Block Ciphers

RSA Laboratories Technical Report TR-601
Version 2.0—August 2, 1995

M.J.B. Robshaw
matt@rsa.com

RSA Laboratories
100 Marine Parkway
Redwood City, CA 94065-1031

003-903041-200-000-000

Contents

1 Introduction

This technical report provides a review of the design and analysis of block ciphers. Its format is built around the twin ideas of design and cryptanalysis and with this in mind we present some of the basic design principles and general methods of cryptanalysis before describing the algorithms. It is hoped that the sections of review and analysis are quite self-contained and that sections can be studied independently of one another.

It is clear to anyone reading the literature that the Data Encryption Standard (DES) [105] has been, since 1977, the focus of practically all research in both the design and cryptanalysis of block ciphers and we feel that this remarkable cipher deserves its own section. This means that other block ciphers are sometimes grouped together and, in some cases, will receive only a short description of their functionality and perceived security together with references which will guide the reader towards further information. This approach may be criticized for providing too much emphasis on DES, but we feel that any survey of block ciphers which aims to reflect the cryptographic literature would be forced to do likewise.

The report closes with two sections that discuss issues that are common to all block ciphers: namely the modes of use and the use of multiple encryption. The latter recently became a particularly important area of cryptanalytic research.

Before closing this section we will provide an answer to the frequent request for a concise distinction between block ciphers and stream ciphers. The issue is slightly complicated by the fact that several modes of use of a block cipher allow it to be used as a stream cipher (see Section 11).

We will quote Rueppel [140]:

> *Block ciphers* operate with a fixed transformation on large blocks of plaintext data; *stream ciphers* operate with a time-varying transformation on individual plaintext digits.

To summarize: using a block cipher in its basic mode to encrypt the same plaintext block using the same key at a different time will yield the same ciphertext block. When using a stream cipher to encrypt the same plaintext digit with the same key but at a different time (where a digit is being used to describe the smallest packet of plaintext information in the implementation, usually a bit) it is not necessarily the case that the same ciphertext is obtained.

1.1 Notation and terminology

Two important attributes of a block cipher are the size of the key and the size of the blocks on which the cipher operates. The data that is encrypted is called the *plaintext*, or sometimes *cleartext*, and it is encrypted to give the *ciphertext*. The *key* is some secret information chosen by those wishing to communicate. The key is the same for both the sender and the receiver[1] and block ciphers give examples of what are termed *symmetric cryptosystems*.

Anyone possessing the key can decrypt the encrypted messages and the fact that both participants have to agree on a secret key before secure transmission can take place introduces problems beyond the scope of this report. These problems are addressed by the fields of key management and key distribution [39, 79]. We note that some modes of use of a block cipher require the use of what is termed an *initialization value, IV*. The value of the IV is often publicly known (since the security of the cryptosystem does not depend on this value being kept secret) and it is not considered to be part of the key.

A block cipher which operates on plaintext blocks of size n will be called an n-bit block cipher, and the encryption of plaintext m using the chosen cipher under key k will be written as $E_k(m)$. Similarly, decryption of the ciphertext c will be written as $D_k(c)$ and clearly $D_k(E_k(m)) = m$.

2 Design

2.1 Confusion and diffusion

In his landmark paper of 1949 [138] Shannon presents the principles of *confusion* and *diffusion*. So successful are they in capturing the essence of the desired attributes of a block cipher that they have become the corner-stone of block cipher design.

Confusion is described as being "the use of enciphering transformations that complicate the determination of how the statistics of the ciphertext depend on the statistics of the plaintext"[140] or, more briefly, to make the relation between the key and the ciphertext as complex as possible [6, 20].

Good diffusion, meanwhile, spreads the influence of individual plaintext characters over as much of the ciphertext as possible, thereby hiding the statistical features of the plaintext.

While it is clear that both these properties might be considered essential in the design of a secure block cipher, we note by way of contrast that in

[1] In contrast to public-key or asymmetric cryptography [47].

the field of stream ciphers, the *one-time pad* or *Vernam cipher* [146], first rigorously analyzed by Shannon and described as having *perfect secrecy*, does not employ diffusion and relies entirely on confusion.

2.2 Product ciphers and iteration

The question of how best to achieve good diffusion and confusion lies at the heart of block cipher design. Two simple ciphers, each quite weak on their own, are often employed as small parts of the larger, and hopefully secure, block cipher.

When the source of the plaintext is known, a great deal of information is immediately available to a waiting cryptanalyst even though nothing might be known about the actual message being encrypted. There are two attributes that are particularly important.

The set of all plaintext characters is known as the alphabet, by analogy with the set of letters in a natural language, and the frequency with which these letters occur in a large collection of plaintexts may not be uniform. For instance, suppose the plaintext is known to be a general message written in English. Since the letter e is the most commonly occurring letter (occurring roughly 13% of the time [6]), analysis of the distribution of characters in the ciphertext produced by a poor cipher might reveal information about the plaintext message as well as the key.

A *substitution cipher* replaces certain characters, or preferably, groups of characters, with others from the same alphabet. (When we consider groups of characters such as pairs or triples then the alphabet becomes the set of all possible pairs and triples.) Such a substitution on a large alphabet makes knowledge of the statistics of the source less relevant.

A second important attribute of a plaintext source is the *redundancy*. To be able to understand a sentence written in English, it is not always necessary to see the entire text. Omitting the u which immediately follows a q in an English sentence will not greatly affect its comprehension. The pairing of q's and u's in English is an example of redundancy and a good cipher will ensure that knowledge of the redundancy in the plaintext source is of little help in decrypting a message.

A *transposition cipher* permutes the plaintext characters, altering the statistical appearance of groups of characters and thereby reducing any advantage provided by the redundancy in the plaintext source.

As we have said, the substitution and transposition ciphers, or at least the principles they exemplify, are often too weak to be of use independently. However, by repeatedly mixing these two ciphers in different combinations

it is possible to develop a strong *product cipher*.

We might feel that repeating the application of some complicated enciphering operation would provide increased complexity and hopefully, increased security. While this need not necessarily be the case when ciphers have a particularly unfortunate design, it is the principle behind a class of ciphers called *iterated block ciphers*. Indeed, work by O'Conner and Golić [120] might be used to provide theoretical justification for this intuitive belief.

In an iterated block cipher, a complex (but perhaps weak) round function is used repeatedly, each time taking as input the output from the previous round. The most familiar example of such a cipher is DES, and the iterated structure in DES has its origins in the *Feistel cipher*, which we describe next.

Interestingly, this iteration of a 'weak' round function might also provide an achilles' heel leading to the cryptanalysis of the block cipher. Several techniques, that we will discuss later, rely on the fact that a weakness in one round of the cipher can be exploited to mount an attack on the outside rounds of the full cipher.

2.3 Feistel cipher

A cipher that is called the *Feistel cipher* by some commentators dates from 1974 [51] and is an embodiment of the following encryption procedure.

Consider a plaintext block m of length n which we wish to encrypt as two blocks m_0 and m_1 of size $n/2$. We write $m = m_0 m_1$. Given a key k, we can define a set of *subkeys* $k_1 \dots k_r$, one for each of r rounds, where each subkey k_i acts as input to a transformation $f(k_i, \cdot)$ on the set of blocks of size $n/2$.

Define $m_2, \dots, m_{r+1}$ where

$$m_i = m_{i-2} \oplus f(k_{i-1}, m_{i-1}).$$

Most frequently, this recurrence is visualized as r identical round transformations followed by a swap of the two halves of the data. In such a visualization, however, there is no swap after the final round and the output of the cipher is given by $m_{r+1} m_r$.

The important feature to notice is that if we start the encryption procedure with $m_{r+1} m_r$ and use the subkeys in the reverse order $k_r, \dots, k_1$, then the output is $m_0 m_1$. This property holds irrespective of which functions $f(k_i, \cdot)$ are used in the encryption process.

This is a particularly nice feature when it comes to the implementation of a Feistel cipher since we do not need to implement two different algo-

rithms, one for encryption and one for decryption. Instead we reverse the scheduling of the subkeys and this allows us to use the same algorithm for both encryption and decryption with the same key k.

This design of a Feistel cipher has been used repeatedly in many block ciphers. It is a particularly appealing structure and as we have seen it is a design which has substantial advantages.

2.4 Performance issues

Many techniques are used in the design of secure block ciphers and they each have different performance attributes. The most fundamental difference seems to be whether a cipher is intended for use in software or hardware. As an example DES [105] performs very well in hardware (see Section 4) but has a number of features that frustrate software implementation such as bit-level permutations.

Other considerations might include the following. Fixed permutations are operations that are easy to execute in hardware, just by hard-wiring the relevant connections, but varying or key-dependent permutations are more difficult. For a software implementation, memory is cheap and the amount of memory required need not be such a concern as it might be when physical space is at a premium for a chip implementation. Consequently large look-up tables (perhaps computed as a function of the user-chosen key) might be considered for a software implementation, since they would be fast to use, but they might not be so desirable in hardware. Note however, that such look-up tables should not be so large as to force an implementation to access slow memory [126].

Hardware trends are now moving towards 32-bit machines and often arithmetic operations are offered at a low level; some arithmetic operations are now particularly easy to implement and fast to run. This has become the motivation for an increasing number of new designs which incorporate a mixture of arithmetic operations in an attempt to get adequate security.

It is becoming apparent that there is a need for a block cipher that can perform at high speed in software. Unfortunately, several of the faster block ciphers that have recently been proposed remain relatively untested or they are already broken. For a useful survey of the software performance of various algorithms and some of the issues involved, Preneel's paper "Software performance of encryption algorithms and hash functions" [126] can be recommended. A less recent comparison of algorithm performance is given by Roe [133].

3 Cryptanalysis

In this section we shall describe some of the cryptanalytic attacks that are applicable to a wide range of block ciphers. There has been a resurgence of interest in cryptanalysis in recent years with several new techniques being particularly successful on some old adversaries. There are of course many techniques that are only applicable to a few or even to a single cipher and these techniques will be described in the relevant sections on each of the block ciphers.

3.1 Classes of cryptanalytic attack

Cryptanalytic attacks are often classified according to the type of information that is available to the cryptanalyst who carries out the attack.

One of the basic assumptions in the field of cryptography is that the cryptanalyst has full knowledge of the algorithm that is being used. Thus the security offered by the system is due entirely to the fact that the key is kept secret; this notion is known as *Kerckhoffs' assumption* or *principle*[2].

It is usually assumed that the cryptanalyst has access to the ciphertext that is being transmitted. If this is the only information available then the attacks that can be considered by the cryptanalyst are termed *ciphertext only*. The cryptanalyst sees t ciphertexts $c_1, \ldots, c_t$ and uses these in the attack (possibly with knowledge of the plaintext statistics as well). A successful attack relying solely on the use of ciphertext is, in the case of a good block cipher, particularly difficult to devise. On the other hand, if such an attack does exist, then it is very easy for a cryptanalyst to gather the relevant information to implement the attack.

It is possible that the cryptanalyst knows more than just the ciphertext and in fact knows some of the plaintext corresponding to the visible ciphertext. An example of this might occur when the data being encrypted is highly formatted, or perhaps when a message is known by some party other than those communicating. Now the cryptanalyst sees n ciphertexts $E_k(m_1), \ldots, E_k(m_n)$ and knows the original plaintexts $m_1, \ldots, m_n$.

An attack that relies on the knowledge of the plaintext encrypted is termed a *known plaintext* attack. While devising such an attack in theory might be easier than a ciphertext only attack, it is much harder practically to use this attack. Not only must the cryptanalyst collect the ciphertext,

[2] A.Kerckhoffs (1835–1903) formulated the basic ground rules of cryptographic study in his book *La Cryptographie Militare*.

but if this is to be of use then the plaintext corresponding to this ciphertext must also be known.

Note that it is always possible for a cryptanalyst to guess the plaintext corresponding to some intercepted ciphertext. If very little known plaintext is required for a successful attack when compared to a ciphertext only attack, then the cryptanalyst may well find it advantageous to try this approach particularly if there are relatively few choices for the plaintext.

A third broad class of cryptanalytic attack is called *chosen plaintext*. In such a case the cryptanalyst actually gets to choose the plaintext that is encrypted as well as seeing the ciphertext that is generated. The cryptanalyst chooses messages $m_1, \ldots, m_n$ for encryption and receives in return $E_k(m_1), \ldots, E_k(m_n)$. An *adaptive chosen plaintext* attack is one for which a message block m_t is only chosen for encryption after the values of $E_k(m_1), \ldots, E_k(m_{t-1})$ have already been observed.

A chosen plaintext attack might perhaps be the easiest to devise (of the three we have encountered so far), but it would almost certainly be the most difficult to mount in practice. There are examples however where such an attack is not necessarily impractical, for instance when the party under attack (such as a database server with a fixed key) encrypts data on request.

Merkle and Hellman [98] describe a chosen plaintext attack, on a standard for multiple encryption [1] (Section 12), as a "certificational attack". A chosen plaintext attack is, in general, a difficult attack to mount, particularly when it requires a vast amount of chosen plaintext. However, Merkle and Hellman go on to say "In many cases, ciphers which have yielded to chosen plaintext attacks have later proven vulnerable to known plaintext or ciphertext only attacks as well." This prophetic view was vindicated when van Oorschot and Wiener [123] presented a known plaintext attack on the same multiple encryption scheme that had been attacked by Merkle and Hellman using chosen plaintext.

Since the collection of chosen plaintext might well be impractical, certain cryptanalytic attacks will use probabilistic arguments to convert them from chosen to known plaintext. The idea is that generating huge quantities of known plaintext will eventually yield sufficient plaintext with the properties required to simulate a chosen plaintext attack [17]. However the vast increase in the amount of data that must be collected would in all likelihood mean that the attack is no more practical than the original chosen plaintext attack.

A final class of attack is termed *chosen ciphertext* and the idea is that the cryptanalyst gives chosen ciphertext for decryption and receives the equivalent plaintext. Thus the cryptanalyst submits $E_k(m_1), \ldots E_k(m_n)$ for decryption and receives $m_1, \ldots, m_n$. This form of attack has some relevance

to television cable decoders, for instance, which decrypt on demand but have the keys concealed within them. The term *chosen text* attack is sometimes used to describe collectively the pair of attacks - chosen plaintext and chosen ciphertext.

Other attacks rely on the special conditions provided by individual cryptosystems to dictate what information is needed by the cryptanalyst. These special attacks will be introduced in the relevant sections of the report.

3.2 Work effort and brute force

What does it mean for an attack to be successful? An attack is successful when the key that was used for the encryption of some plaintext m_i to give ciphertext $E_k(m_i)$ can be deduced, or, when the decryption of some previously unseen ciphertext $E_k(m_{n+1})$ can be deduced from the information available to the cryptanalyst without requiring a prohibitive amount of work.

This answer has provided us with another question - What is a prohibitive amount of work? There is no simple answer to this question since the effort that can be invested by an individual in both time and money is likely to be insignificant when compared to that available to a large organization.

A convenient representation of computing power is provided by the concept of the MIPS-year. A MIPS-year is the number of operations completed by a machine running for a year at the rate of one million instructions per second. A million or around 2^{20} instructions would take 0.04 seconds on a 25 MIPS computer which, by today's standards, is typical of a high-end personal workstation. By contrast 2^{56} operations (a number equal to the number of possible keys for DES) would require $2^{36} \times 0.04$ seconds or approximately 87 years (though of course it takes more than one operation to test a DES key). Being able to invest in parallelization and more powerful machines clearly allows a considerable reduction in the time required for a computation.

The feeling at present seems to be that a work effort of 2^{56}, that is 2^{56} operations (perhaps DES encryptions but not necessarily) can no longer be considered safe, 2^{64} would offer perhaps a minimum level of security, and a work effort of 2^{80} is likely to be safe for the near future.

