined in [1]. The fact that the augmented function may be much weaker than the filtering function, i.e, than F_0 with the previous notation, has been pointed out by Anderson [1] in the context of (fast) correlation attacks. However, the complexity required for computing the low degree relations between the n inputs and m outputs of F_m increases with m. The direct algorithm (used for multi-output functions and described in Section 3.2.3) can only be used for small m. It is an open problem to determine whether there exist relationships between the algebraic immunity of f and the algebraic immunity of F_m. The same problem arises for other cryptographic criteria such as correlation immunity.

Since the computation of low degree relations involving several keystream bits is usually infeasible, Courtois proposed to focus on particular subclasses of relations that can be obtained much faster. The relations considered in the attack are given by linear combinations of relations of the form

$$g(x_0, \ldots, x_{\ell-1}, s_t, \ldots, s_{t+m})$$

where the terms of highest degree do not involve any keystream bits. Then, an additional precomputation step consists in determining the linear combinations of the previous relations which cancel out the highest degree monomials. Some algorithms for this step have been proposed in [22, 2]. This technique helps to decrease the degree of the relations used in the attack for different practical examples. But, here again, we do not have any theoretical result

connecting the algebraic immunity of the function and the existence of such low degree linear combinations.

Criteria related to other attacks. Some other design criteria for filtering and combining functions have appeared with the presentation of new attacks. A well-defined criterion is the *homorphorphicity* of a filtering function. This quantity was defined by Zhang and Zheng [77] for S-boxes but it was not directly related to a known attack. Recently, it has been shown to measure the resistance to some attacks proposed in [50, 58] against filter generators. Indeed, the keystream produced by a filter generator can be distinguished from a random sequence by using the following fact: if we consider w internal state of the LFSR that sum up to zero (these states correspond to multiples with w terms of the feedback polynomial), then the probability that the corresponding keystream bits sum up to zero is strictly greater than 0.5. The success of such a distinguisher is quantified by the w-th order homorphorphicity of the filtering function, which is defined as the probability that $f(x_1) + \cdots + f(x_w) = 0$ when $x_1 + \cdots + x_w = 0$. As shown in [50, 58], this probability can be derived from the sum of w-th powers of the Walsh coefficients of the function. For $w = 4$, which is the optimal value for the attack presented in [58], this quantity is known to be highly related to the sum-of-square indicator and to the nonlinearity of the function [9].

Some other attacks have been proposed on the basic families of LFSR-based generators, but the related resistance criteria still need to be formalized. As an example, the generalized inversion attack [40] clearly involves the properties of some derivatives of the filtering function, but it is not clear whether the complexity of the attack is related to the propagation characteristics of the filtering function.

As mentioned previously, a last class of attacks for which the related security criteria still need to be determined is the class of attacks against filter generators related to the augmented function, such as conditional correlation attacks [51] or fast algebraic attacks.

Intuitive criteria. The designers of filter generators or of combination generators usually consider some additional requirements when they choose a Boolean function, since they want to avoid some particular structures. These requirements sometimes correspond to suitable cryptographic properties, but in another context. For instance, the propagation criterion is obviously a suitable criterion for block ciphers or for hash functions, but it is not imposed by any known attack on stream ciphers. Another example is the lack of linear structure. The existence of linear structures is known to be a weakness for block ciphers. It is not the case for combination generators and for filtering functions, but no concrete design makes use of such a function. Note that it is even recommended in [39] that a filtering function has some linear structure since this guarantees good statistical properties.

3.1.6 More sophisticated functions in LFSR-based ciphers

Many stream ciphers do not use a simple Boolean filtering or combining function; they prefer more sophisticated mappings in order to render the attacks more difficult or in order to increase the throughput of the generator.

Multi-output Boolean functions. A basic technique for increasing the speed of the generator, especially for software dedicated ciphers, consists in using a function with several outputs. Such functions are called vectorial Boolean functions, or *S(ubstitution)-boxes* by analogy with block ciphers. But, as pointed out in [76], the resistance of the generator to fast correlation attacks usually decreases with the number of output bits of the function. For a single output function, the attack exploits the fact that the output may be approximated by an affine function of the input variables. But, for a function S with m outputs, the attacker can apply any Boolean function g of m variables to the output vector $(y_1, \ldots, y_m)$ and he or she can perform the attack on the resulting sequence $z = g(y_1, \ldots, y_m)$. Therefore, the relevant parameter is not the nonlinearity of the vectorial function, which is the lowest Hamming distance between any linear combination of the components of S and the affine functions, but the so-called *unrestricted nonlinearity* [16], which is the lowest distance between any function $g \circ S$ and the affine functions, where g varies in the set of all nonzero Boolean functions of m variables. It is shown in [76] that the function g which satisfies the lowest distance to a given affine function can be deduced from the Walsh coefficients of S. Some lower and upper bounds on the unrestricted nonlinearity of a functions are given in [76, 16]. A family of balanced functions of $2t$ variables with t outputs having a high unrestricted nonlinearity is also exhibited.

For similar reasons, the algebraic immunity of a vectorial function tends to decrease with the number of output bits (see Section 3.2.3).

A particular case of generators based on multi-output Boolean functions are the word-oriented ciphers. In order to increase the performance of software implementations, many ciphers use LFSRs over an extension field $\mathbf{F}_{2^m}$ and the associated filtering function is usually a mapping from $\mathbf{F}_{2^m}^n$ into $\mathbf{F}_{2^m}$. This technique is used for instance in the stream cipher SNOW-v2.0, in the SOBER family and in Turing. The associated filtering function can obviously be seen as a vectorial Boolean function with mn inputs and m outputs. Consequently, all results previously mentioned apply, but the major open issue here is to determine whether word-oriented attacks can be mounted which exploit the particular structure of the function defined as a polynomial over $\mathbf{F}_{2^m}$.

It is important to note that the augmented function associated to a single output function is also a particular vectorial function, with linearly equivalent components.

Functions with memory. In some LFSR-based generators, the combining or filtering function is replaced by a finite automaton with some memory bits. An example is the E_0 keystream generator used in the Bluetooth wireless LAN system, which uses a combining function with 4 inputs and 4 memory bits. However, (fast) correlation attacks [53, 52] and (fast) algebraic attacks [4] can still be applied on such systems. Concerning algebraic attacks, Armknecht and Krause proved that, for any filtering function of n variables with M memory bits, there always exists a relation of degree at most $\lceil \frac{n(M+1)}{2} \rceil$ between $(M+1)$ consecutive output bits and the bits of the initial state, for a given initial assignment of the memory bits. Obviously, relations of lower degree may exist. For instance, the function used in E_0 provides a relation of degree 4 involving 4 consecutive output bits, which leads to an algebraic attack of running-time around 2^{67} [4]. A similar situation occurs for multi-output functions with memory [26].

The main open issue related to the use of such sophisticated functions is to improve the efficiency of the algorithms for computing their cryptographic properties (unrestricted nonlinearity, algebraic immunity, ...), for a large number of input variables. Another related open problem is to find some general constructions which guarantee a high resistance to all these attacks.

3.1.7 Filtering functions for stream ciphers with a nonlinear transition function

The development of algebraic attacks tends to recommend to replace LFSRs by other devices based on a nonlinear transition function.The two main proposals in this direction are based on FCSR [46, 6] and on T-functions [47, 48]. In these generators, the keystream is still obtained by filtering the successive internal states by a (vectorial) Boolean function. But, the requirements on the filtering function may differ very much from the criteria used for LFSR-based ciphers. For instance, it does not seem necessary to use a function with a high algebraic degree or with a high algebraic immunity in that context since the internal state does not depend linearly on the initial state.

However, choosing a very simple function (such as a linear function) may be dangerous. For instance, it was proposed in [48] to output at time t the $n/2$ most significant bits of the n-bit state obtained by iteratively applying a T-function. The fact that the observation of one output word provides half of the current internal state can be exploited and leads to an attack with time-complexity $\mathcal{O}(2^{\frac{n}{4}})$ [5]. It follows that such a generator must use a more complex filtering function (probably with fewer output bits). But, the criteria for designing such a function remain unknown and must be investigated if we want such ciphers to be a good alternative to LFSR-based stream ciphers.

3.2 S-boxes for block ciphers

The development of cryptanalysis in the last twenty years has led to the definition of some design criteria for block ciphers. These criteria correspond to some mathematical properties of the round function which is used in an iterated block cipher. They essentially concern the confusion part of the round function, usually named S(ubstitution)-box. Indeed, the S-boxes are usually the only nonlinear operations performed in a block cipher. Their roles aim at concealing any algebraic structure that may appear in the system. Because most properties involved are invariant under composition with a linear transformation, the properties of the S-boxes used directly provide results on the entire round function for some attacks.

3.2.1 Resistance to differential attacks

Differential cryptanalysis successfully applies when the reduced cipher (i.e., the cipher obtained by removing the final round of the original block cipher) has a derivative which is not uniformly distributed. This means that two inputs with fixed difference lead to outputs whose difference takes a certain value with high probability. Therefore, a necessary security condition is that the output distributions of all derivatives of the S-boxes, $x \mapsto S(x+a) + S(x)$, must be close to the uniform distribution. The relevant parameter for an S-box with n inputs

and m outputs is then

$$\delta_S = \max_{a,b\neq 0} \#\{x \in \mathbf{F}_2^n,\ S(x+a) + S(x) = b\} \ .$$

A general trivial bound is $\delta_S \geq 2^{n-m}$. The functions achieving this bound are called *perfect nonlinear functions* or *bent functions* [57]. Such functions only exist when n is even and $m \leq \frac{n}{2}$ [59]. A major drawback is that these optimal functions are not balanced, and it may be a weakness to use non-invertible S-boxes in a block cipher (see e.g. [64]). When the number of output bits of the S-box is the same as the number of inputs (this is the case in many ciphers), we have an improved bound, $\delta_S \geq 2$, and the functions achieving this second bound are called *almost perfect nonlinear (APN)* [62]. In both cases, i.e., $m \leq \frac{n}{2}$ and $n = m$, the bounds are known to be tight. But, when the number of output bits lie between $\frac{n}{2}$ and n, the lowest possible value that can be achieved for δ_S is still unknown.

