Differential and Linear Cryptanalysis in Evaluating AES Candidate Algorithms

Hannes Kruppa*, Syed Umair Ahmed Shah†

November 30, 1998

Abstract

Since the introduction of DES in 1977, cryptanalysis methods have been developed and constantly improved to assess the security of encryption algorithms. We give a comprehensive explanation of both differential and linear cryptanalysis techniques and examine their structural relationship. Furthermore, we show how cryptanalysis research has created a new design paradigm for secure ciphers, which we call "cipher design by cryptanalysis". This brings us to the current discussion on AES candidate algorithms, that have been submitted about three months ago. By summarizing the cryptanalysis and performance results available so far, we give an overview of the current status of this discussion. We observe that virtually all AES candidates are ciphers designed by cryptanalysis, and hence exhibit strong security properties. Eventually, this means that the winner of the AES competition will not be selected based on security alone, but also on secondary criteria, e.g. performance. We therefore develop an overall ranking scheme reflecting both security and secondary criteria and use this to identify the current frontrunners in the competition.

1 Introduction

In 1977 the National Institute of Standards and Technology (NIST), Chamber of Commerce, officially introduced the Data Encryption Standard (DES). Today NIST is considering proposals for alternative algorithms which offer a higher level of security and could replace DES. Ultimately, NIST intends to establish a new Advanced Encryption Standard (AES) which will specify an unclassified, publicly disclosed encryption algorithm capable of protecting sensitive government information well into the next century. Three months ago, in August 1998, 15 official candidates for AES were announced [33]. Several cryptanalysis groups from all over the world are now attacking these candidate algorithms. So far, only few of them have been found vulnerable. This is not surprising keeping in mind that over 20 years of cryptographic research on DES have led to design methods, in particular involving cryptanalysis, to make ciphers more secure.

The following section will give a brief introduction to cryptanalysis. Section 3 will focus on *differential* cryptanalysis and its application, especially on DES. Section 4 will focus on *linear* cryptanalysis. In section 5 we will give a comparison of the two techniques and examine their structural relationship. Section 6 will illustrate the current cipher design paradigm, which we call cipher design by cryptanalysis. In Section 7 we describe AES and its official evaluation concepts and algorithm requirements. Section 8 will attempt to evaluate the official AES candidates in terms of security and speed. Finally, in Section 9 we draw some conclusions from our study.

2 What is Cryptanalysis?

The basic requirement for a sound encryption scheme is that given a ciphertext C, it is very difficult to find the corresponding plaintext P, without the knowledge of the Key K. *Cryptanalysis* attempts to violate this. Either

*ETH Zurich, Department fur Informatik. Currently on visit at CMU
†Ph.D student in Computer Science Department of CMU

one attempts to determine K without prior knowledge of K, or to determine P from C regardless of K. Some of the most important cryptanalysis attacks include *differential cryptanalysis, linear cryptanalysis*, the exploitation of *weak keys* and *algebraic attacks*.

By definition cryptanalysis is successful as soon as the computational effort for recovering the plaintext is reduced to something less than a brute-force key-space search [7]. For example, DES encryption uses a 56-bit key, for which a brute-force attack would require trying 2^{56} different keys in the worst case. With differential cryptanalysis an attack on DES can be constructed that requires 2^{47} calculations using 2^{47} chosen plaintexts. We observe that successful cryptanalysis of a cipher does not always imply that there is a way to break the encryption in a practical sense. In a realistic setting, the only practical attack against DES is still brute force key-space search, because acquiring vast amounts of plaintexts or ciphertext needed for differential and linear cryptanalysis is feasible in theory only.

2.1 Classical Cipher Design

Before we can turn to cryptanalysis, understanding of some fundamental principles underlying classical cipher design is necessary. Classical ciphers mostly employ two basic techniques to achieve security: *diffusion* and *confusion*.