The interim report [21] into the security of the Skipjack encryption algorithm (see Section 9) makes the following calculation. Allowing for a doubling in the performance of computers every 18 months, it will be 36 years before a work effort of 2^{80} is comparable to the work effort required today to complete 2^{56} operations. The Skipjack encryption algorithm has

an 80-bit key. The general security level at which the components of the Capstone project [45, 50] (of which Skipjack is a part) have been aimed is 80 bits.

Considering DES [107] in particular, it is worth emphasizing that even if an attack appears to break the 2^{56} barrier for the work required, then the attack might still not be practical. Much depends on the kind and the quantity of the information that is required for cryptanalysis; accumulating 2^{47} chosen plaintext/ciphertext pairs for instance, is not a straightforward matter.

We note that with DES in mind, the encryption of 2^{40} blocks of data would take almost 100 days on a single processor running at an encryption rate of 1 Mbyte/sec. It is clear that collecting 2^{43} blocks of encrypted data is a difficult task (the best linear cryptanalytic attack on DES requires 2^{43} known plaintexts) while the collection of 2^{47} blocks (the amount required for the best differential cryptanalytic attack) is quite prohibitive.

3.3 Exhaustive key search

The most basic attack that can be mounted on a block cipher is that of exhaustively testing each of the possible candidates for the key k in turn until a match is obtained. If the key is of length b bits then there are clearly 2^b possible keys, though we would expect to have to try 2^{b-1} until we find the correct one. We have already given some feel for how much work is required in trying out 2^b keys for various values of b. We note, however, that the size of the key is irrelevant when the block cipher displays other weaknesses - a longer key is not necessarily indicative of a stronger cipher.

How much data is required to mount an exhaustive key search attack?

If the cryptanalyst only has access to the ciphertext, then the amount of data required will depend on the redundancy in the message source — in effect, it depends on how easy it is to recognize when the decrypted ciphertext is correct.

With known plaintext, the task is much easier, requiring one pair of known plaintext and ciphertext, though it is possible, depending on the block-size and the length of the key that additional pairs might occasionally be required to distinguish any false alarms that might occur.

At first sight, it might appear that a ciphertext only attack offers little advantage over a known plaintext attack. However, it does provide the opportunity for an opponent to pre-compute a look-up table in which some fixed plaintext is encrypted under every possible key. Then the cryptanalyst need only request the encryption of that one particular plaintext with which

the table was compiled, and then look-up the key that was actually used.

While the computation and storage of such a table is a huge effort for any reasonably-sized key, the table need only be computed, and stored, once.

In fact, Hellman has shown [58] that there is a trade-off in the requirements of time and memory. Exhaustive key search with no storage and the precomputation of a look-up table are essentially solutions which lie at the two extremes of the time and memory trade-off. It is possible to trade an increase in the time required to search over part of the keyspace with possible savings in the storage required to compute part of a look-up table.

3.4 Differential cryptanalysis

Much of the recent advancement in the art of cryptanalysis can be credited to the work of Biham and Shamir. In particular the development of *differential cryptanalysis* [17] has had a quite revolutionary effect on the design of block ciphers.

Differential cryptanalysis is a chosen plaintext attack on iterative block ciphers. It has been particularly successful when applied to more recent block ciphers such as Khafre [97], REDOC-II [31], FEAL [139] and LOKI [24], but DES has, by contrast, fared much better. The reasons and implications of this success will be considered further in Section 4.4. There is considerable evidence to support the claim by Coppersmith [29] that the IBM team designing DES knew about differential cryptanalysis in the early 1970's but that the power of the technique was recognized and it was classified by the authorities.

Whatever the private history of differential cryptanalysis, the first public appearance of differential-type techniques came when Murphy published a chosen ciphertext attack on FEAL requiring 20 chosen plaintexts [102]. These techniques were extended by Biham and Shamir with increasing effect on various cryptosystems [15, 16] culminating in attacks on DES [14, 18].

Differential cryptanalysis succeeds in obtaining information about the key from individual rounds of the iterated cipher. Recall that the main design philosophy behind an iterated cipher was that a weak round function, with several desirable attributes, is repeated in the hope that the resulting cipher is secure.

By choosing the plaintext pairs that are encrypted, the difference between the inputs to the final round of the cipher can be predicted with a certain probability. The particular definition of 'difference' depends on the block cipher under attack - in DES it is merely the bitwise exclusive-or (xor) of the pair. The aim is that the predicted difference between the pair enter-

ing the final round can be used together with the difference in the ciphertext pair (which is observed and hence known) to deduce information about the subkey used in the final round of the cipher. Full details of the attacks conducted by Biham and Shamir can be found in [17].

We note that the probability that a chosen plaintext pair will provide the desired difference at the end of round $n-1$ can be expected to decrease as n increases. Thus attacks on reduced versions of iterated ciphers can be devastatingly effective whereas the full version of the same ciphers can have sufficiently many rounds to deter a cryptanalyst from embarking on such an attack.

The optimal choice for the difference in the plaintext pairs is calculated by investigating *characteristics*; these specify the expected differences for each round of the cipher. A characteristic has some associated probability which is based on the likelihood that the expected difference in the last round (specified by the characteristic) actually occurs given that the specified difference in the first round is used.

While the power of differential cryptanalysis is clear to all, and it has been used with remarkable success in the cryptanalysis of many cryptosystems, DES has only yielded to the extent that an attack on the full 16 rounds of DES requires about 2^{47} chosen plaintexts [18]. Biham and Shamir are the first to acknowledge that though the work effort in a differential cryptanalytic attack on DES is considerably less than that required for an exhaustive key search, the need to accumulate such vast quantities of chosen plaintext means that differential cryptanalysis cannot, in its present form, be considered a threat to DES.

New cryptanalytic attacks provide impetus to the search for new design criteria and differential cryptanalysis depends for its success on the design of the function f which lies at the heart of each round in the iterative cipher.

In the case of DES it is clear that consideration of the iterated f function can be centered on the design of the S-boxes. This design is crucial to the limited success of differential cryptanalysis. The study of S-boxes has become a particularly fruitful field of research and more details will be provided in Section 4.4.

Interesting theoretic work by Lai, Massey and Murphy [84] on the applicability of differential attacks to certain types of iterated ciphers, so-called Markov ciphers, shows how the success of differential cryptanalysis can be limited when certain features are incorporated into the design of the block cipher. Their work introduced the idea of *differentials* which are a broader version of characteristics; only the input and output differences are specified while the differences at intermediate rounds are not considered.

Nyberg and Knudsen [116, 117] make note of the duality between these concepts. They point out that to make a successful differential cryptanalytic attack on a DES-like iterated cipher, the existence of good characteristics is sufficient. To prove the resistance of a cipher against differential attacks however, differentials must be considered and there should be no differential with a high enough probability allowing the attack to succeed. More casually: to attack the cipher any single way through the cipher will do, but to protect the cipher every way through the cipher must be considered and shown to be sufficiently difficult.

There are some variants to the basic differential attack. As an analogy to differentiation in calculus, *higher-order differential attacks* have been considered by Lai [80]. While Knudsen has demonstrated that ciphers can be constructed that are vulnerable to high-order differential attack while being resistant to conventional differential attack [78], it appears that this style of attack might be limited when used on more sophisticated ciphers.

Meanwhile, Knudsen has introduced the idea of *truncated differentials* (formally called *partial differentials*) [71]. Here the cryptanalyst attempts to predict the behavior of part of the difference but not the entire quantity. Again, ciphers can be constructed which are vulnerable to this type of attack [78] but it is likely that this attack, or close variants, will only be useful for ciphers that have a relatively small number of rounds.

3.5 Linear cryptanalysis

Linear cryptanalysis is an attack on iterated ciphers that bears more than a passing resemblance to the differential cryptanalytic attacks of Biham and Shamir. Significantly, however, it differs from these attacks in that it requires only known plaintext rather than chosen plaintext and can, in general, be considered a more practical threat to cryptosystems than differential cryptanalysis.

Introduced by Matsui and Yamagishi [95] at Eurocrypt'92, linear cryptanalysis was used against the FEAL cipher [139]. This attack was then refined by Matsui and used on DES with very exciting results [92, 91]. See Section 4.4 for more details.

The aim of a linear cryptanalytic attack on a cipher is to find an effective linear expression connecting some bits of the plaintext, some bits of the output at round r and some key bits. This linear expression is valid over r rounds and when the probability it holds is not $\frac{1}{2}$ (that is, there is some bias) then by taking sufficiently many plaintext/ciphertext pairs, the correct value of the combination of key bits can be identified, thus providing one

bit of key information.

The greater the bias, the fewer the number of plaintext/ciphertext pairs that need to be taken before the correct key bit value can be deduced. Unfortunately, the greater the number of rounds r spanned by the best linear approximation, the smaller the bias. Some clever techniques can be used to help in a linear cryptanalytic attack. Analyzing other expressions will yield information about other key bits, and counters can be used to search over a small subset of key bits; this potentially allows the cryptanalyst to predict the action of some round and to use a shorter linear approximation with an improved, exploitable bias [91].

As linear cryptanalysis is a new technique, it is not clear quite how successful it will prove to be or even how much further it can be adapted to improve the existing attacks. As it stands, it appears that Matsui's results for DES [91] are near to optimal and it is not clear how linear cryptanalysis can be used to improve upon them.

As an interesting twist to the known plaintext attack, Matsui points out that if the plaintext bits involved in the linear approximation are known to take on certain values with a known probability, perhaps as a result of the plaintext being an ASCII encoded version of English, then the attack can be converted into a ciphertext only attack requiring a vast amount of ciphertext, but still less than the theoretically important 2^{56} ciphertexts.

Like differential cryptanalysis, the success of linear cryptanalysis is dependent on the function f that is used at each step of the iterated cipher. Because of its application to DES, much research is being conducted into the design of S-boxes; see Section 4.6.

There have been various attempts to apply linear cryptanalysis to other ciphers [68, 121, 122, 143, 145]. And while there have also been results which provide a more satisfying theoretical framework for assessing linear cryptanalysis [34], and other results consider its applicability in the average case [119], it is still not clear how to formulate some generic design technique which will protect a specific block cipher against linear cryptanalytic attack.

Knudsen [76] has made some progress in this area by considering some necessary requirements for what have been termed *practically secure* Feistel ciphers; Feistel ciphers which are resistant to differential and linear cryptanalytic attacks. Nyberg has also presented results [114] which introduce an analogy of the differentials in differential cryptanalysis to linear cryptanalysis.

However, the situation is quite complicated when one considers some of the possible variants to this attack. Kaliski and Robshaw [62, 63] have considered the use of more than one linear approximation on the same set

of data. This can allow for a decrease in the amount of data required for a successful attack and has been shown to effective in an attack on FEAL-8 [63]. Meanwhile, Harpes et al. [56] have considered a generalization of linear approximations, *input-output sums*, and their general applicability. Yet another intriguing development is that of *linear-differential* cryptanalysis [85] which provides a fusion of the techniques used in both linear and differential cryptanalysis.

Interestingly several researchers have highlighted a duality between linear and differential cryptanalysis [103]. This duality is also exhibited during the design of techniques to construct good differential characteristics and linear approximations [94, 11] and also in attempts to quantify the protection offered by various Boolean functions against both differential and linear cryptanalysis [26].

3.6 Other considerations

It may well be the case that the structure of the block cipher gives rise to certain surprising features. One example would be the complementation property present in DES whereby $\overline{E_k(m)} = E_{\overline{k}}(\overline{m})$ with $\overline{k}$ denoting the binary complement of k.

Other properties might include a class of weak keys, which for DES have been defined as keys k where $E_k(E_k(m)) = m$ or pairs of semi-weak keys, k_1 and k_2 for which $E_{k_1}(E_{k_2}(m)) = m$. Very often these properties are so rare that they have little practical impact when the cipher is used for encryption. However, if the block cipher is used as the basis for a hash function then they can become vitally important [125].

But it is not clear how every instance of such a property can be excluded from occurring, though Massey did address this issue in his design for SAFER K-64 [89]. It seems that since each block cipher has its own individual architecture, then each block cipher will have its own individual anomalies in practice.

One interesting cryptanalytic device was introduced by Biham at Eurocrypt'93 and is called a *chosen key attack* [8]. More details will be given in Section 4.3.4 but it appears to be one of the first general attacks to concentrate on the key scheduling part of the iterated block cipher and exploits the following design feature. Just as the principle of diffusion suggests that the influence of plaintext bits be distributed over all the ciphertext bits, the influence of the key bits should be spread over all the ciphertext bits. To do this with an iterated cipher involves a sophisticated key scheduling mechanism which allows the key information to be thoroughly mixed in during

the encryption procedure. Biham [8] looks at the key scheduling algorithms in several algorithms and obtains some interesting results. Similar and related work on key scheduling has been pursued independently by Knudsen [73, 75].

4 DES

4.1 Background

In May 1973 the National Bureau of Standards (NBS) invited submissions for an encryption algorithm that could be economically and widely used for the storage and transmission of unclassified data. In response to a second call for submissions in August 1974 an algorithm was proposed by IBM. Some details of this work appear in a report by Coppersmith [29]. After a period of review by the U.S. government, particularly the National Security Agency (NSA), the algorithm was presented for public comment in March 1975. This algorithm became the Data Encryption Standard (DES) and was endorsed by the U.S. government in 1977 [105].

DES was initially recommended with provisions for a review every five years. It was reaffirmed in 1983 and 1988 [107] and again on December 3, 1993 [108]. DES has been the subject of intense cryptanalysis since its publication, and despite many interesting results and much hard work, it is only within the last couple of years that the design criteria of DES have begun to be understood. It is sometimes said that DES trained a generation of cryptographers; certainly the publication of DES helped bring cryptography into the public domain.

At present, DES is the most widely used and trusted symmetric cryptosystem. The export of DES from the U.S. either in hardware or software is particularly rare except to U.S. subsidiaries and banks though of course full published details of the algorithm are widely available overseas. However DES is reaching the end of its useful life and an alternative is being sought. In the absence of any block cipher providing a similar degree of confidence, much research is currently underway concerning modifications to DES in the hope that this might provide a useful cipher for the bridging period before the appearance of another widely trusted block cipher.

4.2 Description

DES is a 64-bit block cipher that uses an effective key of 56 bits. It is an iterated Feistel-type cipher with 16 rounds and can be implemented in

hardware, often at encryption rates of around 20 Mbits/sec or higher; or in software, at encryption rates perhaps around 400 or 500 Kbits/sec with a wide variation possible depending on the platform and the implementation [126, 133]. Very specialized high performance implementations also exist, with a recent chip using gallium arsenide technology achieving an encryption and decryption rate of one Gbit/sec [49]. Since DES is a Feistel-type cipher the same algorithm can be used for encryption as well as for decryption, provided the order in which the subkeys are used is reversed.

The round function, operating on a 32-bit quantity, first expands these 32 bits into 48 bits by bit-repetition. Then 48 bits of key information are combined with this expanded data and the result is used as input to eight different S-boxes. Each S-box takes six bits as input and gives four bits as output. The 32 bits of output from the S-boxes are then permuted and presented as output from the f function.

DES uses a 56-bit key which is supplied by the user as a 64-bit quantity. The expansion from 56 to 64 bits appends a single bit to every block of seven in the user-supplied key so that the number of ones in each eight-bit block is odd.

More details on the workings of DES, and on the internal specifications can be obtained from FIPS Publication 46 [105]. Full details of the algorithm are also published in the following textbooks (among many others) [6, 79, 142].

4.3 Controversy

From the start, DES was embroiled in controversy. Criticism was usually due to one of two factors, either the size of the key or the design of the S-boxes. The latter is a particularly essential concern since the S-boxes provide the non-linearity to the block cipher.