Most works on optimal S-boxes with respect to differential attacks focus on the case where $m = n$. Some infinite families on APN S-boxes have been exhibited [38, 60, 45, 32, 31], but all the known optimal functions are equivalent (under composition by an affine mapping) to a power function, $x \mapsto x^s$ over $\mathbf{F}_{2^n}$. A major open problem concerning APN S-boxes is that we do not know any APN permutation with an even number of variables. Actually, it is conjectured that, for any permutation S of $\mathbf{F}_2^n$ with n even, we have $\delta_S \geq 4$. This statement is proved for some particular cases, most notably for power functions [19] and for functions of degree 2 [61, 42]. As an example, the AES S-box satisfy $\delta_S = 4$. The fact that no APN permutation is known for an even number of variables led the designers of MISTY to choose functions of an odd number of variables. Therefore, it would be very important to solve this open problem.

An APN function of 4 variables which is not equivalent to a power function has been found by computer search [49], but some general constructions of such APN mappings would be suitable, especially of APN permutations. Invertible S-boxes of 8 variables with $\delta_S = 4$ which are not equivalent to a power mapping have also been found by perturbation techniques [34]. The use of an S-box which is linearly equivalent to a power mapping may intuitively appear as a weakness because all its outputs are affinely equivalent. However, the relevance of this property remains unclear since no attack based on this structural property has been mounted.

3.2.2 Resistance to linear attacks

The resistance to linear attacks involves the *nonlinearity* of the S-box, which corresponds to the lowest nonlinearity achieved by a linear combination of its components:

$$\mathcal{NL}(S) = 2^{n-1} - \frac{1}{2}\mathcal{L}(S) \ ,$$

where $\mathcal{L}(S)$ denotes the highest magnitude appearing in the Walsh spectrum of all linear combinations of S. For an S-box with n inputs and m outputs, we have $\mathcal{L}(S) \geq 2^{\frac{n}{2}}$. The functions for which equality holds are the *perfect nonlinear functions*, i.e., the functions which also provide an optimal resistance to differential attacks [57, 59]. This bound is not achieved when the number of outputs exceeds $\frac{n}{2}$. For $m = n$, we know that $\mathcal{L}(S) \geq 2^{\frac{n+1}{2}}$, and the functions achieving the bound are called *almost bent (AB)* [69, 18]. Such optimal functions

obviously only exist for an odd number of variables. When n is even, some functions with $\mathcal{L}(S) = 2^{\frac{n}{2}+1}$ are known and it is conjectured that this value is the minimum [68, 33]. Here again, nothing is known when the number of output bits lie between $\frac{n}{2}$ and n.

When the number of inputs and outputs are the same, the quantities which measure the resistance to both attacks are also related. Actually, Chabaud and Vaudenay [18] proved that any AB function is APN. The converse does not hold, and an additional condition on the Walsh coefficients on an APN function is required for ensuring optimal resistance to linear attacks [10].

Almost bent functions are optimal combinatorial objects. Therefore, they appear in several areas of telecommunications: this property is related to metric properties of some linear codes, especially of binary cyclic codes with two zeros. Almost bent power functions also correspond to pairs of maximum-length sequences with preferred cross-correlation (see e.g. [8]).

As previously mentioned, only a few optimal S-boxes regarding differential and linear attacks are known, and all existing constructions lead to functions which present a very strong algebraic structure. An important research direction in the future would be to further investigate the construction of optimal or of sub-optimal functions with respect to both criteria. The implementation complexity of the S-box must be taken into account for some applications, e.g. for hardware dedicated block ciphers. For instance, it would be interesting to determine whether the S-boxes used in the block cipher ICEBERG are the best ones which could be obtained under similar implementation constraints [70].

3.2.3 Resistance to algebraic attacks

The idea of mounting algebraic attacks on block ciphers have been proposed by Courtois and Pieprzyk [28]. The feasibility of these attacks on concrete ciphers is still questionable since the attacker needs to solve a system of algebraic equations with a huge number of variables. However, it seems suitable to use S-boxes which have a high algebraic immunity. Algebraic attacks on block ciphers rely on the same principle as the attacks on stream ciphers. They exploit the existence of low-degree relations between the inputs and the outputs of the S-box, i.e., for an S-box S with n inputs and m outputs, the existence of a function g of low degree d such that

$$g(x_1, \ldots, x_n, S_1(x_1, \ldots, x_n), \ldots, S_m(x_1, \ldots, x_n)) = 0 \ ,$$

where $S_1, \ldots, S_m$ denote the Boolean components of the S-box. As noted in [3], finding such a relation of degree d is equivalent to finding an annihilator of degree d for the characteristic function Φ_S of S, which is the Boolean function of $(n+m)$ variables defined by

$$\Phi_S(x_1, \ldots, x_n, y_1, \ldots, y_m) = 1 \text{ if and only if } y_i = S_i(x_1, \ldots, x_n), \ \forall i \ .$$

Therefore, all previously mentioned results on the algebraic immunity of Boolean functions cane be applied here. For instance, an S-box with n inputs and m outputs has relations of degree at most d if

$$\sum_{i=0}^{d} \binom{n+m}{i} > 2^n \ .$$

Thus, it appears that low degree relations exist when the number of variables of the S-box is small, which is usually the case because of implementation constraints. For instance, any

S-box with 8 inputs and 8 outputs, such as the AES S-box, has relations of degree 3. It is well-known that the AES S-box, which is defined by the inverse function over the field with 2^8 elements (with $S(0) = 0$) has 39 quadratic relations. Some lower bounds on the general number of quadratic relations induced by some other classes of almost perfect nonlinear power functions of n variables have been computed in [20, 27]. An important research direction is then to find S-boxes with optimal algebraic immunity which oppose a high resistance to differential and linear attacks. The question of the existence of a potential trade-off between the algebraic immunity of an S-box and the criteria related to other classical attacks is still open.

For algebraic attacks on block ciphers, the outputs of the S-box are usually unknown since they are intermediate variables corresponding to the output bits obtained after each internal round. Therefore, the degree of the relations in these variables must be low. But, in some other contexts, the outputs of the S-box are known. We can then allow any degree in the output variables and only a low degree in the input variables. This situation occurs for instance when the S-box is used as a filtering or combining function in an LFSR-based stream cipher. In this case, for an S-box with n inputs and m outputs, there exists a relation of degree at most d in the input variables (and of any degree in the output variables) if

$$\sum_{i=0}^{d} \binom{n}{i} > 2^{n-m} \ .$$

In other applications, the suitable degrees in the input and output variables may differ. For instance, in the context of fast algebraic attacks on stream ciphers, we are interested in finding algebraic relations induced by the augmented function such that the terms of highest degree do not depend on the output variables. For mounting a bilinear attack on a Feistel cipher [25], we are searching for quadratic relations whose quadratic monomials are of the form $x_i y_j$ only. For all these cases, similar bounds can be derived by considering that the algebraic relations correspond to the rows of a given matrix which add up to zero. Each row of the matrix corresponds to the evaluations on all possible inputs of a suitable monomial in the x_i's and y_j's. A trivial bound is directly obtained by noticing that there always exists a subset of rows which sum up to zero if the number of rows exceed the number of columns. However, it would be interesting to go further in this analysis. For instance, we need to determine whether the existence of algebraic relations having these particular forms is related to other classical criteria.

3.2.4 Resistance to other attacks involving the S-boxes

Other classical attacks on iterated block ciphers have introduced further requirements on the inherent S-boxes.

Higher order differential attacks For instance, higher order differential attacks introduced by Knudsen are generalizations of differential attacks in the sense that they rely on some properties of the higher order derivatives of the S-boxes. More precisely, this kind of attack can be mounted if the reduced cipher has a k-th order derivative which is constant, for a small value of k (since the attack requires the knowledge of 2^k pairs of plaintexts / ciphertexts). A

natural candidate arises when the reduced cipher, seen as a multivariate polynomial, has a low degree. Indeed, any $(d+1)$-th derivative of a function of degree d vanishes. The degree of the S-box then provides a trivial upper bound on the complexity of higher order differential cryptanalysis. This bound was directly used by Jakobsen and Knudsen [44] for breaking a cipher example, whose round function is an almost bent permutation of degree 2. However, constant higher-order derivatives can sometimes be found even if the round function has a high degree. For instance, it was shown in [13] that the degree of 2 iterations of the round function $F \circ F$ grows much slower than $\deg(F)^2$ when all Walsh coefficients of F are divisible by a high power of 2. It is important to note that, by definition, the Walsh coefficients of an almost bent S-box of n variables are divisible by $2^{\frac{n+1}{2}}$. Therefore, the use of almost bent S-boxes may make the cipher vulnerable to other attacks.

Interpolation attack. As shown in [44], the S-boxes used in a block cipher must also have a high degree when they are seen as a univariate polynomial in $\mathbf{F}_{2^n}[X]$. Otherwise, the whole cipher can be expressed as a univariate polynomial with low degree, whose coefficients can be found by interpolation from the knowledge of some pairs of plaintexts / ciphertexts. A more powerful variant of the interpolation attack has been presented by Jakobsen [43]. This variant applies when the round function is close to a low-degree univariate polynomial. The relevant cryptographic parameter here is the distance of the round function to the set of all univariate polynomials over $\mathbf{F}_{2^n}$ of degree at most d, i.e., the distance to the Reed-Solomon code of order d (see also [23] for a generalization of this attack).

Other criteria. As for Boolean functions, there exist some structural properties for S-boxes which are not related to any attack, but which are believed to introduce weaknesses. The most interesting one is probably the use of S-boxes which correspond to power functions (up to an affine transformation). Such S-boxes are commonly used because all known optimal functions with respect to differential and linear attacks lie in this class. But, the intuition may be that such a strong algebraic property introduces a weakness which could be exploited by another attack. However, it is difficult to determine which particular characteristics of power mappings may induce a weakness. The fact that all outputs of such a function are affinely equivalent has been pointed out in [37]. It is even recommended in [41] that an S-box must lie as far as possible from the set of all power mappings. Therefore, it seems essential to try to mount an attack based on these structural properties in order to determine whether they must be avoided or not.

3.3 Future directions

Many important open questions arise when a designer needs to choose suitable Boolean functions or S-boxes for a stream cipher or for a block cipher. Besides all previously mentioned research directions, it appears to be important to provide tools to help the designer. For this purpose, the ECRYPT working group on *Open research areas in symmetric cryptography* plans to build a database containing the best known Boolean functions and S-boxes of different sizes for the known design criteria.

References

[1] R. J. Anderson. Searching for the optimum correlation attack. In *Fast Software Encryption - FSE'94*, volume 1008 of *Lecture Notes in Computer Science*, pages 137–143. Springer-Verlag, 1995.

[2] F. Armknecht. Improving fast algebraic attacks. In *Fast Software Encryption - FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 65–82. Springer-Verlag, 2004.

[3] F. Armknecht. On the existence of low-degree equations for algebraic attacks. In *ECRYPT Network of Excellence - SASC Workshop Record*, pages 175–189, 2004. Available via `www.isg.rhul.ac.uk/research/projects/ecrypt/stvl/sasc.html`.

[4] F. Armknecht and M. Krause. Algebraic attacks on combiners with memory. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 162–176. Springer-Verlag, 2003.

[5] V. Benony, F. Recher, E. Wegrzynowski, and C. Fontaine. An improved method to retrieve internal state of Klimov-Shamir pseudo-random sequence generators. In *Proc. of SETA04*. Springer-Verlag, 2004.

[6] T. Berger and F. Arnault. Design of new pseudo-random generators based on filtered FCSR automaton. In *ECRYPT Network of Excellence - SASC Workshop Record*, pages 109–120, 2004. Available via `www.isg.rhul.ac.uk/research/projects/ecrypt/stvl/sasc.html`.

[7] A. Braeken, Y. Borissov, S. Nikova, and B. Preneel. Classification of Boolean functions of 6 variables or less with respect to cryptographic properties. Technical report, IACR Preprint, 2004. Available at `http://eprint.iacr.org/2004/248/`.

[8] A. Canteaut. Cryptographic functions and design criteria for block ciphers. In *Progress in Cryptology - INDOCRYPT 2001*, volume 2247 of *Lecture Notes in Computer Science*, pages 1–16. Springer-Verlag, 2001.

[9] A. Canteaut, C. Carlet, P. Charpin, and C. Fontaine. Propagation characteristics and correlation-immunity of highly nonlinear Boolean functions. In *Advances in Cryptology - EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 507–522. Springer-Verlag, 2000.

[10] A. Canteaut, P. Charpin, and H. Dobbertin. A new characterization of almost bent functions. In *Fast Software Encryption - FSE'99*, volume 1636 of *Lecture Notes in Computer Science*, pages 186–200. Springer-Verlag, 1999.

[11] A. Canteaut, M. Daum, H. Dobbertin, and G. Leander. Normal and non normal bent functions. In *Proceedings of the International Workshop on Coding and Cryptography - WCC 2003*, pages 91–100, Versailles, France, March 2003.

[12] A. Canteaut and M. Trabbia. Improved fast correlation attacks using parity-check equations of weight 4 and 5. In *Advances in Cryptology - EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 573–588. Springer-Verlag, 2000.

[13] A. Canteaut and M. Videau. Degree of composition of highly nonlinear functions and applications to higher order differential cryptanalysis. In *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*. Springer-Verlag, 2002.

[14] C. Carlet. On the coset weight divisibility and nonlinearity of resilient and correlation immune functions. In *Sequences and Their Applications - SETA 2001*, Discrete Mathematics and Theoretical Computer Science, pages 131–144. Springer-Verlag, 2001.

[15] C. Carlet. Improving the algebraic immunity of resilient and nonlinear functions and constructing bent functions. Technical report, IACR Preprint, 2004. Available at `http://eprint.iacr.org/2004/276/`.

[16] C. Carlet and E. Prouff. On a new notion of nonlinearity relevant to multi-output pseudo-random generators. In *Selected Areas in Cryptography - SAC 2003*, volume 3006 of *Lecture Notes in Computer Science*, pages 291–305. Springer-Verlag, 2004.

[17] C. Carlet and P. Sarkar. Spectral domain analysis of correlation immune and resilient boolean functions. *Finite fields and Applications*, (8):120–130, 2002.

[18] F. Chabaud and S. Vaudenay. Links between differential and linear cryptanalysis. In *Advances in Cryptology - EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 356–365. Springer-Verlag, 1995.

[19] P. Charpin, A. Tietäväinen, and V. Zinoviev. On binary cyclic codes with minimum distance $d = 3$. *Problems of Information Transmission*, 33(4):287–296, 1997.

[20] J.H. Cheon and D.H. Lee. Resistance of S-boxes against algebraic attacks. In *Fast Software Encryption - FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 83–94. Springer-Verlag, 2004.

[21] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic programming. *Journal of Symbolic Computation*, (9):251–280, 1990.

[22] N. Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 176–194. Springer-Verlag, 2003.

[23] N. Courtois. The inverse S-box, non-linear polynomial relations and cryptanalysis of block ciphers. In *AES 4 Conference*, volume 3373 of *Lecture Notes in Computer Science*. Springer-Verlag, 2005. To appear.

[24] N. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 345–359. Springer-Verlag, 2003.

[25] N.T. Courtois. Feistel schemes and bi-linear cryptanalysis. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 23–40. Springer-Verlag, 2004.

[26] N.T. Courtois. Algebraic attacks on combiners with memory and several outputs. In *ICISC 2004*, Lecture Notes in Computer Science. Springer-Verlag, 2005. Available from `http://eprint.iacr.org/2003/125/`.

[27] N.T. Courtois, B. Debraize, and E. Garrido. On exact algebraic immunity of s-boxes based on power functions. Preprint, 2004.

[28] N.T. Courtois and J. Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *Advances in Cryptology - ASIACRYPT 2002*, volume 2502 of *Lecture Notes in Computer Science*, pages 267–287. Springer-Verlag, 2002.

[29] D. K. Dalai, K. C. Gupta, and S. Maitra. Results on algebraic immunity for cryptographically significant boolean functions. In *Progress in Cryptology - Indocrypt 2004*, volume 1880 of *Lecture Notes in Computer Science*, pages 92–106. Springer-Verlag, 2004.

[30] H. Dobbertin. Construction of bent functions and balanced Boolean functions with high nonlinearity. In *Fast Software Encryption*, volume 1008 of *Lecture Notes in Computer Science*, pages 61–74, Berlin, Germany, 1994. Springer-Verlag.

[31] H. Dobbertin. Almost perfect nonlinear power functions on $GF(2^n)$: the Niho case. *Information and Computation*, 1998. To appear.

[32] H. Dobbertin. Almost perfect nonlinear power functions on $GF(2^n)$: the Welch case. *IEEE Transactions on Information Theory*, 1998. To appear.

[33] H. Dobbertin. One-to-one highly nonlinear functions on finite field with characteristic 2. *Applicable Algebra in Engineering, Communication and Computing*, 9:139–152, 1998.

[34] F. Epron and L. Granboulan. How to generate a family of key-dependent bijective S-boxes such that there won't be weak keys with respect to linear and differential cryptanalysis. In *Proceedings of YACC'04*, Porquerolles, France, June 2004.

[35] J.-C. Faugère and G. Ars. An algebraic cryptanalysis of nonlinear filter generators using Gröbner bases. Technical Report 4739, INRIA, 2003. Available at `ftp://ftp.inria.fr/INRIA/publication/publi-pdf/RR/RR-4739.pdf`.

[36] E. Filiol and C. Fontaine. Highly nonlinear balanced Boolean functions with a good correlation-immunity. In *Advances in Cryptology - EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 475–488. Springer-Verlag, 1998.

[37] J. Fuller and W. Millan. Linear redundancy in S-boxes. In *Fast Software Encryption - FSE 2003*, volume 2887 of *Lecture Notes in Computer Science*, pages 74–86. Springer-Verlag, 2003.

[38] R. Gold. Maximal recursive sequences with 3-valued recursive crosscorrelation functions. *IEEE Transactions on Information Theory*, 14:154–156, 1968.

[39] J. Dj. Golić. On the security of nonlinear filter generators. In *Fast Software Encryption - FSE'96*, volume 1039 of *Lecture Notes in Computer Science*, pages 173–188. Springer-Verlag, 1996.

[40] J.Dj Golic, A. Clark, and E. Dawson. Generalized inversion attack on nonlinear filter generators. *IEEE Transactions on Computers*, 49(10):1100–1109, 2000.

[41] G. Gong and S.W. Golomb. Transform domain analysis of DES. *IEEE Transactions on Information Theory*, 45(6):2065–2073, 1999.

[42] X.D. Hou. Affinity of permutations of $\mathbf{F}_2^n$. In *Workshop on Coding and Cryptography - WCC 2003*, pages 273–280, 2003.

[43] T. Jakobsen. Cryptanalysis of block ciphers with probabilistic non-linear relations of low degree. In *Advances in Cryptology - CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 212–222. Springer-Verlag, 1998.

[44] T. Jakobsen and L.R. Knudsen. The interpolation attack on block ciphers. In *Fast Software Encryption 97*, volume 1267 of *Lecture Notes in Computer Science*. Springer-Verlag, 1997.

[45] T. Kasami. The weight enumerators for several classes of subcodes of the second order binary Reed-Muller codes. *Information and Control*, 18:369–394, 1971.

[46] A. Klapper and M. Goresky. Feedback shift registers, 2-adic span and combiners with memory. *Journal of Cryptology*, 10(2), 1997.

[47] A. Klimov and A. Shamir. A new class of invertible mappings. In *CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 470–483. Springer-Verlag, 2002.

[48] A. Klimov and A. Shamir. Cryptographic applications of t-functions. In *Selected Areas in Cryptography - SAC 2003*, volume 3006 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.

[49] L. Knudsen. On almost perfect nonlinear functions and permutations. Preprint, 1999.

[50] S. Leveiller, G. Zémor, P. Guillot, and J. Boutros A2. A new cryptanalytic attack for PN-generators filtered by a Boolean function. In *Selected Areas in Cryptography - SAC 2002*, volume 2595 of *Lecture Notes in Computer Science*, pages 232–249. Springer-Verlag, 2002.