4.3.1 Key size

Taking the size of the key first, it remains open to speculation why a key length of 56 bits was chosen when very little change needs to be made to the algorithm to accommodate a 64-bit key. While arguments will continue over various explanations which have been offered, most agree that the key seems to be somewhat shorter than it needs to be.

In 1977 Diffie and Hellman published an article [48] claiming that a machine could be built for about $20 million which would find a DES key by exhaustive search in around 12 hours of computation time.

Since then others have repeated the analysis and the most recent in depth analysis conducted by Wiener [148, 149] has concluded that a machine could be built today for about $1 million that would find a DES key in an average time of about 3.5 hours.

4.3.2 S-boxes

It was soon established that the S-boxes had characteristics implying they had not been chosen at random. Hellman et al. [59] made a thorough investigation of the S-boxes in DES and came up with some surprising results. Their work was commissioned by a body called the Lexar Corporation and is sometimes referred to as the Lexar report [87]. Additionally, the results obtained by Hellman et al. were published as a Stanford University technical report [59] and since the technical report is far more readily available, this is how we shall refer to their work.

The most intriguing findings concerned the fourth S-box, often denoted as $S4$, which in some sense appears to be 75% redundant. The action of the S-boxes in DES are defined in terms of four permutations; one permutation is selected according to the value of two of the input bits to the S-box. The four-bit output of the S-box is then selected by the action of this particular permutation on the remaining input to the S-box.

Hellman et al. discovered that three of the permutations used in the definition of $S4$ are easily expressible in terms of the first. Meanwhile other S-boxes seem to have more features in common with linear transformations than would be expected if they had been chosen at random. Other interesting issues involved affine approximations to the S-boxes. An affine transformation is more complex than a linear transformation but it is still sufficiently simple to be of particular interest in the cryptanalysis of a cipher. Once again $S4$ stands out as possessing an unusual number of affine approximations.

Other features that were discovered include the fact that complementing one of the input bits to an S-box results in at least two of the output bits being complemented. This feature we now know was one of the precautions taken against differential cryptanalysis [29].

Another active area of research (which continues today) considered the possibility that DES contained some cleverly disguised *trapdoor* that would allow effortless decryption by those with access to the design criteria.

Shamir [137] pointed out that the S-boxes appeared to be somewhat imbalanced, though he also remarked that it wasn't clear how such a feature could lead to either a cryptanalytic attack or to the implementation of a

trapdoor. In fact Shamir had stumbled upon the very feature (a peculiarly high correlation between some combination of the input and output bits of the fifth S-box, S5) that Matsui was to exploit nearly eight years later.

Over the years, as more evidence accumulated it was clear that several critical decisions had been made either during the design of the S-boxes at IBM, or, as some allege, during the review carried out by the NSA. The conclusion of the report by Hellman et al. begins with the following paragraph:

> Structures have been found in DES that were undoubtedly inserted to strengthen the system against certain types of attack. Structures have also been found that appear to weaken the system.

On the other hand, as a warning to cryptographers against an over-reliance on strange patterns Hellman et al. [59] also remark that " ... the problem [of the search for structure in S-boxes] is complicated by the ability of the human mind to find apparent structure in random data, which is really not structure at all."

Two workshops were held in the late 1970's by the NBS [104, 19] with the result that DES was pronounced satisfactory as a cryptographic standard despite the objections of the time, and it was widely adopted by government and industry alike.

Though the design criteria for the S-boxes in DES still remain classified, some of the principles used in their design were released at the second workshop [19]. Brickell, Moore and Purtill [22] later reported that by generating S-boxes according to the publicly known design principles, the observed characteristics of the S-boxes in DES could be tied to the choice of design criteria. They also suggested that the same design criteria provided some explanation for the issues raised by Shamir.

Other work into DES continued throughout the 1980's. Davio et al. provided a paper [40] reviewing different approaches that had been tried in the cryptanalysis of DES. These methods include the consideration of S-box weaknesses, finding equivalent formulations for DES and even finding a formal expression for the action of the S-boxes so that the full 16-round DES can be reduced to the analysis of a set of Boolean expressions. This was first suggested by Hellman et al. [59] and was pursued by Schaumuller-Bichl [134] who concluded that this technique was impractical.

DES withstood all these early efforts. With hindsight it is interesting to see how close many researchers came to discovering the technique of linear cryptanalysis which today provides the best known attack on DES.

It is only recently that more details on the design of the S-boxes have been made public. A report by Coppersmith [29] states that the designers of DES were aware of differential cryptanalysis (see Section 3.4) and that steps were taken to hinder a differential attack. Some of these additional steps result in previously unnoticed features of the S-boxes and the permutation P used in the round function f. There is, however, a growing body of evidence that suggests that very basic changes to the design of DES could have offered better protection against linear cryptanalysis while still providing adequate protection against differential cryptanalysis [94]. More details are provided in Section 5.1.

4.3.3 Algebraic structure

In the mid-1980's a series of experiments were conducted by Kaliski et al. in an attempt to determine whether DES is susceptible to any readily identifiable algebraic weakness [65, 66].

Two major issues were addressed: whether DES is *closed* and whether DES is *pure*. If DES were closed then for each pair of keys k_1 and k_2 there would be a third key k_3 so that $E_{k_1}(E_{k_2}(m)) = E_{k_3}(m)$ for every plaintext block m. (In such a case the set of encryption transformations would form a group.)

Such a discovery would imply that the use of multiple encryption (see Section 12) would offer the same security as single encryption. Even worse, DES would also be susceptible to a known-plaintext attack that would require 2^{28} steps on average.

If DES were pure then for any keys i, j and k there would exist a fourth key l such that $E_i(D_j(E_k(m))) = E_l(m)$ for every message m. If this were the case then the particular multiple encryption schemes to be described in Section 12 would, again, provide no additional security above single encryption. A cryptosystem that is closed is necessarily pure, but the converse does not hold.

By following the work of Coppersmith and Grossman [30] it was known that DES could theoretically generate $\mathcal{A}_{64}$ (the so-called alternating group on 2^{64} elements). However a lower bound on the size of the group that can be generated is far more important.

Kaliski et al. ran several sets of cycling experiments which showed interesting properties [28, 65, 66] and gave statistical evidence that DES is neither closed nor pure. Moore and Simmons [101] investigated further properties of DES with cycling experiments. By taking the least common multiples of the cycles that were found, a lower bound on the order of the group generated by

DES can be established. Proof that DES is not closed was finally provided by Campbell and Wiener [25] who made use of some work by Coppersmith.

4.3.4 Weak keys

As well as considering the security of the S-boxes in DES, Hellman et al. [59] made the first observations of the complementation property in DES. Denoting the binary bitwise complement of a binary string s by $\overline{s}$, we have that

$$E_{\overline{k}}(\overline{m}) = \overline{E_k(m)}.$$

In truth, though an interesting property, its only known impact on the security of DES is that the time taken for an exhaustive search over the key space can be reduced by a factor of two [59] in certain cases.

For this attack the cryptanalyst needs to know the ciphertext c_1 and c_2 corresponding to two plaintexts m and $\overline{m}$, one of which may need to be a chosen plaintext. By encrypting m using all keys beginning with 0 and comparing the result with the two ciphertexts, the key can be obtained in 2^{55} operations instead of the usual 2^{56}. This saving in work effort is due to the fact that a key and its complement can be tested at the same time for the cost of only one more comparison; this is practically free when compared to the time required for a DES operation. As remarked in Section 3.1 arguments can be employed to make this into a known plaintext attack.

The next observations were made about the key scheduling algorithm. In the algorithm, two permutations PC–1 and PC–2 are used to select the bits for each subkey. By choosing a highly regular key it is possible to ensure that the subkeys have various properties.

The most important among these are the four keys, presented below in hexadecimal notation with parity bits, for which encryption is identical to decryption, hence $E_k(E_k(m)) = m$.

weak keys in DES
0101010101010101
FEFEFEFEFEFEFEFE
1F1F1F1FE0E0E0E0
E0E0E0E01F1F1F1F

In addition there are six pairs of keys k_1 and k_2 such that $E_{k_1}(E_{k_2}(m)) = m$ - these are termed semi-weak keys.

semi-weak keys in DES	
k_1	k_2
E001E001F101F101	01E001E001F101F1
FE1FFE1FFE0EFE0E	1FFE1FFE0EFE0EFE
E01FE01FF10EF10E	1FE01FE00EF10EF1
01FE01FE01FE01FE	FE01FE01FE01FE01
011F011F010E010E	1F011F010E010E01
E0FEE0FEF1FEF1FE	FEE0FEE0FEF1FEF1

Again, the weak and semi-weak keys seem to be little more than an interesting phenomenon. They certainly do not constitute a practical weakness in DES. As Davis [41] says "Keys should be random; keys should be independent; keys should never have any part predetermined. Failure to follow these rules or compromises in their achievement will compromise the security equivalently."

The idea of *key-clustering* was first described by Hellman et al. [59] in the following way - "If the same plaintext enciphered under two similar keys yields two similar ciphertext blocks, one could attempt to find a key near the correct one and then perform a local search to determine it exactly." Desmedt et al. [46] concentrated closely on the behavior of the algorithm used to obtain the subkeys for each round. Over a few rounds, they were able to take advantage of the many symmetries and predictable patterns that can be traced and they asserted that "if DES had only a few rounds it would be a weak system." Unfortunately the problem becomes impossibly complicated with additional rounds and Desmedt et al. were unable to make much progress towards results on a full-round version of DES. Like much else with DES, the designers appear to have the upper hand.

There is an attack due to Biham [8] that was presented at Eurocrypt '93 and is termed a *chosen key attack*. The attack does not consist of asking for a particular key to be used for encryption and then making a correct guess for its value; such an attack would be particularly successful but would (hopefully) be impossible to mount in practice!

Instead the chosen key attack allows the cryptanalyst to choose the relation between any two keys that are used to encrypt either known or chosen plaintext. The cryptanalyst then proceeds to derive both keys. The main observation that this attack relies on is that at each round of the iterated cipher we can imagine that there is some algorithm for choosing the subkey that is used. If the same algorithm is used for every round then the chosen keys can be specified in such a way that the subkeys generated for one of the keys are "staggered" by one round when compared to the subkeys generated

by the other key. That is: the subkey in round one for one key occurs in round two for the other, and so on. Biham shows how such a situation might be exploited by a cryptanalyst.

Clearly it is an attack that is perhaps of little practical use, but it is interesting for three reasons. First, it is one of the first general cryptanalytic attacks to focus on the key scheduling algorithm in DES. Second, the attack is not dependent on the number of rounds that are used in the cipher. An iterative cipher that is weak to such an attack cannot be strengthened by adding more rounds. Third, DES appears to be impervious to such an attack. The reason for this immunity is that there is a "stutter" in the key scheduling algorithm: sometimes the key registers are rotated by one bit between rounds and sometimes by two bits. As a consequence the subkeys for successive rounds are not generated in exactly the same way and this is sufficient to thwart the attack as outlined by Ben-Aroya and Biham.

Knudsen [71] has pursued similar work in the analysis of the key schedules used in DES, LOKI and LOKI91 [23]. Knudsen [75] introduces the idea of "potentially weak" keys and their existence relies on the fact that most of the subkeys generated using one key might be identical, though off-set by a round, to most of the subkeys generated using another key. However, it is not clear at present how this feature might be exploited in an attack on the underlying block cipher.

4.4 Status

DES is still secure, but it is now at the end of its useful life. It will, however, continue to be used in some other mode, perhaps in some form of triple encryption (see Section 12) until an alternative block cipher can be found.

Perhaps the most important recent result is that of Wiener [148, 149] (previously mentioned in Section 4.3.1). It is estimated that for $1 million dollars a machine could be built that by exhaustive search would take on average 3.5 hours to find a DES key.

From a more analytic approach linear cryptanalysis provides a known plaintext attack that in May 1993 required 2^{47} plaintexts and differential cryptanalysis has provided an attack which requires 2^{47} chosen plaintexts. For the future, it will be interesting to see what new developments arise from the work of Matsui and to see whether any new developments will finally make a practical break-through and seriously undermine the security of DES. Indeed, subsequent optimizations by Matsui have provided an attack requiring 2^{45} known plaintexts [93], and the figure is gradually being improved. In August of 1994, Matsui presented a paper which reported on

the first experimental cryptanalysis of DES [91]. Matsui conducted an experiment during which a DES key was successfully obtained using a linear cryptanalytic attack with only 2^{43} known plaintext/ciphertext pairs. The experiment took 50 days to complete on 12 HP9735 workstations and it is the first published report that anyone has ever obtained a DES key cryptanalytically.

It is clear that many refinements and extensions are being pursued in the area of linear cryptanalysis and it appears that the ultimate fate of DES with respect to linear cryptanalysis will perhaps be decided in the next couple of years. Whatever happens, it is undeniably the case that DES was very well designed and it has exceeded original expectations [57] by still being widely considered as a secure cipher even 20 years after its design and first proposal.

4.5 Reduced round versions

As a starting point for the analysis of DES many cryptanalysts consider reduced round versions of DES. Though many attacks seem to work well on such versions they soon become hopelessly entangled or woefully inefficient when extended to anywhere near the full number of rounds required for DES. However, they do provide a means of verifying intuition about the algorithm, as well as a means for practical implementation of proposed attacks.

Some attacks, such as differential and linear cryptanalysis, theoretically extend over any number of rounds but as the number of rounds is increased the amount of data required for successful cryptanalysis increases. As a consequence, while reduced round versions require a small amount of data, the full round versions require (at present) too much data to be practical. Other attacks, particularly some of those that were proposed in the 1980's are only effective on reduced round versions because as the number of rounds is increased, they become very unwieldy or even infeasible to mount.

Chaum and Evertse [27] considered the use of what they termed *sequences of linear factors* which would allow a meet-in-the-middle attack to be mounted. Unfortunately it was also shown that their approach, as it stood, would not extend beyond eight rounds.

Biham and Shamir [17] report that in 1987 Davies proposed a known plaintext attack that exploited the way bits are repeated into adjacent S-boxes by the action of the E expansion function. The details of this work can now be found in a recent paper due to Davies and Murphy [37]. The attack requires so much data that it is impractical for many rounds though it perhaps offered the best attack at that time on an eight-round version of

DES, requiring a sample size of 2^{40} known plaintext. An improved version of the Davies attack due to Biham and Biryukov has been presented [13] and allows an attack on the full DES which requires 2^{50} known plaintexts.

Finally we report some recent work by Hellman and Langford [85] incorporating techniques due to Biham and Shamir into a linear cryptanalytic attack. This converts the attack into a chosen plaintext attack but enables a great reduction in the number of chosen plaintext/ciphertext pairs from the 5000 required for the attack due to Biham and Shamir on eight round DES [17] down to 512. Hellman and Langford [85] recover 10 key bits correctly 80% of the time and then perform a reduced exhaustive search to recover the rest of the key though many tricks are possible to obtain other key bits. Unfortunately from the cryptanalyst's perspective, their attack doesn't yet extend successfully to more rounds.

4.6 Research directions

The reliance of DES for its security on the S-boxes has prompted much research into the question of what constitutes a good S-box [42]. The results of Biham and Shamir and those due to Matsui, which are intrinsically dependent on the detailed workings of the S-boxes, have also focused attention on how the attacks of linear and differential cryptanalysis can be hindered.

Much of the work on S-boxes has taken place within the general field of Boolean functions [125]. Predictably though, the abstract study of Boolean functions seems to be a more fruitful area than the study of S-boxes alone.

The Boolean functions used in S-box design must satisfy specific conditions to guarantee sufficient security. It is clear that the functions implemented by the S-boxes should not be linear, nor should they be unnecessarily close to linear [111, 112]. Other considerations include the balance between zeros and ones and the correlation between different combinations of bits.