[51] B. Löhlein. Attacks based on conditional correlations against the nonlinear filter generator. Technical report, IACR Preprint, 2003. Available at `http://eprint.iacr.org/2003/020/`.

[52] Y. Lu and S. Vaudenay. Cryptanalysis of bluetooth keystream generator two-level E0. In *Advances in Cryptology - ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 483–499. Springer-Verlag, 2004.

[53] Y. Lu and S. Vaudenay. Faster correlation attack on Bluetooth keystream generator E0. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 407–425. Springer-Verlag, 2004.

[54] S. Maitra and E. Pasalic. Further constructions of resilient boolean functions with very high nonlinearity. *IEEE Transactions on Information Theory*, 48(7):1825 – 1834, 2002.

[55] S. Maitra and P. Sarkar. Maximum nonlinearity of symmetric Boolean functions on odd number of variables. *IEEE Transactions on Information Theory*, 48(9), 2002.

[56] W. Meier, E. Pasalic, and C. Carlet. Algebraic attacks and decomposition of Boolean functions. In *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 474–491. Springer-Verlag, 2004.

[57] W. Meier and O. Staffelbach. Nonlinearity criteria for cryptographic functions. In *Advances in Cryptology - EUROCRYPT'89*, volume 434 of *Lecture Notes in Computer Science*, pages 549–562. Springer-Verlag, 1990.

[58] H. Molland and T. Helleseth. An improved correlation attack against irregular clocked and filtered generator. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 373–389. Springer-Verlag, 2004.

[59] K. Nyberg. Perfect nonlinear S-boxes. In *Advances in Cryptology - EUROCRYPT'91*, volume 547 of *Lecture Notes in Computer Science*, pages 378–385. Springer-Verlag, 1991.

[60] K. Nyberg. Differentially uniform mappings for cryptography. In *Advances in Cryptology - EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 55–64. Springer-Verlag, 1993.

[61] K. Nyberg. S-boxes and round functions with controllable linearity and differential uniformity. In *Fast Software Encryption - FSE'94*, volume 1008 of *Lecture Notes in Computer Science*. Springer-Verlag, 1995.

[62] K. Nyberg and L.R. Knudsen. Provable security against differential cryptanalysis. In *Advances in Cryptology - CRYPTO'92*, volume 740 of *Lecture Notes in Computer Science*, pages 566–574. Springer-Verlag, 1993.

[63] E. Pasalic, S. Maitra, T. Johansson, and P. Sarkar. New constructions of resilient and correlation immune Boolean functions achieving upper bound on nonlinearity. In *Workshop on Coding and Cryptography 2001 - WCC 2001*, pages 425–434, 2001.

[64] V. Rijmen, B. Preneel, and E. De Win. On weaknesses of non-surjective round functions. *Designs, Codes and Cryptography*, 12(3):253–266, 1997.

[65] P. Sarkar and S. Maitra. Construction of nonlinear Boolean functions with important cryptographic properties. In *Advances in Cryptology - EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 485–506. Springer-Verlag, 2000.

[66] P. Sarkar and S. Maitra. Nonlinearity bounds and constructions of resilient boolean functions. In *Advances in Cryptology - CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 515–532. Springer-Verlag, 2000.

[67] P. Sarkar and S. Maitra. Construction of nonlinear resilient Boolean functions using "small" affine functions. *IEEE Transactions on Information Theory*, 50(9):2185 – 2193, 2004.

[68] D.V. Sarwate and M.B. Pursley. Crosscorrelation properties of pseudorandom and related sequences. *Proceedings of the IEEE*, 68(5):593–619, 1980.

[69] V.M. Sidelnikov. On mutual correlation of sequences. *Soviet Math. Dokl.*, 12:197–201, 1971.

[70] F.-X. Standaert, G. Piret, G. Rouvroy, J.-J. Quisquater, and J.-D. Legat. ICEBERG: an involutional cipher efficient for block encryption in reconfigurable hardware. In *Fast Software Encryption - FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 279–298. Springer-Verlag, 2004.

[71] P. Stanica, S. Maitra, and J. Clark. Results on rotation symmetric bent and correlation immune Boolean functions. In *Fast Software Enncryption - FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.

[72] Y. Tarannikov. New constructions of resilient Boolean functions with maximal nonlinearity. In *Fast Software Encryption - FSE 2001*, volume 2355 of *Lecture Notes in Computer Science*, pages 66–77. Springer-Verlag, 2001.

[73] M. Videau. On some properties of symmetric Boolean functions. In *Proceedings 2004 IEEE International Symposium on Information Theory*, page 500. IEEE Press, 2004.

[74] I. Wegener. *The complexity of Boolean functions*. Wiley, 1987.

[75] M. Zhang. Maximum correlation analysis of nonlinear combining functions in stream ciphers. *Journal of Cryptology*, 13(3):301–313, 2000.

[76] M. Zhang and A. Chan. Maximum correlation analysis of nonlinear S-boxes in stream ciphers. In *Advances in Cryptology - CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 501–514. Springer-Verlag, 2000.

[77] X.-M. Zhang and Y. Zheng. The nonhomomorphicity of Boolean functions. In *Selected Areas in Cryptography - SAC'98*, volume 1553 of *Lecture Notes in Computer Science*, pages 280–295. Springer-Verlag, 1999.

4 Provable security in symmetric cryptography

This section briefly outlines available provable security results and proof techniques applicable to symmetric algorithms and perspectives for this important research topic.

The notion of provable security is to be understood here in a broad sense. Like in asymmetric cryptography, reduction techniques allow to relate in a reasonably general adversarial model the security of an encryption or message authentication algorithm to the conjectured intractability of a well-defined mathematical problem. However, unlike in asymmetric cryptography, the security proofs in symmetric cryptography usually don't reduce to a mathematical problem from number theory, but often reduce to a cryptographic assumption on some underlying component, or are based on a hypothesis regarding the category of attacks that the adversary is allowed to use. Until now nearly all known examples of reductions to number theoretical assumptions are of theoretical rather than practical interest for symmetric algorithms (e.g. stream ciphers), and it is a major open issue whether there exist practical symmetric algorithms presenting implementation complexity and performance characteristics close to those for say AES for which global reduction theorems can be proved.

Therefore, we will also consider proofs of more partial aspects of security of a symmetric algorithm, i.e. which only validate the resistance of a cipher to a specific category of attacks (e.g. linear or differential cryptanalysis), or only validate one level of a cryptographic construction (e.g. a block cipher structure, a mode of operation) under the assumption that the underlying level of the construction is itself secure as done in the so-called Luby-Rackoff security model. However, proofs that a cryptographic component of a cipher, e.g. an S-box, a boolean functions involved in a stream-cipher, a linear permutation involved in an SP network, etc., satisfies some cryptographic criteria (low associated differential and linear probabilities, high nonlinearity, MDS property, etc.) will be considered outside the scope of this section.

One of the most impressive achievements in cryptology in the 20 last years is the proof of a chain of reduction theorems relating the main categories of primitives encountered in symmetric cryptology. This chain starts from the conjectured existence of one way functions (OWF) or alternatively from the conjectured existence of one way permutations (OWP) [1] and successively relates by reduction theorems; i.e. by constructing at each step a new object whose insecurity would imply the insecurity of the previous object.

1. **OWF**s (or **OWP**s) to Pseudo-Random Number Generators (**PRNG**s) which represent the keystream generation part of stream ciphers [2, 6, 9].
2. **PRNG**s to Pseudo-Random Function Generators (**PRF**s), using the so-called binary tree construction [5].

[1] A one way function (OWF) can be informally defined as an easy to compute but hard to invert function. In the sequel we will only consider strongly one way functions F, i.e. functions such that there exists no inversion algorithm allowing, when given an F output value y corresponding to an uniformly drawn input value x, to return a preimage x' of y (equal or not to x) with a probability greater than a non negligible amount ϵ (e.g. greater than the inverse of a polynomial function of the input size in the polynomial security model). The existence of OWFs represents a basic assumption in cryptography since computational security based cryptology would collapse otherwise. Numerous candidate one way functions have been identified and are used as building blocks in numerous cryptographic designs.

3. **PRF**s to Pseudo-Random Permutation Generators (**PRP**s) which can be seen as an idealisation of blockciphers or functions associated with modes of operation of blockciphers [10].

Applications of the above reduction theorems are not restricted to the area of cryptologic foundations, but also include proofs of security for fully instantiated algorithms: some provable security results related to (1) are applicable to the construction of stream ciphers whose security is equivalent to the conjecture that a candidate one-way function is indeed one-way, and some provable security results related to (3) allow to validate the security of block cipher construction methods (e.g. the use of the Feistel construction) or to validate modes of operation in the Luby-Rackoff security model.

Examples of other less closely related, but quite important achievements in provable security for symmetric cryptography include the following.

- Wegman and Carter's message authentication paradigm, which is based upon the concept of universal hash function families and allows to construct a provably secure message authentication scheme indexed by a short secret key provided that a secure stream cipher is available.
- Proofs of resistance of some block ciphers against differential or linear attacks exploiting the existence of high probability differentials or linear hulls, by K. Nyberg and L. Knudsen, and developed later on by K. Aoki et al. This proof technique was in particular applied to validate the resistance of the MISTY blockcipher designed by Matsui and its variant KASUMI (used in UMTS) to differential cryptanalysis.
- Vaudenay's decorrelation theory, which can be considered both as a tool to validate cryptographic constructions in the Luby-Rackoff model, and has thus strong connection with (3) above, and as a tool for proving the resistance of some block ciphers against statistical attacks of order 2 such as differential or linear attacks, or more generally against statistical attacks of order d. This proof technique was in particular applied to validate the resistance of the AES candidate DFC block cipher against differential and linear cryptanalysis.
- Merkle and Damgard's proofs that an iterated n-bit hash function based on successive invocations of an $(n+m)$-bit to n-bit compression function is collision free if the underlying compression function is collision free, and their transposition to the proof that the MAC function associated with the HMACconstruction [1] (which allows to convert an iterated hash function to a MAC) is secure if the hash function is collision-free and the underlying compression function behaves as a strong (unforgeable) fixed length MAC of its m-bit input when keyed by its n-bit input.