One property that has received much attention is that of the avalanche of information, that is how many output bits will change on the alteration of some subset of the input bits; recall that this was a property that Desmedt et al. were hoping to exploit [46]. Conditions on Boolean functions which can guarantee a known avalanche of information are now well known though results on the enumeration and construction of such functions are less common. The *strict avalanche criteria (SAC)* is the condition that ensures that exactly half the output bits from an S-box change when exactly one of the input bits changes [147] and there are numerous extensions to this idea [88, 53, 125, 32].

Biham and Shamir remark [17] that some researchers have recommended

choosing S-boxes so that the difference distribution table for each S-box is uniform. This would provide immunity against differential attack by depriving the cryptanalyst of any statistical advantage for a particular round [3, 42, 111]. Biham and Shamir point out however, that variants of DES with such S-boxes turn out to be easier to attack because this new regularity allows differences to be contained within single boxes, rather than propagating on into other S-boxes [17]. O'Conner [118, 119] has established bounds on the strength against differential and linear attack of a randomly chosen S-box which implements a permutation and his work implies that such S-boxes are more likely to be secure if they are larger.

Meanwhile, Nyberg [115] has taken a different approach and analyzed the effect of different transformations in controlling the differential uniformity and the non-linearity of both S-boxes and round functions. Intriguingly, some of the commonly used transformations which protect against differential attack are weak with regards to linear cryptanalysis. Other transformations have exactly the opposite property!

Kim [69] lists five criteria for the construction of S-boxes based on Boolean functions satisfying the strict avalanche criteria including resistance to differential attacks. He generates eight alternative S-boxes for use in a new version of DES called s^2-DES and claims that this version is more resistant to differential cryptanalysis. Knudsen [74] shows that this is not the case. The latest set of alternative S-boxes, s^5-DES, have already been presented by Kim, Lee, Park and Lee [70]. While these S-boxes are intended to provide resistance to all current, major techniques of cryptanalysis, there are still some cryptanalytic techniques that have not been accounted for.

It has become clear that choosing good S-boxes is particularly difficult. Nyberg [110] has made the following distinctions in S-box design by identifying four ways of generating S-boxes. The first is to pick S-boxes randomly; but we know from our experience with DES that S-boxes have to be designed to satisfy certain criteria and they have to be chosen very carefully. Secondly one could choose S-boxes randomly, test them and then throw away those that don't satisfy certain conditions. Both these methods seem to be somewhat inelegant and not particularly satisfying.

The final two options are more appealing. One is termed "man-made," which Nyberg describes as implementation oriented using simple or little mathematics and where S-box generation is usually conducted using more intuitive techniques. The final option, described as "math-made," is to generate S-boxes according to mathematical principles. By using mathematical constructions, S-boxes can be constructed which offer proven security against linear and differential cryptanalysis together with good diffusive properties.

As an example of this approach one could draw attention to a recent proposal by Nyberg [113] which perfectly embodies this "math-made" approach.

It is perhaps unlikely that any technique will be sufficient on its own to generate cryptographically secure S-boxes. Perhaps the best approach is to use a judicious mix of both "math-made" and "man-made" techniques. Preneel [125], writing about the choice of Boolean functions for S-boxes, says that " ... theoretically interesting criteria are not sufficient ... " and further contends that " ... ad hoc design criteria are required."

5 DES variants and Lucifer

5.1 DES variants

It is of interest to see how DES performs when certain features are changed. Particular interest is often focused on the S-boxes though other alterations to both the detail and the structure of DES can be considered.

Regarding the S-boxes, randomly chosen S-boxes are very unlikely to be secure. It is now apparent that many criteria were used in the choice of the S-boxes (see Section 4.3.2) and it is exceedingly unlikely that an S-box, or combination of S-boxes, with security equal to or better than those in DES will be found at random. Biham and Shamir in [17] looked at many aspects of the round function with respect to differential cryptanalysis and discovered that even minor changes, such as the ordering of the S-boxes, can seriously weaken DES. Changing the ordering in time of other operations like the E expansion seems also to introduce deficiencies in security.

Most interestingly, it appears that one cannot say the same when considering the protection offered by DES variants against linear cryptanalysis. Matsui [94] has searched for good linear approximations of all $40,320$ variants of DES obtained by permuting the order of the S-boxes. By exhaustively testing for linear approximations of two types, Matsui concludes that the order of S-boxes finally chosen for DES lies among the 9–16% that offer least protection to linear cryptanalysis.

For differential cryptanalysis the chosen order of the S-boxes in DES gives a cipher which lies among the 2.5% that offer the most protection. There is however, no trade-off between protecting against differential cryptanalysis and protecting against linear cryptanalysis. Matsui demonstrates that there are S-box orderings which would provide adequate protection against both linear and differential cryptanalysis simultaneously.

This observation has an interesting side-effect. Namely, that if one is in a position to change the order of the S-boxes used in some implementation

of DES, then one can arrive at an improved version of DES without the need for any additional software or hardware [12].

Some consideration has been given to the idea of using independent and randomly chosen subkeys in each round. The effect of this change on the success of differential cryptanalysis is, in general, minimal. In fact, Biham and Shamir [17] often use this assumption in the analysis of the effectiveness of their attacks. They claim that experimental results using the genuine DES key scheduling differ only slightly from those obtained assuming independent subkeys. Thus, even with a 768-bit key (16 48-bit subkeys, one for each round) DES would be little more secure against the basic differential cryptanalytic attack than it is now with a 56-bit key. Biham estimates that 2^{61} chosen plaintexts are required to attack DES with independent subkeys [10]. Note, however, that the most effective differential-based attack on DES requires 2^{47} chosen ciphertexts and this form of the attack would not be successful against a version of DES with independent round keys since it relies on details within the key-schedule.

Independent subkeys would, however, have an effect on the success of the basic linear cryptanalytic attacks. As for differential cryptanalysis, the use of independent subkeys is assumed in the theoretical development of the attack, and this is considered a close approximation to the truth. Basic techniques derive 26 bits from the 56 in the user-provided key and so it appears that the use of independent subkeys in reality would ensure that the cryptanalyst derives only 26 subkey bits out of 768.

Instead, the cryptanalyst can use additional techniques and eventually gain enough information to remove a round from this version of DES with independent subkeys. Once this is done, the data can be reused to attack the remaining rounds. The data requirements for this attack are quoted by Biham [10] as being 2^{60} known plaintexts.

G-DES was proposed by Schaumuller-Bichl [134] as an attempt to improve the speed performance of DES by using a more sophisticated architecture to allow larger block sizes without a corresponding increase in the amount of computation. It was claimed that the security of G-DES could be related to the security of DES, and that it was no less secure. Biham and Shamir show [17] that this is not the case, and that G-DES with the recommended parameter sizes can be easily broken. While several parameters in the description of G-DES can be changed, Biham and Shamir conclude that "any G-DES which is faster than DES is also less secure"[17]. G-DES provides a good example of a cipher that is built using trusted techniques as basic building blocks and yet ends up being less secure.

Finally we consider another variant of DES that has been proposed, DES-

X [67]. In this variant, DES is used exactly as it appears in the literature, but the input to the algorithm is exclusive-ored with 64 bits of key material (which should be independent of that used in the DES encryption) and the output is exclusive-ored with either the same, or perhaps a different set of 64 bits of secret key material. While the resistance of the resultant cipher to linear and differential attack is no greater than that of DES with independent subkeys, the strength against exhaustive search is dramatically improved [67].

5.2 Lucifer

Lucifer [141] is often mentioned as the starting point for the development of DES. Indeed there are many similarities between the two cryptosystems, but there are also significant differences which make a look at Lucifer interesting in its own right. There appear to be two variants of Lucifer in the open literature [52, 141]; Biham and Ben-Aroya describe the version outlined by Sorkin [141] as the final variant of the Lucifer project. Biham and Shamir [17, 16] have claimed that this variant is in fact weaker, with respect to differential cryptanalysis, than the one described by Feistel [52]. We note, however, that some assumptions to fill the gaps left by Feistel's description were made during this analysis.

There is no doubt that Lucifer is a block cipher that operates on blocks of size 128 bits and uses a key of length 128 bits. It is a Feistel-type cipher, using 16 rounds like DES, though the round function f and the key scheduling are somewhat simpler.

The f function introduced in the description of the Feistel cipher (Section 2.3) can be represented in Lucifer by the action of eight so-called T boxes. Each T box outputs eight bits while taking nine bits as input. The T boxes are in fact a convenient representation of the action of two alternative S boxes which map four bit inputs to four bit outputs, the extra bit in the T box description choosing which S box is used. The round is ended with a permutation P.

To some people, Lucifer might appear to be more secure than DES both because of the lack of published results on the cryptanalysis of Lucifer and because the key for Lucifer is 128 bits in length. However previous results by Biham and Shamir have cast doubt on Lucifer [16] while more recent results due to Ben-Aroya and Biham [7] imply that more than half the possible keys are insecure and can be found using a differential-type attack with complexity 2^{36}.

The conclusion Ben-Aroya and Biham drew from this research is that

the changes made to Lucifer in the development of DES were improvements and that DES can certainly be considered to be more secure than Lucifer despite the smaller key size.

6 IDEA

6.1 Introduction

The International Data Encryption Algorithm (IDEA) first appeared as the Proposed Encryption Standard (PES) at Eurocrypt'91 and was designed by Lai and Massey [83, 82]. It is an iterative cipher that operates on 64-bit blocks and uses a 128-bit key. The aim was to design a block cipher that could be efficiently implemented in both hardware and software , unlike DES which is primarily suitable for hardware encryption. It is claimed [83] that a VLSI chip being developed at ETH in Zurich will achieve a data rate of between 45 Mbits/sec and 115 Mbits/sec, depending on the architecture. In fact a speed of 166 Mbits/sec has been achieved using a specially designed chip [81]. In software it appears to run at about the speed of DES which is perhaps disappointingly slow [124].

At Eurocrypt'92 the Improved Proposed Encryption Standard (IPES) was proposed and this has now become IDEA. The changes made to PES in the development of IPES were due to the discovery of differential cryptanalysis by Biham and Shamir [15, 16]. This motivated the development of a new design criterion that can be used to analyze the effectiveness of a differential cryptanalytic attack. The proposal of IPES and the definition of Markov Ciphers, for which the behavior and effect of the differentials used in a differential attack can be modeled and hence quantified, can be found in [84].

It is not clear what the future holds for IDEA. There has been no rush to adopt the cipher by implementers, perhaps because they are waiting to see how well the algorithm fares during the coming years at the hands of cryptanalysts. However the cryptographic software package Pretty Good Privacy (PGP) supports IDEA. In the following section we shall see that though there are perhaps no major weaknesses yet identified, initial work on the cryptanalysis of IDEA has made some progress despite the impressive theoretical foundations on which the cipher is based.

6.2 Design

The designers used an increasingly common approach [129, 109] to attain security: mixing different arithmetic operations so that no single framework can be used to fully analyze the round function used in IDEA. Operations acting on 16-bit words and those acting in a bitwise fashion have different properties. Combining these operations tends to make cryptanalysis more difficult.

The operations used in IDEA are bitwise exclusive-or, addition modulo 2^{16} and multiplication modulo $2^{16}+1$, with the value 0 corresponding to 2^{16}. The framework of the cipher which establishes how outputs from one operation become the input to following operations, is carefully designed to ensure that the output from one type of operation is used as the input to a different type of operation. Confusion (Section 2.1) is provided by the incompatibility of these operations, while diffusion is obtained by means of a structure termed the *MA-structure* [83].

As previously mentioned, the changes made to PES in the proposal of IDEA (IPES) are very slight, but they make IDEA more resistant to a differential cryptanalytic attack [84].

6.3 Cryptanalysis and status

There are two results of note in the literature at present concerning IDEA. The first is concerned with quite how incompatible the operations actually are in practice, and the second reveals large classes of weak keys.

Meier [96] points out that though it is quite correct to state that the three operations are incompatible, there are instances where the action of the three can be simplified. Thus, a seemingly incompatible relation can be replaced by another that facilitates cryptanalysis and is correct some percentage of the time.

One of the main claims for the security of IDEA is that the three operations do not satisfy what is termed a *distributive law*. Meier refers to a partial distributive law which holds with a certain probability. This effect can be used to analyze the first few rounds of IDEA, but Meier also comments that it is of little use in the analysis of the full eight-round version of IDEA [84].

The second result of note is due to Daemen [35] who reports that a vast number of weak keys can be found. The weak keys allow particular values of the bitwise exclusive-or of a plaintext pair to guarantee a particular value for the bitwise exclusive-or of the ciphertext pair. As an example, for all

keys where only bits 33 – 40 or 92 – 115 may be non-zero, the input xor 0000800080000000 in hexadecimal notation gives rise to the output xor 0000800000008000 with probability 1 [35].

More classes of weak keys are shown in [35] and they represent the first real weaknesses in IDEA. Interestingly, apart from Meier's work [96], they also represent the only weakness in IDEA that has so far been discovered. Whether this is due to a lack of attention to the cryptanalysis of IDEA or to the genuine security that is offered by IDEA is, as yet, unclear.

What is significant is that the design criteria used in the development of IDEA are openly discussed and have a firm theoretical basis. This is most refreshing when compared to the secrecy and the seemingly ad-hoc techniques used in the design of DES — but the field of cryptography is littered with practically insecure designs that rest on theoretically superior bases.

7 SAFER K-64

7.1 Introduction and design

SAFER K-64 (Secure And Fast Encryption Routine with a Key of length 64 bits) was first proposed at the Cambridge Algorithms Workshop in December of 1993 [89]. It is a byte-oriented iterated block cipher designed for efficient implementation in both software and hardware. See [126, 133] for sample encryption speeds.

There has been considerable initial interest in the cipher and Massey has reported the development of a version with a 128-bit key for use in Singapore [90].

It was initially proposed that six rounds would be sufficient for the cipher, but more rounds can be used for greater security. Each round consists of a set of non-linear operations, including two different S-box permutations, that operate in parallel on each of the eight bytes in a block. Two different subkeys of 64 bits are used in each round. They are derived using the key schedule and introduced during this non-linear stage. The second part of each round is a series of linear mixing operations which is termed a *Pseudo-Hadamard Transform*. This provides diffusion across the block.

At the end of the last round, the final iteration of the linear transformation is followed by one further partial round of non-linear transformation using key material.

7.2 Status

Since its publication, a number of researchers have looked at SAFER and there has been some variety in their findings.

SAFER is a Markov cipher [84] and this facilitates analysis with regards to differential cryptanalysis [89]. It is claimed that SAFER K-64 is practically secure against differential cryptanalysis after six rounds and that eight or more rounds render differential cryptanalysis completely ineffective. It is also suggested that three rounds render linear cryptanalysis ineffective [90]. Meanwhile, Harpes et al. have considered applying their generalization of linear cryptanalysis to SAFER K-64 and report no success [56].

Vaudenay has considered variants of SAFER K-64 in which the two S-boxes mentioned earlier are replaced by an alternative permutation and its inverse [145]. With these alterations, for 3.3% of all permutations SAFER K-64 would be vulnerable to attack faster than exhaustive search. The true version of SAFER K-64 however, is not vulnerable to attack.

The only substantial weakness in SAFER K-64 which has so far been published, is in the key schedule and was observed by Knudsen [77]. Essentially the key schedule is not sufficiently complicated and the same bytes in the user-supplied key are used in the same place in every round. While this has not yet led to an attack on SAFER K-64 when used for encryption, it is a substantial weakness when it is used as the basis for a hash function. So much so, in fact, that Massey now recommends that no fewer than 10 rounds of SAFER K-64 be used in these circumstances.

All in all, while SAFER K-64 is rather attractive, it is still early days with regards to cryptanalysis. Unfortunately, there are sufficiently many reported observations and partial results available that even though there is no attack on the cipher, many people are still quite wary. Hopefully there will be some more substantial results, either in favor of it's security or in favor of it's cryptanalysis, soon.