In the rest of this section, we consider some open issues arising from these different research directions.

4.1 Stream ciphers and pseudo-random generators

In this section we only consider *synchronous stream ciphers*, and not *self-synchronised stream ciphers*, where the keystream generation depends on the ciphertext.

Self-synchronised stream ciphers were used in contexts where data loss was less annoying that latency. Therefore the encrypted message was sent in a long stream, and it was important to be able to resynchronise the decryption even if part of the encrypted stream was lost. Self-synchronised stream ciphers are no longer used today, at least in industry. Instead of sending an encrypted message in a long stream, messages are now split into a number of packets that are acknowledged by the receiver, and if some packet is lost it is resent.

For performance reasons, stream ciphers are often used for encryption of packets, but block ciphers are also suited to this application. This explains why A5/1 (a stream cipher) has been superceded by A5/3 (a block cipher) for cell phone encryption. Stream ciphers used in the encryption of packets have three inputs: the message, the secret key, and the packet number. They operate in two separate steps: first, the secret key and the packet number are used to generate the keystream sequence. Then, the keystream sequence obtained is bitwise combined with the plaintext by a XOR and the result is the ciphertext. Most studies of provable security for stream ciphers neglect the packet number.

Pseudo-random stream generation. Stream ciphers encrypt data in two separate steps. The first step does not involve the plaintext data: the n-bit secret key $\mathbf{k}$ (which is assumed to be uniformly drawn from $\{0,1\}^n$) is expanded into an m-bit keystream sequence $\mathbf{z} = G(\mathbf{k}) = (z_i)_{i=1\cdots m}$. In the second step, the obtained keystream sequence is bitwise combined with the binary sequence $(x_i)_{i=1\cdots m}$ representing the plaintext to provide the ciphertext sequence $(y_i = x_i \oplus z_i)_{i=1\cdots m}$ [2]. Since encryption is involutive, the decryption algorithm operates in the same way as the encryption algorithm.

Stream ciphers represent a generalization of the well known one-time pad encryption method, which can be described as the particular case where the first encryption step is the identity function, i.e. where the keystream sequence is a perfect random sequence. It is well known (and easy to prove) that the one time pad provides unconditional security: the ciphertext provides strictly no information about the plaintext, even to an adversary with unbounded computing resources. The price to pay for this provable unconditional security, namely the fact that the length of the clear text should not exceed the key length [3], makes it impractical for most applications.

From now on, we will only consider stream ciphers which are required to still provide computational security beyond the unicity distance where unconditional security is compromised. For that purpose, one requires that the keystream generation function G be a strong pseudo-random number generator (PRNG), allowing to expand a short n-bit seed uniformly drawn from $\{0,1\}^n$ (the key $\mathbf{k}$) into a longer m-bit ($m > n$) key stream sequence $\mathbf{z} = G(\mathbf{k}) \in \{0,1\}^m$ in such a way that the probability distribution of $\mathbf{z} = G(\mathbf{k})$ be computationally indistinguishable with a probability better than a negligible bound ϵ (by means of a computationally

[2] These definitions can be trivially generalized to situations where the key symbols and the plaintext, keystream and ciphertext sequences are chosen from another finite alphabet than $\{0,1\}$ and where the binary addition mod 2 is replaced by any group law over this alphabet

feasible testing algorithm) from the uniform distribution U_m over $\{0,1\}^m$. It is easy to see that if this condition is satisfied, then the stream cipher associated with the PRNG G is secure, i.e. the probability for an adversary to guess any a priori unknown information about one encrypted plaintext (resp. about the plaintexts of q encryptions with independent keys) is bounded over by ϵ (resp $q \cdot \epsilon$).

The full formalization of the above definitions has so far mainly been done in an asymptotic uniform security model, i.e. one considers families of objects (e.g. generators) indexed by a security parameter (say the key size) and increasing input and output sizes polynomial in the security parameter instead of one single object with fixed input and output size, computing resources are modelled as polynomial probabilistic Turing machines and a probability is considered as negligible if it is lower than the inverse of any monomial function of the security parameter for sufficiently large values of the security parameter. This has turned out to be useful for establishing complexity-theoretic results about the existence (under some conditions) of cryptographic objects like PRNGs, but the validity of this approach is more questionable when it comes to analyzing the security of constructions used in actual ciphers, with fixed parameter values. Fortunately a large fraction of the known provable security results can be formulated both in an asymptotic and a non-asymptotic model.

Proofs of security established so far allow to construct pseudorandom generators whose security can be reduced to the one-wayness of an underlying function involved in the construction: typically one shows that if the underlying function is one way, then the resulting pseudorandom generator is secure, i.e. its output distribution is computationally indistinguishable [3] from the uniform law U_m). The concept of a *hardcore predicate* (or *hardcore bit*) for a one-way function emerged as a key tool for such constructions. A hardcore bit $b(\mathbf{x})$ for a function F can be informally defined as a computable boolean function of $\{0,1\}^n$ such that for a randomly chosen F input value $\mathbf{x}$, $b(\mathbf{x})$ be computationally difficult to predict based on the mere knowledge of $\mathbf{y} = F(\mathbf{x})$ with a success probability better than $1/2 + \epsilon$ for non negligible ϵ. Security proofs for PRNGs were the focus of an intensive research activity that was initiated in the early 80's by seminal papers by Shamir [12], Blum and Micali [2], and Yao [13].

This research developed primarily in two strongly interacting directions, namely generic and specific PRNG constructions and associated reduction theorems.

Generic, and specific, reduction theorems. These provide constructions involving potentially any one-way permutation, or any one-way function. Only some of these constructions are efficient and applicable to the design of practical stream ciphers.

A useful result when it comes to characterizing secure PRNGs is the proof by Yao in [13] that a PRNG G is secure (i.e. its output distribution is computationally indistinguishable from the uniform distribution) iff its output passes the next bit test, i.e. if for any bit position $i \in \{1, \cdots m\}$ no computable test allows to predict the output bit z_i of a random generator

[3] Two probability distributions $D1$ and $D2$ over $\{0,1\}^m$ are said to be computationally indistinguishable if given any computable testing algorithm T which on input value $\mathbf{z} \in \{0,1\}^m$ produces an output value $0 =$ reject or $1 =$ accept, the acceptance probabilities of T for random input values $\mathbf{z}$ respectively drawn according to the probability distributions $D1$ and $D2$ differ in absolute value by a distinguishing probability bounded over by a negligible amount ϵ

output $\mathbf{z} = G(\mathbf{x})$ based on the mere knowledge of the prefix bits $z_j, j < i$ of $G(\mathbf{x})$ (if any).

An important tool for generic PRNG constructions is represented by results on the existence of hardcore bits for one-way functions. It has been established by Yao [13] that if there exists a one-way function, then this function can be used to construct a modified function which is still one-way and for which there exists a hardcore predicate. A few years later, Goldreich and Levin established [6] the more simple and practical result that if F is any n-bit to m-bit one-way function, then the $2n$-bit boolean function $b(\mathbf{x}\|\mathbf{r}) = \mathbf{x} \cdot \mathbf{r}$ (where $\mathbf{x} \cdot \mathbf{r}$ represents the scalar product of $\mathbf{x}$ and a random value $\mathbf{r}$ of $\{0,1\}^n$) is a hardcore bit for the $2n$-bit to $m+n$-bit function $F' : \mathbf{x}\|\mathbf{r} \mapsto F(\mathbf{x})\|\mathbf{r}$, and F' is still one way.

A general methodology for constructing an n-bit to m-bit PRNG is the iterated invocation of an n-bit to $n+1$-bit function G which has the property that the output distribution of G is indistinguishable from the uniform distribution over $\{0,1\}^{n+1}$. Such a function G is easy to construct if there exists an n-bit to n-bit one-way function F such that (1) there exists a hardcore bit $b(\mathbf{x})$ for F and (2) the output distribution of F is computationally indistinguishable from the uniform distribution U_n: the function G defined by $\mathbf{x} \mapsto \mathbf{x}||b(\mathbf{x})$ satisfies conditions (1) and (2).

A direct application of the above methodology is the reduction theorem allowing to construct a secure PRNG based upon any one-way permutation (OWP). A technically more difficult theorem (whose proof is based upon a much less practical construction) states that a secure PRNG can be constructed based upon any one way function (OWF). This theorem was established by Impagliazzo, Levin, Luby and Hastad [9]. It represents a major theoretical result but its practical applicability to stream cipher designs is unclear.

Looking at the instantiation of this work, we turn our attention to using specific candidate one-way functions.

Most of the known provable stream cipher constructions proposed so far use the hardcore predicate results, i.e. they iterate a candidate one-way function and output a hardcore bit at each iteration. The main candidate one-way functions that are used are the Discrete Logarithm function, the Quadratic Residuosity and the RSA function. Some of the proposed constructions are:

- The Blum-Micali construction, which uses the fact that under the strong Discrete Logarithm assumption the most significant bit of $\mathbf{x}$ is a hardcore predicate for the exponential function.
- The Blum-Blum-Shub construction which uses the hardness of the parity of number x_i with $x_{i+1} = x_i^2 \mod n$ where n is a Blum number. The reduction is done under the Quadratic Residuosity assumption.
- The Alexi-Chor-Goldreich-Schnorr construction which is based on the RSA assumption.

However, even with some improvements (e.g. outputting up to $log(n)$ bits instead of 1), all of these constructions are very slow, because the number of bits output for a single iteration is very small, and every iteration has high complexity ($O(n^2)$ or more). Some more practical constructions have been proposed and in these two constructions, more bits are output for each single iteration.

- The Syndrome Decoding Algorithm [4], which uses the NP-hardness of the Syndrome Decoding Problem.
- BMGL [8], submitted to the NESSIE project, built around the Rijndael block cipher.

Some open areas of interest include the following.

- The transposition of known results to a non-asymptotic security model.
- Finding more constructive and tighter reductions (as is done in asymmetric cryptography).
- Identifying suitable intractable problems which might lead to more practical provable PRNGs.
- Security proofs allowing to output more than $log(n)$ bits at each iteration (as is done in BMGL).
- Extending available security proofs for PRNGs to stream ciphers with an additional input parameter, namely an initialization value (IV).