8 RC5

8.1 Introduction and design

RC5 is a new block cipher designed by Rivest for RSA Data Security, Inc. Presented at the Leuven Algorithms Workshop in December of 1994 [131], RC5 is neither confidential nor proprietary.

The cipher is fully parameterized in that the block size, the key size and the number of rounds can all vary. A likely version of RC5 is perhaps

RC5-32/16/10 where the block size is 64 bits, there are 16 rounds and the key is 10 bytes in length.

The algorithm begins by expanding a variable-length key into a set of look-up tables. Then two very simple operations are used repeatedly to mix in the key and transform the data. RC5 is another example of an iterated, non-Feistel cipher.

Due to its elegant simplicity, it will be of no surprise to find that RC5 is very fast — 32 rounds of RC5 with a 64-bit block size has roughly the same encryption speed as SAFER K-64 with six rounds.

However, this isn't the whole story. So far, it is not clear how many rounds are in fact required to provide adequate protection, and this is something that only future analysis will provide.

8.2 Status

RC5 has only been available for public scrutiny for about six months at the time of writing, and this is not sufficient time to provide anywhere near a reasonable review of the cipher in the public domain.

However, some early indications are quite promising. Work by Kaliski and Yin [68] have established the limits of certain differential and linear cryptanalytic attacks on RC5 and the twelve rounds proposed by Rivest do in fact appear to ensure that both attacks are impractical. Interestingly the major primitive used by Rivest, *data-dependent rotations*, appears to be particularly successful in combination with other operations in thwarting linear cryptanalytic attacks.

The stunning simplicity of RC5 has two important benefits for the cryptanalyst. First, analysis of the cipher is greatly simplified and simulations can easily be run on reduced versions of RC5. Hopefully, the true strength of RC5 will become apparent relatively soon. Second, if any weakness is found, then the reason for the weakness should be easy to trace and to avoid in future designs. Lessons will be easily learnt from the eventual strength or weakness of RC5.

So far there are no weaknesses to report. However, despite this promising start it is still too soon to recommend the use of RC5. If there are no more significant results after a year, then perhaps we have found a simple cipher, that is easy to implement and yet offers adequate security.

9 Skipjack

9.1 Introduction

The first mention of Skipjack came in April of 1993 when the White House announced a cryptographic initiative which included the *Clipper chip* [45]. The controversy surrounding the Clipper chip and the other issues raised by this announcement are well known and discussed elsewhere [50, 132].

Despite the fact that Skipjack is a classified algorithm and full details of the algorithm will remain secret, the few details that have emerged suggest that Skipjack is an iterative block cipher, using 32 rounds and a key of length 80 bits [44]. It has been suggested that Skipjack might be structurally similar to DES, but there is considerable uncertainty about this issue. Since the algorithm is classified, it can only be implemented in tamper-proof hardware; the estimated encryption rate is given as 16 Mbits/sec [44].

9.2 Status

Clearly there can be no public domain scrutiny of Skipjack. Instead a group of independent experts were asked to review the security of the algorithm. The first interim report has been published and no reason was found to question the security offered by Skipjack [21]. Indeed, the design and internal evaluation of Skipjack is said to have been in progress since 1987 [21] and taking the considerable success of DES into account, it would be surprising if Skipjack were not very secure.

There are however major obstacles to the widespread acceptance of Skipjack. Much of the early concern about DES was due to the fact that many of the design criteria remained secret, even though the algorithm itself was public. With Skipjack, even the algorithm remains secret so it is to be expected that there will be considerable reluctance by implementers to adopt Skipjack. Since Skipjack remains classified, there can be no software implementations, no customized hardware implementations and there will be at best very few places where the chip carrying Skipjack is manufactured.

Finally we note that some developers, all too aware of the present shortcomings of DES, are keenly awaiting the arrival of a new standard encryption algorithm. Many had hoped that Skipjack would be this long awaited algorithm but the fact the algorithm is classified means they will have to wait much longer before they can plan for the future with much confidence.

10 Other block ciphers

10.1 RC2

RC2 [130] was designed by Rivest for RSA Data Security, Inc. It is a confidential and proprietary cipher and so there are few details that can be readily disclosed.

Like DES it is a 64-bit block cipher but it has a variable key size. One advantage is that the process for granting export approval for RC2 is greatly simplified if the key length is restricted to 40 bits, or 56 bits for foreign subsidiaries and overseas offices of U.S. companies [50].

RC2 is about three times the speed of DES in software and uses one of two operations at each round. The choice of these operations shows some regularity, but RC2 is not an iterative block cipher. This suggests that RC2 offers more protection against differential and linear cryptanalysis than other block ciphers which have relied for their security on copying the design of DES.

10.2 FEAL

The Fast Data Encryption Algorithm (FEAL) was proposed by Shimizu and Miyaguchi at Eurocrypt '87 [139]. It was intended to be very efficient when implemented in software, and was claimed to offer at least as much security as DES. Unfortunately, the security was soon found to be lacking.

FEAL's first incarnation was as a four-round version, and an immediate attack was provided by den Boer [43]. Later Murphy supplied an attack that required only 20 chosen plaintexts [102]. The eight-round version of FEAL did not fare much better. A wide range of attacks [54, 15, 95, 11, 122, 5, 63] have together shown that the eight-round version of FEAL is insecure and they have cast doubts on any of the remaining versions of FEAL that have been proposed.

The remaining versions are FEAL-N with any even number of rounds and FEAL-NX with extended 128-bit keys [99]. Unfortunately even these last two versions are not secure when the number of rounds is less than or equal to 31 [17] and there can be little faith left in the use of any of the FEAL derivatives as a secure block cipher or as a basis for use in a hash function as has also been proposed [100].

10.3 REDOC-II

REDOC-II [31] was billed as a fast confusion/diffusion/arithmetic cryptosystem and was proposed by Cryptech Inc. It has 10 rounds but when first proposed it was suggested that even a one-round or two-round version would be secure. It was claimed that it could encipher at a rate of 800 Kbits/sec on a 20 MHz machine. Cash prizes were offered for theoretical and practical attacks on these versions.

An 80-bit block cipher with an 80-bit key, REDOC-II uses several tables to implement substitutions and permutations and some of the tables are generated using the key. In all, the round function seems to be incredibly complex, undoubtedly the major factor in the claim that the one-round version should be secure.

REDOC-II has not proven to be immune to differential analysis, at least in the reduced round versions [16]. The one-round version is breakable using about 2300 encryptions whilst the four-round version can be attacked using known plaintext to obtain three bytes of internal and key-dependent information [17]. REDOC-II is now rarely mentioned in the literature though Schneier [135] reports that a streamlined version of REDOC-II, known as REDOC-III, has also been proposed.

10.4 LOKI

LOKI [24] was initially proposed in 1989 and is a DES-like iterative cipher that operates on 64-bit blocks and uses a 64-bit key. Its security is based on the use of a large S-box, taking 12 bits and outputting eight, which in turn is based on the use of irreducible polynomials. Further specifications allow LOKI to be used as a hash function.

Weaknesses in LOKI were identified in several places, most notably its vulnerability to differential type attacks and also the presence of fixed points (keys for which the plaintext and ciphertext are equal), equivalent keys (for each key there are 15 that are equivalent) and complementation properties [72]. Together these structural weaknesses reduced the complexity of a chosen plaintext attack and mean that LOKI should not be used as part of a hash function.

Regarding attacks using differential cryptanalysis, results can be quoted in terms of the number of rounds that must be added before a differential attack requires more effort than exhaustive search. The interested reader can find more complete details in [17], but in summary Biham and Shamir cast doubts on the security of LOKI with up to 11 rounds and they report

a result by Kwan which can be used with success on LOKI with up to 14 rounds.

LOKI91 [23] was a redesign of LOKI to offer improved protection against differential-type attacks. Later work has confirmed that these changes are successful in protecting against differential cryptanalysis though it appears that they have resulted in a very slightly weaker cipher with regards to linear cryptanalysis [143].

Knudsen [73] has noted an attack on LOKI91 that exploits a weakness in the key schedule to reduce the work involved in a brute force key search by a factor of four. The key schedules for both the original LOKI89 and LOKI91 were also attacked by Biham [8] using a related keys attack.

10.5 CAST

Designed by Adams and Tavares [2, 4], CAST is a 64-bit Feistel cipher. Instead of employing eight fixed S-boxes which map six bits to four, as we find in DES, CAST uses four S-boxes which are generated as a function of the user-supplied key. These S-boxes map eight bits to 32, and the output of all four S-boxes is exclusive-ored together to produce the output from the round function.

The use of larger S-boxes which are computed as a function of the key has been recommended by several commentators [135]. In fact there is considerable theoretical work available to show that, on average, CAST is likely to be successful in combating differential and linear cryptanalysis if the S-boxes are derived from partially bent functions [3, 4] or if they are randomly generated [60, 86].

Perhaps like Khufu, which we will mentioned in the next section, the lack of a fixed set of S-boxes makes analysis difficult. In addition, it is not clear how much confidence one should place in an average case analysis of the strength of a cipher — perhaps it is preferable to know that every case offers some minimum level of protection. Whatever the reason, CAST is not that widely accepted.

The only significant result on the cryptanalysis of CAST currently available is due to Rijmen and Preneel [128]. Here it is shown that the key schedule in CAST, combined with the use of round functions which do not generate every output value, can lead to unforeseen potential weaknesses.

10.6 Khufu and Khafre

Merkle [97] has designed two related block ciphers, Khufu, suitable for fast bulk encryption of large amounts of data, and Khafre which is more suitable for the encryption of small amounts of data. Both are intended to be fast when implemented in software. They are iterative 64-bit block ciphers with a variable number of rounds and a variable key size, which in Khafre is limited to being a multiple of 64 bits.

Khufu uses the key information to compute S-boxes which take 8 bits as input and give 32 bits as output. The number of S-boxes is dependent on the number of rounds and their computation can be a large operational overhead since this takes place at the time of encryption. However, this overhead is fixed, and this is why Khufu is viewed as being more suitable for bulk encryption. Khafre on the other hand has no initial set-up computation and uses a fixed set of S-boxes.

Merkle mentions [97] that eight-round versions of Khufu are susceptible to a chosen plaintext attack, but states that 16 rounds appears to be sufficient to prevent these attacks and that Khafre is likely to need more rounds than Khufu to achieve equivalent security.

Biham and Shamir [17] provide results on the differential cryptanalysis of Khafre and report that 16-round and 24-round versions can be broken using 1536 and 2^{53} chosen plaintexts respectively.

Until recently, there were no published attacks on Khufu perhaps chiefly because there are no fixed S-boxes. However Gilbert and Chauvaud [55] have devised an attack on the 16-round version of the cipher which requires 2^{43} chosen plaintexts and 2^{43} operations. While this an impractical attack, it is important as it represents the first cryptanalytic breakthrough with regards to Khufu.

Despite the nice features of these ciphers and what is, so far, fairly limited success in cryptanalysis, they have failed to capture much attention outside the research community.

10.7 MMB and 3-WAY

In connection with his work in the analysis of IDEA, Daemen has made a proposal for a different block cipher based on the use of modular multiplication [35]. MMB (Modular Multiplication based Block cipher) is a 128-bit block cipher with a key of length 128 bits.

It is quite closely related to IDEA in that it is based around the devices of modular multiplication and bitwise exclusive-or (xor), but it is claimed

that MMB is immune from the weak keys that are now known to affect IDEA and that it can be more efficiently implemented in both hardware and software.

The theoretical work underpinning MMB ensures that considerable diffusion is obtained in each round independent of the key; Daemen makes the point that the round diffusion in IDEA is to some extent dependent on the particular sub-keys. In addition Daemen is able to prove that differential cryptanalysis of MMB will be no more successful than exhaustive key search. Unfortunately Daemen [33] notes that some weaknesses have been discovered by Biham that rely on the cryptanalyst choosing the difference between keys.

It is too early to decide whether MMB-like approaches to the design of block ciphers will yield great rewards. Intuitively one wonders whether a cryptosystem built using rich arithmetic structures might not also prove susceptible to analysis and attack using those very same structures.

3-WAY [36] which was also designed by Daemen is constructed differently. By using a very simple construction high encryption speeds become possible in hardware and potentially across a wide-variety of software environments. In addition, the theoretical foundations on which the cipher is designed means that the cipher should be resistant to major forms of cryptanalytic attack [34].

There do not appear to be any other results on the analysis of either of these ciphers in the literature.

10.8 Other schemes and further reading

There has been a lively period of algorithm proposal in the literature over the past couple of years. Gradually, cryptanalysts will turn their attention to each proposal and eventually two or three will surface as favorites for proposed use and continuing cryptanalysis.

There is not sufficient space here to go into each proposal individually; instead Schneier's Applied Cryptography [135] gives a substantial overview of the field. In fact one cipher we have not yet mentioned is due to Schneier [136]. Blowfish has some nice performance capabilities [126] and there has still been no significant cryptanalytic results against it despite a competition and monetary reward of $1500 for any substantial results.

The reader interested in seeing a wide variety of different research directions might also find the proceedings of two workshops, *The Cambridge Algorithms Workshop, December 9–11, 1993*, edited by R. Anderson, and *The Leuven Algorithms Workshop, December 14–16, 1994*, edited by B. Preneel,

useful reading. Both proceedings are published by Springer-Verlag.

11 Modes of use

FIPS publication 81 [106] gives details on the different modes of operation for the DES algorithm. These different ways of implementing a block cipher are independent of the actual block cipher being used so we can consider these modes of use without restricting ourselves to specific examples.

11.1 ECB

The Electronic Code Book (ECB) mode is the most obvious way of implementing an n-bit block cipher. The plaintext is split into blocks, each of length n bits, which are then encrypted using the particular encryption algorithm, to give a set of n-bit blocks of ciphertext.

The shortcomings of this approach are obvious. First, any repeated 64-bit block of plaintext is encrypted in exactly the same way under the same key. This allows a cryptanalyst to deduce certain information about the form of the plaintext and perhaps to build up a dictionary of frequently used plaintext/ciphertext pairs.

Additionally a cryptanalyst could remove, insert or replay some block of encrypted data without detection. There is no immediate way for the recipient to know that blocks have been removed or inserted into a message during transmission. Thus the integrity of the message cannot be guaranteed. If the encryption only consists of a single block, perhaps the encryption of a DES key, then these concerns about the ECB mode are not so relevant.

Note that all the modes of use for a block cipher relate back to the underlying block cipher for security against a brute-force attack. However the ECB mode also allows statistical attacks to be mounted. This is something the other modes we describe hinder.

11.2 CBC

A second and more popular mode of use for a block cipher is the Cipher Block Chaining (CBC) mode. Before a plaintext block m_i is encrypted, it is combined using exclusive-or (xor), with the previously calculated ciphertext block so we have $c_i = E_k(m_i \oplus c_{i-1})$. The ciphertext 'previous' to the first plaintext block is called the *initialization value* or *vector* and denoted IV. Though the value of the IV need not be secret, it is recommended that it changes often to prevent a cryptanalyst building up a code book of

encryptions of the first block. The easiest way to ensure the IV changes is to take its value from some incrementing counter.

Clearly the encryption of any particular plaintext block in CBC mode is dependent on all preceding plaintext blocks. Any attempt by a cryptanalyst to insert or remove blocks of plaintext, without affecting any other parts of the plaintext, is significantly hindered. Observe, however, that decryption is given by $m_i = D_k(c_i) \oplus c_{i-1}$ and so we have the following potential problems:

- A last block can be added to a communication encrypted using CBC since this last block will not have any influence on the original content of the message. It is, however, easy enough to structure the content of the message to detect the addition of extra blocks.

- A cryptanalyst can alter a ciphertext block to introduce controlled changes into the following decrypted plaintext block. While the corresponding plaintext block is received in error the following one contains controlled changes. Consequently, once any block is received in error, it might be advisable to discard the whole chain. However, because it may be difficult to determine whether a given block is received in error, some kind of checksum is needed for the entire plaintext.

As we have seen, a block cipher can be used to provide both privacy and authentication; indeed, the last block of a CBC encryption is sometimes used as a Message Authentication Code (MAC) [125].

11.3 CFB and OFB

The two modes of use described in this section provide encryption by considering the plaintext as a stream of b-bit blocks which are then bitwise exclusive-ored with some b bits obtained as output from the block cipher. Thus the operation of the block cipher is like that of a keystream generator in a stream cipher implementation [140].

Analogous to the two modes in the previous section, one mode uses feedback of the ciphertext while the other uses the block cipher to generate a keystream that is totally independent of the ciphertext.

11.3.1 CFB

In Cipher Feedback Mode (CFB) the b bits of the plaintext are exclusive-ored with b bits output from the encryption of some block. Initially this block

is provided by an initialization value IV, but at subsequent encryptions the block is shifted by b bit positions and b bits of the ciphertext are introduced.

Thus we see that the ciphertext is used as feedback, meaning that the insertion and deletion of ciphertext can be readily detected. A ciphertext block containing an error will affect the decryption (using an n-bit block cipher) of the current and following $\lceil \frac{n}{b} \rceil - 1$ ciphertext blocks; after receiving $\lceil \frac{n}{b} \rceil$ correct ciphertext blocks proper decryption will resume. Attempts by a cryptanalyst to make changes to the ciphertext in the hope of making undetectable changes to the plaintext are severely hampered.

Recall that with the CBC mode, the direct combination of ciphertext with the following plaintext block allowed a cryptanalyst to introduce controlled change into some block of received plaintext. With the CFB mode, changes made to a b-bit section of ciphertext give controlled changes in the plaintext obtained for that section, while the following b-bit sections are received in error. Thus it is possible for a cryptanalyst to change the final b-bits of a message without detection. The importance of this attack is usually reduced by not including important information in the last block, or by using the last block to carry a checksum or message authentication code which is difficult to change without detection.

The choice of b is dependent on the implementation but we note that b plaintext bits are processed with each block encryption and so small values of b are not often recommended with DES; $b = 8$ seems to be a reasonable value to use.

Preneel [127] has considered the use of DES in CFB mode, particularly with reference to a meet-in-the-middle attack and a differential cryptanalytic attack. While the security of 16 rounds of DES is not questioned in CFB mode, Preneel does point out that any attempts to improve the performance of DES in CFB mode by using a reduced-round version of DES should be resisted.

11.3.2 OFB

In the Output Feedback Mode (OFB) the block cipher is used as a keystream generator which runs independently of both the plaintext and the ciphertext. After each DES encryption, b bits are taken from the output block and used as a keystream to encrypt the plaintext using the exclusive-or operation. The input block is left shifted by b bits and the b bits that were used as part of the keystream are also used as the left-most b bits of the next input block to DES.

Since the OFB mode does not use cipher feedback, errors in one received

ciphertext b-bit section will not affect the decryption of any other ciphertext b-bit section. Note that the block cipher is used as in the CFB mode except the b bits that are introduced in the plaintext to the block cipher are obtained from the previous output of the block cipher and not from the ciphertext.

It was initially suggested that like the CFB mode of use, the value of b in the OFB mode could be changed according to the implementation without any effect on the security offered by the scheme. Unfortunately this is not the case.

When generating a keystream using a deterministic mechanism with a finite number of states, it is clear that the period (essentially the number of bits generated before the sequence repeats) of the keystream is an important attribute. If this period is too low then plaintext might be encrypted using identical portions of the keystream. When this occurs the xor of the plaintexts and the corresponding ciphertexts will be identical giving a considerable advantage to the cryptanalyst.

Davies and Parkin [38] and Jueneman [61] point out that the value of b when used with an n-bit block cipher in OFB mode is crucially important. When $b = n$ the block cipher encryption acts as a permutation of n-bit values and the average cycle length (and hence the period for the keystream) of a random permutation is around 2^{n-1}.

When $b < n$, however, the encryption provided by the block cipher in OFB mode is an example of a random function mapping n bits to n bits. It can be shown that in this case the average cycle length drops to around $2^{n/2}$. Using DES as a particular instance of this phenomenon we note that a period of 2^{32} is not adequate.

We conclude that in OFB mode an n-bit block cipher should use n bits as input to the encryption function.

11.4 Other modes

Partly in response to the short-comings of DES in OFB mode, Brassard [20] mentions that Diffie has proposed an additional mode of use termed the *counter mode*. This differs from the OFB mode of use in the way the input for the next encryption is determined; instead of taking some of the output from the previous encryption Diffie suggests encrypting the number $i + IV (\mathrm{mod} 2^{64})$ for the i^{th} block where IV is some initialization value.

Preneel [125] describes many other ways of introducing feedback to define new modes of use each having their own characteristics, their own advantages and their own shortcomings. However, the modes of use outlined above are those that are most frequently used and are those for which the security is

most thoroughly assessed.

11.5 Error propagation and synchronization

Error propagation and synchronization are often viewed as different aspects of the same phenomenon. While it may be important for implementational reasons to ensure that errors are more easily detected, for instance if a file is being encrypted prior to storage for a long time, it is also cryptographically important since it gives some feel for how difficult it might be for some cryptanalyst to manipulate the ciphertext in the hope of changing the plaintext in some undetectable way.

In ECB mode each block is encrypted independently of the other, thus a block received in error will have no effect on the decryption of subsequent blocks. For a cryptanalyst this means that changing several bits within a block will affect the decryption of the one block, but no others. Additionally, as we have already commented, it is easy to insert or remove entire blocks of the ciphertext though the best way to combat this is to use some method of authentication such as a checksum to ensure the integrity of the entire plaintext message.

Regarding synchronization, we note that provided an entire block of ciphertext is removed or inserted then decryption of the remaining blocks will remain unaffected.

In CBC mode the use of feedback ensures that an error in one ciphertext block will affect the decryption of two successive blocks - an error is more easily detected. However, this error propagation is not infinite and this mode of use is described as *self-synchronizing* (at the block level); an incorrect block only has influence on the following block and after two consecutive blocks are received correctly successful decryption will resume.

Similarly the CFB mode which also employs ciphertext feedback is also self-synchronizing and has some limited error propagation. The amount of propagation, and the time to resynchronize, is dependent on the size of b which denotes the size of the feedback block.

As far as error propagation is concerned, OFB mode is similar to ECB mode since errors introduced to one particular b-bit ciphertext block result in the incorrect decryption of just b bits. In ECB we note that any number of errors in a ciphertext block will affect the decryption of the whole block.

There is also the issue of synchronization to consider since the keystream used to encrypt and decrypt the message is generated independently of both the message and the ciphertext. Once the keystream generation falls out of step with the ciphertext in OFB mode, all decryption will be incorrect until

synchronization is re-established by some external intervention. This might be achieved by including in the transmission some synchronization marks or by restarting the block cipher encryptions at regular time intervals with a known sequence of initialization values.

11.6 Effect of modes on cryptanalysis

When we quote the data requirements to attack some block cipher, we are assuming that we are using the cipher in ECB mode. But if, as a cryptanalyst, we are intercepting some long message then it is unlikely that the ECB mode is being used for encryption. What effect might the choice of block cipher mode have on the potential success of a cryptanalyst?

We shall consider the effect of a change in block cipher mode on three forms of cryptanalysis; exhaustive search, differential cryptanalysis and linear cryptanalysis.

Exhaustive search

In an exhaustive search attack, the cryptanalyst tries each key in turn. The only complicating issue in mounting an exhaustive key search is in being able to recognize the correct plaintext when it has been recovered.

The requirements for the ECB mode have been discussed elsewhere (Section 3.3). If the CFB mode is used, then the previous ciphertext will also be required to recover a candidate pair of plaintext and ciphertext. This also holds for the CFB mode and the OFB mode if full feedback is used. (Note that this is the only recommended mode for OFB.)

When CFB mode is used without full feedback, then the number of previous ciphertexts required will depend on how many are required to construct a complete block of data. For a 64-bit block cipher with 8-bit CFB, eight consecutive ciphertext blocks will be required to provide a candidate pair of plaintext and ciphertext.

Differential cryptanalysis

Differential cryptanalysis (Section 3.4) is, as we have previously discussed, essentially a chosen plaintext attack. If we use any mode of the block cipher that effectively randomizes the input in an unpredictable way then a differential attack can be greatly hindered.

With the CBC mode of use, the previous ciphertext block is exclusive-ored with the current plaintext prior to encryption. If a cryptanalyst had previously chosen two successive plaintext blocks with the correct difference

between them, then such planning is wasted since the unpredictable outcome of the encryption of the first plaintext will effectively randomize the encryption for the second block.

There are two approaches. First, the cryptanalyst can change the attack. If the cryptanalyst observes the ciphertext output from the first encryption before choosing the second plaintext, it is possible to make changes to allow for the effect of the first ciphertext. Such an attack would be an adaptive chosen plaintext attack and may well be totally impractical. In addition, allowance must be made for the fact that the IV might change between encryptions!

Alternatively, the cryptanalyst could change the attack into a known plaintext attack with the corresponding increase in data requirements (Section 3.4).

Similar arguments hold for the CFB and OFB modes with full feedback. The uncontrollable output will randomize the input to the encryption function thereby forcing a vast increase in the plaintext requirements to mount a differential style attack.

When the CFB mode is used without full feedback then while the success of differential cryptanalysis will decrease, it may still be possible to use differential-style techniques [127].

Linear cryptanalysis

Since we do not need to choose the plaintext in a linear cryptanalytic attack (Section 3.5), this form of attack is much more robust against a change in the mode of the block cipher than is differential cryptanalysis.

In short, linear cryptanalysis of the CBC, OFB and CFB modes with full feedback can be mounted as effectively as an attack against the ECB mode. The only complication might occur when the CFB mode is used without full feedback. In such circumstances, the possible success of the attack will depend on which bits in the output are accessible to the cryptanalyst and which approximations can be used. Clearly, this is heavily dependent on the block cipher under consideration [127].

12 Multiple encryption

Intuitively we might expect a message to be more secure when it is encrypted more than once with the same block cipher, either with the same key or with an alternative key. As we saw in Section 4.3.3 it is important that the block cipher does not have any algebraic structure which would diminish the

security of multiple encryption; we know that the issue is settled for DES and that multiple encryption using DES is not susceptible to an algebraic weakness.

However, it is not always obvious just how much extra security is attained by using multiple encryption. Consider an n-bit block cipher using a k-bit key and generate the ciphertext for a plaintext block m by $E_{k_1}(E_{k_2}(m))$ where k_1 and k_2 are independent keys.

In the absence of any analytic weakness it seems that the work facing the cryptanalyst has been squared from being forced to try 2^k keys to trying 2^{2k} keys. However an observation by Merkle and Hellman [98] shows that this isn't necessarily true. They use known plaintext to mount a meet-in-the-middle attack which requires 2^k time and 2^k words of memory. Thus the time required for the cryptanalyst has not been increased by the introduction of double encryption, though of course there is a substantial increase in the amount of memory that is required.

The next suggestion is to use triple encryption, which with the block ciphers at our disposal today, might well result in a substantial operational overhead.

Merkle and Hellman [98] attribute the following scheme to Tuchman [144]; later it appeared in a standard [1]. The idea is to use two independent keys k_1 and k_2 and to encrypt a message m by $E_{k_1}(D_{k_2}(E_{k_1}(m)))$. The decryption mode of the block cipher is used in the middle to provide what is referred to as *backward compatibility*; by fixing $k_1 = k_2$ the triple encryption operation becomes equivalent to a single encryption and so hardware designed for triple encryption can still be used with information that has been encrypted only once.

However a chosen plaintext attack is described by Merkle and Hellman for this form of triple encryption using around 2^k operations, 2^k words of memory and 2^k chosen plaintexts. Note that though the operational overheads are similar to that required to break the double encryption, this attack is a chosen plaintext attack and hence difficult to mount in practice.

Van Oorschot and Wiener [123] provide a known plaintext attack on two-key triple encryption using an n-bit block cipher with a k-bit key. Their attack requires of the order of $2^{n+k}/t$ operations and t words of memory so there is a trade-off between the time required for the attack to succeed and the amount of necessary storage.

Instead of two-key triple encryption, Merkle and Hellman propose using three independent keys k_1, k_2 and k_3 and encrypting a message block m by $E_{k_1}(D_{k_2}(E_{k_3}(m)))$, denoted as the EDE3 mode. This still allows for backward compatibility since putting $k_1 = k_2 = k_3$ makes the triple encryption

equivalent to single encryption.

There are still further open questions about triple encryption in general, such as how EEE compares to EDE? Does one compromise security for the sake of backward compatibility? In the case of DES however with EEE, making the first two keys equal to the same weak key (Section 4.3.4) would yield an operation equivalent to single encryption, so both alternatives are backward compatible.

Now that DES is seemingly not secure enough as a single encryption cipher much attention is being focused on these questions and also on how to implement other modes of use such as cipher block chaining when the encryption is provided by a triple encryption using some block cipher [67].

Particularly with CBC mode (see Section 11) questions do arise as to how the security is affected when triple encryption CBC is implemented in one of two ways. The first way is to perform three CBC single encryption/decryption modes in series, referred to as *inner CBC* and denoted CBC-EDE; the second way is to implement *outer CBC* (EDE-CBC) where the feedback is taken from the output of the third encryption and used as input to the first.

There are performance issues. If the CBC is inner, that is the feedback is done on each of three single encryptions in series, then the triple encryption can be pipelined and the three units can be processing three different blocks at the same time allowing the same speed as ordinary single encryption. On the other hand, if the CBC is outer, then the encryption of a block is delayed until the feedback from the previous block is obtained, giving only one-third the speed of ordinary single encryption. Of course, this could be combated by interleaving three message blocks and processing them independently, but this requires a more sophisticated message processing mechanism.

As well as performance issues there are security issues. It is still not clear how the two modes compare. Work by Kaliski [64] noted that with respect to brute-force attacks, the outer CBC mode is stronger than inner CBC, but Biham [9] has established that with respect to differential cryptanalysis outer CBC is more secure. Indeed, Biham [10] demonstrates the viability of whole class of various attacks on different techniques of multiple encryption, even when mixing the ciphers that are used at the various stages.

Biham goes further in suggesting a principle that should be widely adopted; that is to avoid the use of any internal feedback values, or the feedback of external values to the inner workings of the algorithm. In some sense using such feedback allows controlled or observed changes to be introduced into some intermediate position of the algorithm thus giving an attacker considerable scope for mounting a cryptanalytic attack.

In the particular case of triple-DES Biham opposes the use of inner CBC. Viewing triple-DES as a 48-round cipher, the feedbacks in inner CBC are either entirely internal or they allow the introduction of external and known information into the internal mechanics of the 48-round algorithm. It is far better, he claims, to use outer CBC and to consider the three iterations of DES as a 48-round cipher without any modifications.

13 Conclusions

While a lot has changed in the year since the first version of this technical report, little of that change has had any substantial effect on the conclusions we can draw. And while there have been some clever developments in both the cryptanalysis and the design of new block ciphers, these developments will only become significant with time.

Instead, despite its age, DES is still the only block cipher that is used generally and trusted universally. More recent ciphers, proposed with more than an eye for adoption by the cryptographic community at large, have either succumbed to later cryptanalysis or still failed to accumulate a sufficient level of confidence.

Since no one can equal the resources and expertise that can be marshaled by the federal agencies both for the design and the cryptanalysis of cryptosystems, it is disappointing to find that the next proposed block cipher for federal standardization is to be classified.