4.2 Partial validation in the Luby-Rackoff security model

A key dependent cryptographic function such as a block cipher or a mode of operation of a block cipher can be viewed as a random function or permutation generator allowing one to derive an n-bit to m-bit function or permutation from a randomly selected key value. It is generally defined using a recursive construction process. Each step of the recursion consists of deriving a random function (or permutation) F from r previously defined random functions (or permutations) $f_1, \cdots, f_r$, and can be represented by a relation of the form $F = \Phi(f_1, \cdots, f_r)$.

One of the strongest security requirements one can put on such a random function or permutation generator F is that F be impossible to distinguish with a non-negligible success probability from a perfect random function or permutation F^* uniformly drawn from the set of all functions (or permutations) with the same input and output sizes, even if a probabilistic testing algorithm A of unlimited power is used for that purpose and if the number q of adaptively chosen queries of A to the random instance of F or F^* to be tested is large.

It is generally not possible to prove indistiguishability properties for "real life" cryptologic random functions and large numbers of queries, because this would require a key length that is far too long. However, it is often possible to prove or disprove that if a random function F encountered at a given level of a cryptologic function construction is related to random functions encountered at the lower recursion level by a relation of the form $f = \Phi(f_1, \cdots, f_r)$, then if we replace the actual f_1 to f_r random functions of the cipher by independent perfect random functions or permutations f_1^* to f_r^* (or, in a more sophisticated version of the same approach, by f_1' to f_r' functions which are sufficiently indistinguishable from f_1^* to f_r^*), then the resulting modified random function F is indistinguishable from a random function (or permutation). This provides a useful method for assessing the soundness of block cipher constructions.

For instance, in the case of a three-round Feistel construction, a well-known theorem first proved by Luby and Rackoff [10] provides upper bounds on the $|p - p^*|$ advantage of any testing algorithm A in distinguishing the $2n$-bit random permutation $F = \Psi(f_1^*, f_2^*, f_3^*)$ deduced from three independent perfect random functions f_1^*, f_2^* and f_3^* from a perfect random $2n$-bit permutation F^* with q adaptively chosen queries to the tested instance of F or F^*. This advantage is less than $\frac{q^2}{2^n}$. Another example is for the $F = \Phi_{CBCMAC}(f)$ CBC-MAC construction allowing to derive a tn-bit to n-bit message authentication function from chained invocations of an n-bit to n-bit function f. It was shown by Bellare, Kilian and Rogaway that if $q^2t^2 \leq 2^{n+1}$, then the advantage of any testing algorithm A in distinguishing the random function $F = \Phi_{CBCMAC}(f^*)$ derived from a perfect nt -bit to n-bit random function using q adaptively chosen queries is less than $3\frac{q^2t^2}{2^{n+1}}$.

In proofs of security in the Luby-Rackoff model, one wants to upper bound the probability of any algorithm to distinguish whether a given fixed φ function is an instance of a $F = \Phi(f_1^*, f_2^*, .., f_r^*)$ random function or an instance of the perfect n-bit to m-bit random function F^*, using less than q queries to φ.

Let A be any distinguishing algorithm of unlimited power that, when input with a φ function (which can be modelled as an "oracle tape" in the probabilistic Turing Machine associated with A) selects a fixed number q of distinct chosen or adaptively chosen input values x^i (the queries), obtains the q corresponding output values $y^i = F(x^i)$, and based on these results outputs 0 or 1. Denote by p (resp by p^*) the probability for A to answer 1 when applied to a random instance of F (resp of F^*). We want to find upper bounds on the advantage $Adv_A(F, F^*) = |p - p^*|$ of A in distinguishing F from F^* with q queries.

As first pointed out by Patarin [11], the best advantage $Adv_A(F, F^*)$ of any distinguishing algorithm A in distinguishing F from F^* is entirely determined by the q-ary transition probabilities $Pr[\mathbf{x} \stackrel{F}{\mapsto} \mathbf{y}]$ associated with each $\mathbf{x} = (x^1, \cdots, x^q)$ q-tuple of pairwise distinct n-bit values and each $\mathbf{y} = (y^1, \cdots, y^q)$ q-tuple of corresponding m-bit values.

4.3 Partial proof techniques for hash functions and MACs

Two important areas of provable security have involved the following approaches.

- The Merkle-Damgard and HMAC constructions.

  The approach here is to show that collisions on the whole VIL (variable input length) function (resp. forgery on the whole function) implies collisions (resp. collision or forgery) on the underlying FIL (fixed input length) compression function.

- The Carter-Wegman MAC construction.

  Informally, we let H denote a family of non-cryptographic (e.g. linear) n-bit to m-bit hash functions h. H is said to be a 2-universal family of (non-cryptographic) hash functions if, and only if, given any two fixed distinct input values x and x', the $(h(x), h(x'))$ pair associated with a randomly selected $h \in H$ is distributed according to the uniform law over $\{0,1\}^{2m}$.

  Such 2-universal families of non-cryptographic functions are not only applicable to the design of unconditionally secure authentication codes (allowing to authenticate one sin-

gle message, whilst not revealing any information about the authentication code associated with any other message, but also, when combined with a secure streamcipher, to provide a provably secure n-bit to m-bit MAC allowing to authenticate an unbounded number of messages. In the latter case, the MAC key consists of the selected h function (which is kept secret as in the first case, but is used to authenticate all messages unlike in the first case) and the stream cipherkey K, and the MAC process consists in computing h(M) and encrypting this value under key K using a fresh m-bit keystream sequence generated by the stream cipher. The requirements on families of hash functions can be relaxed whilst still providing some provable unforgeability properties for the associated MAC: it was shown in [7] that a sufficient condition is that H be a so-called $\Delta - \epsilon$-universal family of hash functions (where ϵ represents a negligible probability).

Some interesting open issues might include the following.

- Finding alternative constructions for fixed input length compression functions.
- Exploring new applications of families of non-cryptographic hash functions.
- Further explorations of the interplay between different techniques; for instance viewing decorrelation modules as a generalisation of Carter-Wegman families of non-cryptographic hash functions.

4.4 Provable resistance against classes of attacks

Constructions or design methodologies that help guarantee resistance to certain types of attacks are very important. These have been particularly useful in understanding the limitations of differential and linear cryptanalysis.

- The Nyberg-Knudsen bound relating the maximum average input to output differential transition probability DPmax for an r-round, $r \geq 3$ (resp $r \geq 4$) Feistel scheme based upon r permutations f (resp r not necessarily one to one functions) to the p_{max} coefficients for the underlying f function. The initial upper bound ($2p^2_{max}$) was improved to (p^2_{max}) by Aoki et al, who also generalised the maximum average input to output linear transition coefficient LPmax for a r-round Feistel scheme (same results) and for the variant of the Feistel scheme represented by the MISTY construction. By applying these results at various levels of the nested construction of the MISTY blockcipher, Matsui derive upper bounds guaranteeing that MISTY is resistant against linear and differential cryptanalysis. A similar result can be derived for KASUMI.
- Vaudenay's decorrelation theory and resulting constructions of ciphers offering some provable immunity against attacks of order 2 (or of even larger order for some constructions such as the one proposed for the PEANUT cipher). This technique was applied to validate the resistance of the DFC algorithm against linear and differential cryptanalysis and more generally attacks of order 2. In blockciphers such as PEANUT and DFC, a new subkey embedding technique involving decorrelation modules instead of subkey addition followed by a key-independent S-box is applied.

- Direct proofs in particular provable bounds in the number of "active" S-boxes (at least 25 boxes for the 4-round AES) could be also derived for specific ciphers, and allow to derive upper bounds on maximum probabilities of differential and linear characteristics. Bounds on maximum input to output differential and linear transition probabilities (named differentials and linear hulls by some authors) are technically more difficult to derive than bounds on linear or differential characteristics where the differential or linear behaviour is specified at each round, but such bounds have also been established for AES.

Some interesting open issues might include the following.

- Proofs that exploit the deviation of the maximum average input to output differential or linear transition probabilities DPmax and LPmax (the average being taken over possible key values) do not formally preclude the existence of high probability differentials or linear input to output transition probabilities for a large fraction of key values or even all key values (provided that the high probability differentials or linear hulls are not the same for the various possible key values).
- Can provable resistance against other classes of attacks, such as algebraic attacks, be provided?
- Can we provide new design methodologies based upon partially provable constructions as done in MISTY, KASUMI, and DFC?

References

[1] M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag, 1996.

[2] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J.Computing*, 13(4):850–863, 1984.

[3] R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. *Theory of Cryptography Library Record 99-01*, 1999.

[4] J.-B. Fischer and J. Stern. An efficient pseudo-random generator provably as secure as syndrome decoding. In *Advances in Cryptology – EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 245–255. Springer-Verlag, 1996.

[5] O. Goldreich, S. Goldwasser, and S. Micali. On the cryptographic applications of random functions. In *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 276–288. Springer-Verlag, 1985.

[6] O. Goldreich and L. Levin. A hard-core predicate for all one way-functions. In *21st ACM Symposium on Theory of Computing*. ACM press, 1989.

[7] S. Halevi and H. Krawczyk. MMH: Software message authentication in the Gbit/Second rates. In *Fast Software Encryption – FSE'97*, volume 1267 of *Lecture Notes in Computer Science*, pages 172–189. Springer-Verlag, 1997.

[8] J. Håstad and M. Naslund. BMGL: Synchronous key-stream generator with provable security. Technical report, Nessie project, 2001.

[9] R. Impagliazzo, J. Håstad, L. Levin, and M. Luby. Pseudo-random generation from one-way function. In *21st ACM Symposium on Theory of Computing*, pages 12–24. ACM Press, 1989.

[10] M. Luby and C. Rackoff. How to construct pseudorandom permutations and pseudorandom functions. *SIAM J.Computing*, 17(2):373–386, 1988.

[11] J. Patarin. New results on pseudorandom permutation generators based on the DES scheme. In *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 301–312. Springer-Verlag, 1991.

[12] A. Shamir. On the generation of cryptographically strong pseudo-random sequences. In *ICALP*, volume 544-550. Springer-Verlag, 1981.