What seems to be needed is a fast secure block cipher that can be efficiently implemented in software. Until then, implementers will be forced to adopt a half-way house whereby the security they require is achieved by the use of a DES derivative such as triple-encryption DES.

With hindsight, it seems that the issuance of a standard, and that standard being DES has both helped and hindered the cryptographic community. On the one hand it has been a focal point for cryptanalysis and for trying out new cryptanalytic tools. It has brought the science of cryptology into academic surroundings and provided a blossoming of interest throughout the technologically minded public.

On the other hand DES has perhaps overshadowed all other block cipher developments. While there is an increasing number of block cipher proposals, they are often very similar in design to DES. By contrast, the field of stream ciphers provides a vast number of alternatives, each with individual advantages and claims of security. Most types of stream cipher have their own characteristics and this means that different approaches are required

for the cryptanalysis of each type.

While DES has remained secure there is no impetus (apart from export concerns) to use any other block cipher, particularly one that is not standardized and has not been thoroughly cryptanalyzed. It is only now that those who are implementing block ciphers realize that a DES replacement is needed — it will take some time before a block cipher can ever be viewed as offering the same level of security as DES does even today.

References

[1] Accredited Standards Committee X9. *American National Standard X9.17: Financial Institution Key Management (Wholesale)*, 1985.

[2] C.M. Adams. *A Formal and Practical Design for Substitution - Permutation Network Cryptosystems.* PhD thesis, Queen's University, Kingston, Canada, 1990.

[3] C.M. Adams. On Immunity against Biham and Shamir's "Differential Cryptanalysis". *Information Processing Letters*, 41(2):77–80, 1992.

[4] C.M. Adams and S.E. Tavares. Designing S-boxes for ciphers resistant to differential cryptanalysis. In W. Wolfowicz, editor, *Proceedings of the 3rd symposium on State and Progress of Research in Cryptography, 1993*, pages 181–190. Fondazione Ugo Bordoni, 1993.

[5] K. Aoki, K. Ohta, S. Araki, and M. Matsui. Linear cryptanalysis of FEAL-8 (experimentation report). Technical Report ISEC 94-6 (1994-05), IEICE, 1994.

[6] H. Beker and F. Piper. *Cipher Systems.* Van Nostrand, London, 1982.

[7] I. Ben-Aroya and E. Biham. Differential cryptanalysis of Lucifer. In D.R. Stinson, editor, *Advances in Cryptology — Crypto '93*, volume 773 of *Lecture Notes in Computer Science*, pages 187–199, New York, 1994. Springer-Verlag.

[8] E. Biham. New types of cryptanalytic attacks using related keys. In T. Helleseth, editor, *Advances in Cryptology — Eurocrypt'93*, volume 765 of *Lecture Notes in Computer Science*, pages 398–409, Berlin, 1994. Springer-Verlag.

[9] E. Biham. On modes of operation. In R. Anderson, editor, *Fast Software Encryption*, volume 809 of *Lecture Notes in Computer Science*, pages 116–121, Berlin, 1994. Springer-Verlag.

[10] E. Biham. Cryptanalysis of multiple modes of operation. In J. Pieprzyk and R. Safavi-Naini, editors, *Advances in Cryptology — Asiacrypt '94*, volume 917 of *Lecture Notes in Computer Science*, pages 278–292, Berlin, 1995. Springer-Verlag.

[11] E. Biham. On Matsui's linear cryptanalysis. In *Advances in Cryptology — Eurocrypt '94*, *Lecture Notes in Computer Science*, Berlin. Springer-Verlag. To appear.

[12] E. Biham and A. Biryukov. How to strengthen DES using existing hardware. In J. Pieprzyk and R. Safavi-Naini, editors, *Advances in Cryptology — Asiacrypt '94*, volume 917 of *Lecture Notes in Computer Science*, pages 398–412, Berlin, 1995. Springer-Verlag.

[13] E. Biham and A. Biryukov. An improvement of Davies' attack on DES. In *Advances in Cryptology — Eurocrypt '94*, *Lecture Notes in Computer Science*, Berlin. Springer-Verlag. To appear.

[14] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991.

[15] E. Biham and A. Shamir. Differential cryptanalysis of FEAL and N-Hash. In D.W. Davies, editor, *Advances in Cryptology — Eurocrypt '91*, volume 547 of *Lecture Notes in Computer Science*, pages 1–16, Berlin, 1991. Springer-Verlag.

[16] E. Biham and A. Shamir. Differential cryptanalysis of Snefru, Khafre, REDOC-II, LOKI and Lucifer. In J. Feigenbaum, editor, *Advances in Cryptology — Crypto '91*, volume 576 of *Lecture Notes in Computer Science*, pages 156–171, New York, 1992. Springer-Verlag.

[17] E. Biham and A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag, New York, 1993.

[18] E. Biham and A. Shamir. Differential cryptanalysis of the full 16-round DES. In E.F. Brickell, editor, *Advances in Cryptology — Crypto '92*, volume 740 of *Lecture Notes in Computer Science*, pages 487–496, New York, 1993. Springer Verlag.

[19] D.K. Branstad, J. Gait, and S. Katzke. Report of the Workshop on Cryptography in Support of Computer Security. Technical Report NBSIR 77-1291, National Bureau of Standards, September 1977.

[20] G. Brassard. *Modern Cryptology*, volume 325 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1988.

[21] E.F. Brickell, D.E. Denning, S.T. Kent, D.P. Maher, and W. Tuchman. *Skipjack Review, Interim Report: The Skipjack Algorithm*, July 28, 1993.

[22] E.F. Brickell, J.H. Moore, and M.R. Purtill. Structure in the S-boxes of the DES. In A.M. Odlyzko, editor, *Advances in Cryptology — Crypto '86*, volume 263 of *Lecture Notes in Computer Science*, pages 3–9, New York, 1987. Springer-Verlag.

[23] L. Brown, M. Kwan, J. Pieprzyl, and J. Seberry. Improving resistance to differential cryptanalysis and the redesign LOKI. In *Advances in Cryptology — Asiacrypt '91*. To appear.

[24] L. Brown, J. Pieprzyl, and J. Seberry. LOKI: A cryptographic primitive for authentication and secrecy applications. In J. Seberry and J. Pieprzyk, editors, *Advances in Cryptology — Auscrypt '90*, volume 453 of *Lecture Notes in Computer Science*, pages 229–236, Berlin, 1990. Springer Verlag.

[25] K.W. Campbell and M.J. Wiener. Proof that DES is not a group. In E.F. Brickell, editor, *Advances in Cryptology — Crypto '92*, volume 740 of *Lecture Notes in Computer Science*, pages 512–520, New York, 1993. Springer-Verlag.

[26] F. Chabaud and S. Vaudenay. Links between differential and linear cryptanalysis. In *Advances in Cryptology — Eurocrypt '94*, *Lecture Notes in Computer Science*, Berlin. Springer-Verlag. To appear.

[27] D. Chaum and J. Evertse. Cryptanalysis of DES with a reduced number of rounds, sequences of linear factors in block ciphers. In H.C. Williams, editor, *Advances in Cryptology — Crypto '85*, volume 218 of *Lecture Notes in Computer Science*, pages 192–211, New York, 1986. Springer-Verlag.

[28] D. Coppersmith. The real reason for Rivest's phenomenon. In H.C. Williams, editor, *Advances in Cryptology — Crypto '85*, volume 218 of *Lecture Notes in Computer Science*, pages 535–536, New York, 1986. Springer-Verlag.

[29] D. Coppersmith. The data encryption standard (DES) and its strength against attacks. Technical Report RC 18613 (81421), IBM Research Division, December 1992.

[30] D. Coppersmith and E. Grossman. Generators for certain alternating groups with applications to cryptography. *SIAM Journal on Applied Mathematics*, 29:624–627, 1975.

[31] T.W. Cusick. The REDOC-II cryptosystem. In A.J. Menezes and S.A. Vanstone, editors, *Advances in Cryptology — Crypto '90*, volume 537 of *Lecture Notes in Computer Science*, pages 545–563, New York, 1991. Springer-Verlag.

[32] T.W. Cusick. Boolean functions satisfying a higher order strict avalanche criterion. In T. Helleseth, editor, *Advances in Cryptology — Eurocrypt '93*, volume 765 of *Lecture Notes in Computer Science*, pages 102–117, Berlin, 1994. Springer-Verlag.

[33] J. Daemen. 1993. Personal communication.

[34] J. Daemen. *Cipher and Hash Function Design.* PhD thesis, Katholieke Universiteit Leuven, 1995.

[35] J. Daemen, R. Govaerts, and J. Vandewalle. Block ciphers based on modular arithmetic. In *State and Progress in the Research of Cryptography, 1993*, pages 80–89, 1993.

[36] J. Daemen, R. Govaerts, and J. Vandewalle. A new approach to block cipher design. In R. Anderson, editor, *Fast Software Encryption*, volume 809 of *Lecture Notes in Computer Science*, pages 18–32, Berlin, 1994. Springer-Verlag.

[37] D.W. Davies and S. Murphy. Pairs and triplets of DES S-boxes. September 1993. Preprint.

[38] D.W. Davies and G.I.P. Parkin. The average cycle size of the key stream in output feedback encipherment. In *Advances in Cryptology — Crypto '82*, pages 97–98, New York, 1983. Plenum Press.

[39] D.W. Davies and W.L. Price. *Security for Computer Networks: An Introduction to Data Security in Teleprocessing and Electronic Funds Transfer.* John Wiley & Sons, New York, 1984.

[40] M. Davio, Y. Desmedt, M. Fosséprez, R. Govaerts, J. Hulsbosch, P. Neutjens, P. Piret, J.J. Quisquater, J. Vandewalle, and P. Wouters. Analytic characteristics of the DES. In D. Chaum, editor, *Advances in Cryptology — Crypto '83*, pages 171–202, New York, 1984. Plenum Press.

[41] R. Davis. The Data Encryption Standard in perspective. *IEEE Comms. Soc. Mag.*, 16(6):5–10, 1978.

[42] M.H. Dawson and S.E. Tavares. An expanded set of S-box design criteria based on information theory and its relation to differential-like attacks. In D.W. Davies, editor, *Advances in Cryptology — Eurocrypt '91*, volume 547 of *Lecture Notes in Computer Science*, pages 352–367, Berlin, 1991. Springer-Verlag.

[43] B. den Boer. Cryptanalysis of FEAL. In C.G. Günther, editor, *Advances in Cryptology — Eurocrypt '88*, volume 330 of *Lecture Notes in Computer Science*, pages 293–300, Berlin, 1988. Springer-Verlag.

[44] D.E. Denning. The Clipper Chip: A technical summary. June 23, 1993. Unpublished.

[45] D.E. Denning. The Clipper encryption system. *American Scientist*, 81(4):319–323, July–August 1993.

[46] Y.G. Desmedt, J.J. Quisquater, and M. Davio. Dependence of output on input in DES: Small avalanche characteristics. In G.R. Blakley and D. Chaum, editors, *Advances in Cryptology — Crypto '84*, volume 196 of *Lecture Notes in Computer Science*, pages 359–376, New York, 1985. Springer-Verlag.

[47] W. Diffie and M.E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22:644–654, 1976.

[48] W. Diffie and M.E. Hellman. Exhaustive cryptanalysis of the NBS Data Encryption Standard. *Computer*, 10:74–84, 1977.

[49] H. Eberle. A high-speed DES implementation for network applications. In E.F. Brickell, editor, *Advances in Cryptology — CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 521–539, New York, 1993. Springer-Verlag.

[50] P. Fahn. *Answers to Frequently Asked Questions About Today's Cryptography.* RSA Laboratories, September 1993. Version 2.0.

[51] H. Feistel. Block cipher cryptographic system. U.S. Patent No. 3,798,359, 1974.

[52] H. Feistel. Cryptography and Data Security. *Scientific American*, 228(5):15–23, 1973.

[53] R. Forré. The strict avalanche criterion: spectral properties of Boolean functions and an extended definition. In S. Goldwasser, editor, *Advances in Cryptology — Crypto '88*, volume 403 of *Lecture Notes in Computer Science*, pages 450–468, New York, 1990. Springer-Verlag.

[54] H. Gilbert and G. Chasse. A statistical attack on the FEAL-8 cryptosystem. In A.J. Menezes and S.A. Vanstone, editors, *Advances in Cryptology — Crypto '90*, volume 537 of *Lecture Notes in Computer Science*, pages 22–33, New York, 1990. Springer-Verlag.

[55] H. Gilbert and P. Chauvaud. A chosen plaintext attack of the 16-round Khufu cryptosystem. In Y.G. Desmedt, editor, *Advances in Cryptology — Crypto '94*, volume 839 of *Lecture Notes in Computer Science*, pages 359–368, New York, 1994. Springer Verlag.

[56] C. Harpes, G.G. Kramer, and J.L. Massey. A generalization of linear cryptanalysis and the applicability of Matsui's piling-up lemma. In L.C. Guillou and J.J. Quisquater, editors, *Advances in Cryptology — Eurocrypt '95*, volume 921 of *Lecture Notes in Computer Science*, pages 24–38, Berlin, 1995. Springer-Verlag.

[57] M.E. Hellman. DES will be totally insecure within ten years. *IEEE Spectrum*, 16:32–39, July 1979.

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

[59] M.E. Hellman, R. Merkle, R. Schroeppel, L. Washington, W. Diffie, S. Pohlig, and P. Schweitzer. Results of an initial attempt to cryptanalyze the NBS data encryption standard. Technical Report SEL 76-042, Stanford University, 1976.

[60] H.M. Heys and S.E. Tavares. On the security of the CAST encryption algorithm. In *Canadian Conference on Electrical and Computer Engineering*, pages 332–335, Halifax, Nova Scotia, Canada, 1994.

[61] R.R. Jueneman. Analysis of certain aspects of output feedback mode. In *Advances in Cryptology: Proceedings of CRYPTO '82*, pages 99–127, New York, 1983. Plenum Press.

[62] B.S. Kaliski and M.J.B. Robshaw. Linear cryptanalysis using multiple approximations. In Y.G. Desmedt, editor, *Advances in Cryptology — Crypto '94*, volume 839 of *Lecture Notes in Computer Science*, pages 26–39, New York, 1994. Springer Verlag.

[63] B.S. Kaliski and M.J.B. Robshaw. Linear cryptanalysis using multiple approximations and FEAL. In B. Preneel, editor, *Fast Software Encryption*, Lecture Notes in Computer Science, Berlin. Springer Verlag. To appear.

[64] B.S. Kaliski Jr. *On the Security and Performance of Several Triple-DES Modes.* RSA Laboratories, January 1994.

[65] B.S. Kaliski Jr., R.L. Rivest, and A.T. Sherman. Is DES a pure cipher? In H.C. Williams, editor, *Advances in Cryptology — Crypto '85*, volume 218 of *Lecture Notes in Computer Science*, pages 212–226, New York, 1986. Springer Verlag.

[66] B.S. Kaliski Jr., R.L. Rivest, and A.T. Sherman. Is the data encryption standard a group? *J. of Cryptology*, 1:3–36, 1988.

[67] B.S. Kaliski Jr. and M.J.B. Robshaw. Multiple encryption: Weighing up security and performance. *Dr. Dobb's Journal.* To appear.

[68] B.S. Kaliski and Y.L. Yin. On differential and linear cryptanalysis of the RC5 encryption algorithm. In *Advances in Cryptology — Crypto '95*, *Lecture Notes in Computer Science*, New York. Springer Verlag. To appear.

[69] K. Kim. Construction of DES-like S-boxes based on Boolean functions satisfying SAC. In *Advances in Cryptology — Asiacrypt '91.* To appear.

[70] K. Kim, S. Lee, S. Park, and D. Lee. Securing DES S-boxes against three robust cryptanalysis. Presented at Selected Areas in Cryptography '95, Ottawa, Canada. May 18–19, 1995.

[71] L. Knudsen. *Block Ciphers - Analysis, Design and Applications.* PhD thesis, Aarhus University, 1994.

[72] L.R. Knudsen. Cryptanalysis of LOKI. In M. Ganley, editor, *Cryptography and Coding III*, volume 45 of *The Institute of Mathematics and its Applications Conference Series*, pages 223-236, Oxford, 1993. Clarendon Press.

[73] L.R. Knudsen. Cryptanalysis of LOKI91. In J. Seberry and Y. Zheng, editors, *Advances in Cryptology — Auscrypt '92*, volume 718 of *Lecture Notes in Computer Science*, pages 196–208, Berlin, 1993. Springer-Verlag.

[74] L.R. Knudsen. Iterative characteristics of DES and s^2-DES. In E.F. Brickell, editor, *Advances in Cryptology — Crypto '92*, volume 740 of *Lecture Notes in Computer Science*, pages 497–511, New York, 1993. Springer-Verlag.

[75] L.R. Knudsen. New potentially 'weak' keys for DES and LOKI, May 1994. Presented at rump session, Eurocrypt '94.

[76] L.R. Knudsen. Practically secure Feistel ciphers. In R. Anderson, editor, *Fast Software Encryption*, volume 809 of *Lecture Notes in Computer Science*, pages 211–222, Berlin, 1994. Springer-Verlag.

[77] L.R. Knudsen. A key-schedule weakness in SAFER K-64. In *Advances in Cryptology — Crypto '95*, *Lecture Notes in Computer Science*, New York. Springer Verlag. To appear.

[78] L.R. Knudsen. Applications of higher order differentials and partial differentials. In B. Preneel, editor, *Fast Software Encryption*, *Lecture Notes in Computer Science*, Berlin, To appear. Springer Verlag.

[79] A.G. Konheim. *Cryptography: A Primer*. Wiley, New York, 1981.

[80] X. Lai. *On the Design and Security of Block Ciphers*. PhD thesis, ETH, Zurich, 1992.

[81] X. Lai. 1993. Personal communication.

[82] X. Lai and J. Massey. Device for the conversion of a digital block and use of same. U.S. Patent No. 5,214,703, 1993.

[83] X. Lai and J.L. Massey. A proposal for a new block encryption standard. In I.B. Damgård, editor, *Advances in Cryptology — Eurocrypt '90*, volume 473 of *Lecture Notes in Computer Science*, pages 389–404, Berlin, 1991. Springer-Verlag.

[84] X. Lai, J.L. Massey, and S. Murphy. Markov ciphers and differential cryptanalysis. In D.W. Davies, editor, *Advances in Cryptology — Eurocrypt '91*, volume 547 of *Lecture Notes in Computer Science*, pages 17–38, Berlin, 1992. Springer-Verlag.

[85] S.K. Langford and M.E. Hellman. Differential-linear cryptanalysis. In Y.G. Desmedt, editor, *Advances in Cryptology — Crypto '94*, volume 839 of *Lecture Notes in Computer Science*, pages 17–25, New York, 1994. Springer Verlag.

[86] J. Lee, H.M. Heys, and S.E. Tavares. On the resistance of the CAST encryption algorithm to differential cryptanalysis. Presented at Selected Areas in Cryptography '95, Ottawa, Canada. May 18–19, 1995.

[87] Lexar Corporation. An evaluation of the NBS Data Encryption Standard. Technical report, Lexar Corporation, 11611 San Vicente Blvd., Los Angeles, 1976.

[88] S. Lloyd. Counting binary functions with certain cryptographic properties. *Journal of Cryptology*, 5(2):107–131, 1992.

[89] J. Massey. SAFER K-64: A Byte-Oriented Block-Ciphering Algorithm. In R. Anderson, editor, *Fast Software Encryption*, volume 809 of *Lecture Notes in Computer Science*, pages 1–17, Berlin, 1994. Springer Verlag.

[90] J. Massey. SAFER K-64: One year later. In B. Preneel, editor, *Fast Software Encryption*, *Lecture Notes in Computer Science*, Berlin, 1995. Springer Verlag. To appear.

[91] M. Matsui. The first experimental cryptanalysis of the Data Encryption Standard. In Y. G. Desmedt, editor, *Advances in Cryptology — Crypto '94*, volume 839 of *Lecture Notes in Computer Science*, pages 1–11, New York, 1994. Springer-Verlag.

[92] M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology — Eurocrypt '93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397, Berlin, 1994. Springer-Verlag.

[93] M. Matsui. Linear cryptanalysis of DES cipher (I), January 1994. Preprint.

[94] M. Matsui. On correlation between the order of the S-boxes and the strength of DES. In *Advances in Cryptology — Eurocrypt '94*, *Lecture Notes in Computer Science*, Berlin. Springer-Verlag. To appear.

[95] M. Matsui and A. Yamagishi. A new method for known plaintext attack of FEAL cipher. In R.A. Rueppel, editor, *Advances in Cryptology — Eurocrypt '92*, volume 658 of *Lecture Notes in Computer Science*, pages 81–91, Berlin, 1992. Springer-Verlag.

[96] W. Meier. On the security of the IDEA block cipher. In T. Helleseth, editor, *Advances in Cryptology — Eurocrypt '93*, volume 765 of *Lecture Notes in Computer Science*, pages 371–385, Berlin, 1994. Springer-Verlag.

[97] R.C. Merkle. Fast software encryption functions. In A.J. Menezes and S.A. Vanstone, editors, *Advances in Cryptology — Crypto '90*, volume 537 of *Lecture Notes in Computer Science*, pages 476–501, New York, 1991. Springer-Verlag.

[98] R.C. Merkle and M.E. Hellman. On the security of multiple encryption. *Communications of the ACM*, 24:465–467, July 1981.

[99] S. Miyaguchi. The FEAL cipher family. In A.J. Menezes and S.A. Vanstone, editors, *Advances in Cryptology — Crypto '90*, volume 537 of *Lecture Notes in Computer Science*, pages 627–638, New York, 1990. Springer-Verlag.

[100] S. Miyaguchi, K. Ohta, and M. Iwata. 128-bit hash function (N-Hash). In *SECURICOM '90*, pages 123–137, 1990.

[101] J.H. Moore and G.J. Simmons. Cycle structure of the DES with weak and semi-weak keys. In A.M. Odlyzko, editor, *Advances in Cryptology — Crypto '86*, volume 263 of *Lecture Notes in Computer Science*, pages 9–32, New York, 1987. Springer-Verlag.

[102] S. Murphy. The cryptanalysis of FEAL-4 with 20 chosen plaintexts. *Journal of Cryptology*, 2(3):145–154, 1990.

[103] S. Murphy, F. Piper, M. Walker, and P. Wild. Likelihood estimation for block cipher keys, 1994. Preprint.

[104] National Bureau of Standards. Report of the workshop on estimation of significant advances in computer technology. Technical Report NBSIR 76-1189, National Bureau of Standards, 1976.

[105] National Institute of Standards and Technology (NIST). *FIPS Publication 46: Announcing the Data Encryption Standard*, January 1977. Originally issued by National Bureau of Standards.

[106] National Institute of Standards and Technology (NIST). *FIPS Publication 81: DES Modes of Operation*, December 2, 1980. Originally issued by National Bureau of Standards.

[107] National Institute of Standards and Technology (NIST). *FIPS Publication 46-1: Data Encryption Standard*, January 22, 1988. Originally issued by National Bureau of Standards.

[108] National Institute of Standards and Technology (NIST). *FIPS Publication 46-2: Data Encryption Standard*, December 30, 1993.

[109] National Institute of Standards and Technology (NIST). *FIPS Publication 180: Secure Hash Standard (SHS)*, May 11, 1993.

[110] K. Nyberg. December 1993. Comments made at *The Cambridge Algorithms Workshop*, December 9–11, 1993, Cambridge, U.K.

[111] K. Nyberg. Perfect nonlinear S-boxes. In D.W. Davies, editor, *Advances in Cryptology — Eurocrypt '91*, volume 547 of *Lecture Notes in Computer Science*, pages 378–386, Berlin, 1991. Springer-Verlag.

[112] K. Nyberg. On the construction of highly nonlinear permutations. In R.A. Rueppel, editor, *Advances in Cryptology — Eurocrypt '92*, volume 658 of *Lecture Notes in Computer Science*, pages 92–98, Berlin, 1993. Springer-Verlag.

[113] K. Nyberg. Differentially uniform mappings for cryptography. In T. Helleseth, editor, *Advances in Cryptology — Eurocrypt '93*, volume 765 of *Lecture Notes in Computer Science*, pages 55–64, Berlin, 1994. Springer-Verlag.

[114] K. Nyberg. Linear approximation of block ciphers, May 1994. Presented at rump session, Eurocrypt '94.

[115] K. Nyberg. S-boxes and round functions with controlled linearity. In B. Preneel, editor, *Fast Software Encryption*, *Lecture Notes in Computer Science*, Berlin. Springer Verlag. To appear.

[116] K. Nyberg and L.R. Knudsen. Provable security against differential cryptanalysis. In E.F. Brickell, editor, *Advances in Cryptology — Crypto '92*, volume 740 of *Lecture Notes in Computer Science*, pages 566–574, New York, 1993. Springer-Verlag.

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

[118] L. O'Conner. On the distribution of characteristics in bijective mappings. In T. Helleseth, editor, *Advances in Cryptology — Eurocrypt*

'93, volume 765 of *Lecture Notes in Computer Science*, pages 360–370, Berlin, 1994. Springer-Verlag.

[119] L. O'Conner. Properties of linear approximation tables. In B. Preneel, editor, *Fast Software Encryption*, *Lecture Notes in Computer Science*, Berlin. Springer Verlag. To appear.

[120] L. O'Conner and J. Golić. A unified Markov approach to differential and linear cryptanalysis. In J. Pieprzyk and R. Safavi-Naini, editors, *Advances in Cryptology — Asiacrypt '94*, volume 917 of *Lecture Notes in Computer Science*, pages 387–398, Berlin, 1995. Springer-Verlag.

[121] K. Ohta and K. Aoki. Linear cryptanalysis of the fast data encipherment algorithm. Technical Report ISEC 94-5 (1994-05), IEICE, 1994.

[122] K. Ohta and K. Aoki. Linear cryptanalysis of the fast data encipherment algorithm. In Y. G. Desmedt, editor, *Advances in Cryptology — Crypto '94*, volume 839 of *Lecture Notes in Computer Science*, pages 12–16, New York, 1994. Springer-Verlag.

[123] P.C. van Oorschot and M.J. Wiener. A known-plaintext attack on two-key triple encryption. In I.B. Damgård, editor, *Advances in Cryptology — Eurocrypt '90*, volume 473 of *Lecture Notes in Computer Science*, pages 318–325, Berlin, 1991. Springer-Verlag.

[124] B. Preneel. 1994. Personal communication.

[125] B. Preneel. *Analysis and design of cryptographic hash functions.* PhD thesis, Katholieke Universiteit Leuven, 1993.

[126] B. Preneel. Software performance of encryption algorithms and hash functions, Presented at Selected Areas in Cryptography '95, Ottawa, Canada. May 18–19, 1995.

[127] B. Preneel, M. Nuttin, V. Rijmen, and J. Buelens. Cryptanalysis of the CFB mode of the DES with a reduced number of rounds. In D.R. Stinson, editor, *Advances in Cryptology — Crypto '93*, volume 773 of *Lecture Notes in Computer Science*, pages 212–223, New York, 1994. Springer-Verlag.

[128] V. Rijmen and B. Preneel. On weaknesses of non-surjective round functions. Presented at Selected Areas in Cryptography '95, Ottawa, Canada. May 18–19, 1995.

[129] R.L Rivest. The MD4 message digest algorithm. In A.J. Menezes and S.A. Vanstone, editors, *Advances in Cryptology — Crypto '90*, volume 537 of *Lecture Notes in Computer Science*, pages 303–311, New York, 1991. Springer-Verlag.

[130] R.L. Rivest. *The RC2 Encryption Algorithm.* RSA Data Security, Inc., March 12, 1992.

[131] R.L. Rivest. The RC5 encryption algorithm. In B. Preneel, editor, *Fast Software Encryption*, *Lecture Notes in Computer Science*, Berlin. Springer Verlag. To appear.

[132] M.J.B. Robshaw. Recent proposals to implement Fair Cryptography. Technical Report TR - 301, RSA Laboratories, October 1993.

[133] M. Roe. Performance of symmetric ciphers and one-way hash functions. In Ross Anderson, editor, *Fast Software Encryption*, volume 809 of *Lecture Notes in Computer Science*, pages 83–89, Berlin, 1994. Springer-Verlag.

[134] I. Schaumuller-Bichl. Cryptanalysis of the Data Encryption Standard by a method of formal coding. In T.Beth, editor, *Cryptography, Proc. Burg Feuerstein 1982*, volume 149, pages 235–255, Berlin, 1983. Springer-Verlag.

[135] B. Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* John Wiley & Sons, New York, 1993.

[136] B. Schneier. Description of a new variable-length key, 64-bit block cipher (Blowfish). In R. Anderson, editor, *Fast Software Encryption*, volume 809 of *Lecture Notes in Computer Science*, pages 191–204, Berlin, 1994. Springer-Verlag.

[137] A. Shamir. On the security of DES. In H.C. Williams, editor, *Advances in Cryptology — Crypto '85*, volume 218 of *Lecture Notes in Computer Science*, pages 280–281, New York, 1986. Springer-Verlag.

[138] C.E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28:657–715, 1949.

[139] A. Shimizu and S. Miyaguchi. Fast data encipherment algorithm FEAL. In D. Chaum and W.L. Price, editors, *Advances in Cryptology — Eurocrypt '87*, volume 304 of *Lecture Notes in Computer Science*, pages 267–280, Berlin, 1988. Springer-Verlag.

[140] G.J. Simmons, editor. *Contemporary Cryptology, The Science of Information Integrity.* IEEE Press, New York, 1992.

[141] A. Sorkin. Lucifer, a cryptographic algorithm. *Cryptologia*, 8(1):22–41, 1984. Erratum: *ibid.* 7, p. 118, 1978.

[142] H.C.A. van Tilborg. *An Introduction to Cryptology.* Kluwer Academic Publishers, Boston, 1988.

[143] T. Tokita, T. Sorimachi, and M. Matsui. Linear cryptanalysis of LOKI and s^2DES. In J. Pieprzyk and R. Safavi-Naini, editors, *Advances in Cryptology — Asiacrypt '94*, volume 917 of *Lecture Notes in Computer Science*, pages 293–303, Berlin, 1995. Springer-Verlag.

[144] W.L. Tuchman, June 1978. Talk presented at the Nat. Computer Conf., Anaheim, C.A.

[145] S. Vaudenay. On the need for multipermutations: Cryptanalysis of MD4 and SAFER. In B. Preneel, editor, *Fast Software Encryption*, *Lecture Notes in Computer Science*, Berlin. Springer Verlag. To appear.

[146] G.S. Vernam. Cipher printing telegraph systems for secret wire and radio telegraphic communications. *J. Am. Inst. Elec. Eng.*, 55:109–115, 1926.

[147] A.F. Webster and S.E. Tavares. On the design of S-boxes. In H.C. Williams, editor, *Advances in Cryptology — Crypto '85*, volume 218 of *Lecture Notes in Computer Science*, pages 523–534, New York, 1986. Springer-Verlag.

[148] M.J. Wiener. Efficient DES key search. Presented at Crypto '93 rump session, August 20, 1993.

[149] M.J. Wiener. Efficient DES key search. Technical Report TR - 244, School of Computer Science, Carleton University, Ottawa, Canada, May 1994.