[13] A.C. Yao. Theory and applications of trapdoor functions. In *23rd IEEE FOCS*, pages 80–91, 1982.

5 Industrial Needs

Cryptography is a science and a technique and its ultimate goal is applicability. The conversion of scientific results on cryptography into actual products is performed mostly by private companies and public organisations who wish to incorporate security features within various applications. These organisations are collectively described as "the industry" and the purpose of this section is to provide a rough sketch of what are their current needs in the area of symmetric cryptography; in other words, what are the problems which the industry would like the researchers to tackle most immediately.

These needs can be sorted into the following categories:

- standards;
- secure protocols;
- high-performance specialised algorithms;
- random number generators;
- implementation issues.

We will now review what these categories are, what already exists and what is still eagerly expected by the industry.

5.1 Standardization

A variety of standardization challenges are presented.

5.1.1 Data representation

Scientific research on cryptography operates on abstract, mathematical objects such as integers or elements in a finite field. Industrial applications are more concerned about sending and receiving streams of bits or bytes[4]. For proper interoperability, an algorithm must be specified *completely* and *unambiguously*, which means that two distinct and independent implementers should be able to produce the exact same output with the same input without resorting to "common knowledge". For instance, byte ordering in the representation of an integer value (the "endianness" problem) must be specified.

For symmetric cryptography, what algorithm (scientific) descriptions most often lack is a precise definition of the ordering of bits within a byte, and the ordering of bytes within a multi-byte integer value. Some standards, such as the AES [4] and SHS [3] do it correctly: they take great pain to define bit and byte ordering precisely.

Without such a precise definition, any algorithm specification is next to useless for the industry; if the algorithm is really needed (for instance, it has very good properties which no properly specified existing algorithm provides), the industrial organisation will fill in the blanks with its own proprietary definitions, and thus is interoperability forfeit.

[4]We use here "byte" to designate an 8-bit quantity; "octet" could equally have been used.

5.1.2 Responsibility

Apart from being clear and precise, a good standard must be *accepted* and *maintained*. What this means is that the industry will be reluctant to use a standard unless it originates from a source which is both generally considered as authoritative, and also responsible for the production and publication of amendments when necessary.

For instance, the FIPS publications by the NIST [11] (such as AES and SHS) fit this description well. They are backed by the US government, and as such their use is mandated throughout many government applications; conformance to these standards can be qualified as a legal obligation. Moreover, the NIST is officially responsible for the contents of the standards, and *must* publish revisions when the need arises. Using these standards is economically justified.

The RFC [8] system is not as good. RFC publication is relatively easy (and free), but most RFCs are tagged as "informational" which gives no hint on their acceptance and foreseeable future. Only those RFCs which are related to Internet protocols get a chance to become RFC "standards". RFCs are *never* modified in any way; revisions may be issued but there is no guarantee of that.

A good example of this is the TLS protocol [15] and the RC4 stream cipher: RC4 is, technically, a trademark corresponding to a secret algorithm. However, there exists another algorithm (usually called "alleged RC4" or "Arcfour") which is not secret and which appears to be compatible with the official RC4. The "alleged" situation is not satisfactory, hence a new RFC describing Arcfour was scheduled, and the TLS RFC specifies that Arcfour, as described in that new document, is compatible with the official RC4. This was stated when the TLS RFC was written, back in January 1999. However the Arcfour description was never published and, since nobody is responsible for this document, it cannot be said when, if ever, Arcfour will be described in a public authoritative reference.

5.2 Secure protocols

Cryptographic algorithms are only bricks which are combined into *protocols* which provide some high-level security features. An example of such a protocol is TLS [15], which combines symmetric encryption, MACs and asymmetric cryptography (key exchange and signature) in order to provide a confidential authenticated integrity-checked two-way tunnel for arbitrary byte streams; the underlying medium is any other two-way tunnel for byte streams.

The performance and adequacy of a protocol in a specific situation is highly dependent on the protocol definition. The regular "standard" protocols are usually high-level and best suited to Internet-like communications. For small lightweight applications (e.g. embedded mobile devices), implementers are reluctant to implement TLS or IPsec tunnelling because they cannot afford a fully functional IP implementation. For these devices special lightweight protocols must be designed which have provisions for the characteristics of the environment.

5.2.1 Encryption modes

A raw symmetric encryption algorithm is either a block cipher or a stream cipher.

A block cipher encrypts only one block; to encrypt more, the incoming data must be split into several blocks and a special mode of encryption used. The most nave mode, the ECB mode, is known to be weak. The usual recommendation is to use CBC [6] mode, but this has the following problems:

- CBC needs an initial value which should be added to the encrypted message, thus enlarging it. If the IV is not added, but inferred from some contextual information, then some weaknesses may arise depending on the way the IV is computed.
- CBC encrypts data only if it has a length which is a multiple of the block size. If the underlying data does not have this property then some padding must be applied, which usually results in further message length increase. Some specific padding modes have been devised to avoid this problem (e.g. "ciphertext stealing", aka CTS) but their security still needs some serious analysis.
- CBC can be interleaved by splitting the input into separate streams in order to improve throughput (e.g. on Triple DES platforms that have three DES processors). But, in general, CBC cannot be made parallel. This is a problem for high-bandwidth devices.

Stream ciphers do not have padding issues, but they have security issues when a key is reused. The usual approach is to include an IV and combine it somehow with the key, but the actual combining process is rarely described, and can have adverse effects if not done properly. Moreover, if the IV cannot be derived from some contextual information (e.g. message sequence number), it must be sent along with the message, which increases its length. Note that a block cipher in CTR mode is only a way to make a stream cipher from a block cipher; the IV in this case is the counter initial value.

The industry needs modes of encryption which provide some or all of the following characteristics:

- little or no increase in message size;
- precise and complete specification of proper ways to derive IVs and other values;
- possibility of parallel implementation (for high-speed devices with specific hardware);
- low cost;
- possibility of encryption of very short messages;
- if possible, patent-free.

The ability to encrypt very short messages (less than eight bytes, for instance) without expansion is a requirement that is important to applications with severe constraints on network bandwidth. What happens from a security perspective when encrypting very short messages is not well known by public research, and consequently no really secure way to do it is currently described.

The "cost" of an encryption mode is difficult to express generically, because it depends on the actual implementation context. On desktop computers and big servers, the cost is mostly

the raw execution cycle count and the additional pressure on the CPU level 1 cache. On high-speed hardware platforms with specialized circuits, a more adequate measure is the die surface needed to achieve some specified bandwidth. On low-power devices (e.g. smartcards), the most important characteristics are RAM and ROM consumption (RAM is very limited, but ROM is not very large either – this prohibits the implementation of more than one or two basic algorithms, and also the use of large pre-computed tables).

5.2.2 Combined encryption and MAC

MACs are used to provide integrity checks on data. Assurance of integrity is needed in most protocols, in order to thwart active attacks (e.g. modifying data). Although such attacks are usually much more difficult to perform than simple eavesdropping, they are nonetheless increasingly important for the industry.

Integrity checks can be used to provide authentication (by knowledge of a shared secret), independently of any need for confidentiality; however, it is often the case that both confidentiality and data integrity are needed. Some historical applications (e.g. GSM mobile phones) use encryption for data authentication, by having some conventional data encrypted and checked. Such a way to build a MAC is known to be insecure in most situations; for instance, the WEP protocol, designed to protect WiFi connections, was a spectacular failure in that matter. However, industrial implementers are often reluctant to compute both encryption and MAC, because it basically doubles the cost. This is especially true for low-power mobile devices, which have limited computing and electrical power.

Therefore there is a clear need for modes of operation which combine the functions of encryption and MACs, and which at the same time have the desirable characteristics identified in the previous section (low message length increase, possibility of parallel computation,...). Recently, several new modes have been proposed, such as CWC [10], CCM [16], EAX [1] or OCB [17]. It should be noted that those new modes with the best performance are not patent-free.

5.2.3 Hash functions

A good hash function must have specific properties such as collision resistance. Although other candidates have been proposed, the most used nowadays derive from the MD family[5]. MD4 [12] is considered as broken, and its successor MD5 [13] is also technically broken. The SHA family [3] (SHA-1, SHA-256, SHA-384 and SHA-512) is still considered secure, but some new results on SHA-0 (a predecessor to SHA-1) raise some concerns about the whole family.

Moreover, implementing both a block cipher and a hash function in a limited low-power device (e.g. a smartcard) can be troublesome, due to hard limitations on the ROM size, or die surface. It is conceivable to use a key-agile block cipher as a hash function by using the data as private key, with some chaining and padding; however, this problem has not been well studied, and has been the subject of only limited standardization (see ISO/IEC 10118-2 [9]). Such a construction also seems to exercise some security properties of the block cipher,

[5] We consider here MD4, MD5 and the various SHA-*; MD2 is structurally completely different.

properties which are seldom considered as important enough to warrant extensive study (e.g. weak keys).

A generic way to build a secure hash function out of a block cipher would be most appreciated by the industry. This would allow low-power devices to implement more protocols with less resources; optimised encryption cores could be reused cost-effectively for hashing and, consequently, MAC computation (with the HMAC [14] construction). Such a construction would have to be carefully analysed, especially with regards to the actual security properties which the underlying block cipher would need to provide; the usability of the existing standard block ciphers (such as the AES [4] or 3DES [5]) should then be assessed.

5.3 High-performance specialised algorithms

Some applications need specialised algorithms with very high constraints on performance. They can be split into roughly two groups:

- high-speed specialised network nodes;
- low-power devices.

5.3.1 High-speed specialised network nodes

We consider here devices which have to handle huge amounts of data; the cryptographic algorithms they use must be able to process data with a very high bandwidth and very low latency.

High bandwidth is usually achieved using pipelining, and this is possible so long as the algorithm itself can be expressed as a circuit. This is true for most, if not all, block ciphers, but the mode of operation is also important because non-parallel modes such as CBC defeat pipelining.

Latency cannot be reduced easily. Among block ciphers, those with little diffusion and many rounds usually imply a high latency because the critical path for each data bit must traverse many layers. Modern block ciphers are quite good in that respect (but there is always an application for which the existing method is not "good enough").

5.3.2 Low-power devices

Low-power devices are applications where cryptography must be applied by hardware which is very limited in either or both of computational power and electrical resources. Some extreme applications are RFID tags, which receive very low power through electromagnetic induction, and have very limited time to compute and send back an answer through radio waves. Other examples include Bluetooth-enabled devices which are often battery-powered and yet must sustain radio communications with medium to high data bandwidth.

Most of those applications have to ultimately rely on a general purpose 8-bit or 16-bit processor, with limited room for code and static data (in ROM) and *very* limited room for mutable data (in RAM). Most modern cryptographic algorithms are designed as a compromise

between high speed on workstations and "generically acceptable performance" for low-power embedded devices. It so happens that some applications require better than "generically acceptable"; the industry needs some algorithms which can be relied upon for security and which perform well on limited platforms. Trust in the security of an algorithm can be achieved only with thorough peer evaluation; such work has been done for block ciphers (the AES competition) and for other algorithm types (that's what the NESSIE project was about) but not in the specific context of low-power devices.

5.4 Random number generators

Random number generation is difficult, especially for cryptographic purposes, because the "quality" of the produced random data cannot be measured. Usual statistical tests provide only a very superficial view of the problem; cryptographic protocols require computational unpredictability. A random number generator which does not pass successfully the statistical tests is bad indeed; but a generator which does cannot be thus declared "good".

A cryptographic random number generator is usually the combination of some *seed*, which is a random value provided externally, and a pseudo-random number generator (PRNG) which expands that seed into an arbitrarily long stream of bits which are computationally indistinguishable from random bits. The two main problems are:

- how to make a good seed;
- how to define and implement a good PRNG.

5.4.1 Random seeds

A seed is expressed as a value in a format which is suitable for the purpose of the PRNG which will be used; it usually is a stream of bits of some specified length (160 bits is common practice). The seed must have an *entropy* which is good enough to thwart exhaustive search; actually, entropy can be defined as the average cost for an exhaustive search to succeed.

Entropy estimation and concentrations are the two main parts of the problem of producing seeds. Entropy estimation is about measuring how much "unpredictability" can be attached to the result of measuring a physical event. For instance, a coin flip yields one bit of entropy – if we assume that the coin is not biased in any way. Entropy sources commonly used vary, depending on the context:

- In embedded devices such as smartcards, a specialized hardware random number generator is mandatory; usual generators use a reverse-biased PN junction, implemented with Zener diodes or a smart combination of transistors.
- In workstations, the physical events are usually precise timings of external interruptions, which correspond to network activity, key strokes, etc. Some processors include a hardware random number generator which uses a technology equivalent to those used in smartcards.

- There are some environment where almost no source of randomness is supplied, for instance virtual machines used to run applets. Those machines are *meant* to not have any randomness. There exists a seed gathering procedure which has been published for the Java Virtual Machine; the idea is to measure the efficiency of the thread scheduler, because that efficiency should depend on the actual load of the host multitasking machine.

Once random data has been gathered, up to some estimated amount of entropy, that data must be concentrated in order to produce the seed. The usual way to do this is to input the entire data into some hash function, whose output will be the seed. This process should be more carefully analysed from a security point of view; moreover, using a hash function means that this function must be implemented, which can be a problem for smartcards and other devices where code size is very limited.

To sum up, the industry has the following needs with regards to seed generation for cryptographic purposes:

- recipes for gathering random data in various situations;
- accurate estimators for the entropy thus obtained;
- secure ways to concentrate random data, using either a hash function, a block cipher or a stream cipher.

5.4.2 PRNG

The performance requirements for a PRNG are about the same as for encryption software. The main difference is that a PRNG has no concept of input bandwidth, just output bandwidth. An encryption system which just encrypts an endless stream of zeroes is supposed to be a good PRNG; similarly, a PRNG can be transformed into a reasonable stream cipher by using the seed as a key, and combining the output bits with the data using a bitwise exclusive-or.

Parallelism is still important: it can be exploited by hardware implementations to provide better output bandwidth; but the same effect can be achieved by including several instances of the generator, working on several seeds which have been derived from a master seed using another PRNG.

Another detail which makes PRNG performance easier than encryption performance is that a PRNG works only for producing random bytes, whereas an encryption system definition must be wary of the feasibility and performance of the corresponding decryption system.

What the industry needs here is a list of research-approved ways to build a PRNG using a primitive cryptographic operation, where that operation may be either a hash function, a block cipher or a stream cipher.

5.5 Implementation issues

5.5.1 Side-channel attacks

The implementation of cryptographic algorithms is a complex matter, because it deals with security and this is not easily measured. An implementation which passes all test vectors may still have security issues related to, for instance, side-channel attacks.

Such attacks exploit some data leakage due to an implementation detail; the usual leak mediums are timing (algorithm computation time is not independent of the processed data and secret key) and power consumption (especially for smartcards or other devices which have an external power supply). Defending against such leaks is not easy, especially with power consumption for smartcards, because power is provided externally, by a potentially hostile entity.

Devising generic ways to implement primitive operations in ways which do not leak private data are an active research area, and much work still needs to be done.

Another usual leak medium is an error reporting channel. For instance, some SSL implementations using RSA were successfully attacked by Bleichenbacher [2] because those implementations leaked information about the RSA private key by telling at which point in the SSL handshake some invalid client message made the computation fail. In brief, precise error reporting is considered as a good property for most systems, but it can have an adverse effect on security; the cryptographers should analyse and specify in extensive details where and what error reporting must be avoided.

5.5.2 Testing

Implementation of symmetric algorithms can be difficult to test. The very nature of the algorithms implemented (with propagation of errors and avalanche effect) implies that most flawed implementations are spectacularly faulty: a flawed implementation of AES will usually give the wrong answer for almost all possible input blocks. However, some algorithms use operations which can be incorrectly implemented in subtle ways.

One such example is the DFC block cipher [7]. That algorithm includes a modular affine transform; the reduction is performed modulo $2^{64} + 13$. It is relatively difficult to implement that reduction in a way which is both correct for every input, and also efficient. It is actually easy to implement it in a way which is correct for most inputs but incorrect with a probability of 2^{-64} for random input data. Random test vectors have a very low probability of catching such an error; specific test vectors must be devised, so as to exercise the specific input values which may be handled improperly.

Many algorithms use hard-coded look-up tables, where the tables have been chosen for specific properties. A typing error could corrupt one table entry and remain undetected by the test vectors if none of them uses that table with the corresponding entry.

In brief, for an algorithm specification to be properly usable by the industry, appropriate validation procedures (test vectors, mostly) should be included. These procedures must be defined so as to be very likely to catch the most common implementation errors.

5.6 Ongoing challenges

Here is a brief summary of the most common industrial needs in symmetric cryptography:

- precise standards actively maintained and supported by large, and if possible public, organisations;
- new enhanced modes of operations, both for encryption and combined modes which provide both encryption and authentication;
- precise guidelines on the handling of related data such as IVs;
- specialised algorithms for use in contexts were usual algorithms do not provide adequate performance, especially low-power embedded devices;
- analysed hardware random number generators with accurate entropy estimators;
- high-performance secure PRNGs based on various algorithms such as block ciphers, hash functions and stream ciphers;
- guidelines for methods which reduce secret data leakage through side channels;
- proper validation procedures for all standard algorithms.

In particular, there is a strong need for high-level constructions (hashing, PRNG, MAC, etc) which use a block cipher as the unique underlying cryptographic operation. This is for small, lightweight applications which cannot afford the concurrent implementation of a block cipher and a hash function.

References

[1] M. Bellare, P. Rogaway, and D. Wagner. The EAX mode of operation. In *Fast Software Encryption - FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 389–407. Springer-Verlag, 2004.

[2] D. Bleichenbacher. Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS#1. In *Advances in Cryptology - CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 1–12. Springer-Verlag, 1998.

[3] FIPS 180-2. Secure Hash Standard. Federal Information Processing Standards Publication 180-2, 2002. U.S. Department of Commerce/N.I.S.T.

[4] FIPS 197. Advanced Encryption Standard. Federal Information Processing Standards Publication 197, 2001. U.S. Department of Commerce/N.I.S.T.

[5] FIPS 46-3. Data Encryption Standard. Federal Information Processing Standards Publication 46-3, 1999. U.S. Department of Commerce/National Bureau of Standards.

[6] FIPS 81. DES Modes of Operation. Federal Information Processing Standards Publication 81, 1980. U.S. Department of Commerce/National Bureau of Standards.

[7] H. Gilbert, M. Girault, P. Hoogvorst, F. Noilhan, T. Pornin, G. Poupard, J. Stern, and S. Vaudenay. Decorrelated Fast Cipher: an AES candidate, 1998.

[8] The Internet Society (ISOC). RFC-Editor.

[9] ISO/IEC 10118-2. Information technology – Security techniques – Hash-functions – Part 2: Hash-functions using an n-bit block cipher algorithm. International Organization for Standardization, 1994.

[10] T. Kohno, J. Viega, and D. Whiting. CWC: A High-Performance Conventional Authenticated Encryption Mode. In *Fast Software Encryption - FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 408–426. Springer-Verlag, 2004.

[11] National Institute of Standards and Technology (NIST). Federal Information Processing Standards (FIPS).

[12] RFC 1320. The MD4 Message Digest Algorithm. Internet Request for Comments 1320, 1992. R.L. Rivest.

[13] RFC 1321. The MD5 Message Digest Algorithm. Internet Request for Comments 1321, 1992. R.L. Rivest.

[14] RFC 2104. HMAC: Keyed-Hashing for Message Authentication. Internet Request for Comments 2104, 1997. H. Krawczyk and M. Bellare and R. Canetti.

[15] RFC 2246. The TLS Protocol. Internet Request for Comments 2246, 1999. T. Dierks and C. Allen.

[16] RFC 3610. Counter with CBC-MAC (CCM). Internet Request for Comments 3610, 2003. D. Whiting, R. Housley and N. Ferguson.

[17] P. Rogaway, M. Bellare, and J. Black. OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. *ACM Trans. Information System and Security*, 6(3):365–403, 2003.