Diffusion dissipates the redundancy of the plaintext or keybits by spreading it out over the ciphertext. Typically, diffusion is realized using permutations. We observe that for example a bit-rotation is only a special form of a permutation. Since permutations are isomorphisms, they not only introduce linearity, but they are also easily invertible. If a cipher bit C_n is calculated by some combination of plaintext bits P_n and key bits K_n using linear functions, than the key bit can be reverse calculated using linear algebra. That is:

$C_i = K_n \oplus P_n \Rightarrow K_n = C_i \oplus P_n$

Of course, in a realistic setting, different linear operations would be mixed to combine different subsets of plaintext, ciphertext and keybits. A permutation might copy input bits to more than one output bit (expansion permutation) or just ignore certain input bits (compression permutation). This certainly makes reverse calculation more complicated, but it still remains linear and hence derivation of an equation still remains possible. Therefore, as a general rule, ciphers relying on diffusion alone can usually be broken without much effort [2].

Confusion obscures the relationship between the plaintext, the ciphertext and the key. This makes statistical or redundancy analysis more difficult, just as diffusion does. However, confusion is nonlinear and therefore difficult to invert. Typically, confusion is implemented using substitution tables (for example S-Boxes in DES, GOST, FEAL, Blowfish) and is most effective if the selected substitution depends on every bit of the plaintext block. In theory, confusion alone is enough for security, but the problem is that it requires large lookup tables consuming lots of memory. However, embedded systems and especially smartcards have very limited memory resources. More recently designed ciphers (e.g. IDEA) sometimes use modular multiplication for confusion and trade memory gains against performance costs. The art is to balance the security-cost trade-off by mixing confusion and diffusion in the one-round cipher design in an appropriate way.

3 Differential Cryptanalysis

Differential Cryptanalysis is basically a chosen-plaintext attack and relies on an analysis of the evolution of the differences between two related plaintexts as they are encrypted under the same key. By careful analysis of the available data, probabilities can be assigned to each of the possible keys and eventually the most probable key is identified as the correct one.

It is a statistical attack that can be applied to any iterated mapping (i.e., any mapping which is based on a repeated round function). It was first introduced by Murphy in an attack on FEAL-4 [34], but came into prominence when Biham and Shamir used it to come up with the most powerful, publicly known attack on DES at that time. Don Coppersmith (a member of the DES design team at IBM) had remarked that they were aware of this attack in 1974 when they were designing the S-boxes of DES and so the S-boxes were optimized against this attack. However, Don has also refused to reveal that whether this is the strongest attack on the DES that his team was aware of [1].

In the basic Biham and Shamir attack, substantially fewer pairs are required if DES is truncated to 6 or 8 rounds

[1]. For full DES this attack is impractical because it requires so many chosen plaintexts (2^{47}). They also showed that if the key schedule was removed from DES and a 16*48 = 768-bit key was used, the key could be recovered in similar steps thus showing that the independent subkeys do not add substantial security to DES. Furthermore, they showed that S-boxes of DES are extremely sensitive and that introducing slight changes in the entries of the S-box tables makes DES much more vulnerable to differential attack. Differential cryptanalysis has also been useful in attacking other cryptographic algorithms such as hash functions.

3.1 How does Differential Cryptanalysis work?

Differential Cryptanalysis is based on observing the difference in ciphertexts C, C' corresponding to plaintexts P, P' that satisfy a chosen known difference D(P, P'). Cryptanalysts choose many pairs of plaintexts for some specified difference between the members of the pair. They then study the difference between the members of the corresponding pair of ciphertexts. Statistics of the plaintext pair-ciphertext pair differences can yield information about the key used in encryption. Probabilities can be assigned to each of the possible keys and eventually the most probable key is identified as the correct one.

3.2 Differential Cryptanalysis of DES

The differential cryptanalysis of DES is based on analyzing the differential behavior of the F function in DES which takes a 32-bit input and gives a 48-bit. The F function contains an expansion function E, XOR of the key and the input, S-box and a final permutation P.

For DES and many other DES like cryptosystems the difference is chosen as a fixed XORed value of the two plaintexts. Given the XOR value of an input pair to the function it is easy to determine its XOR value after the expansion by the formula

$E(X) \oplus E(X') = E(X \oplus X')$

The XOR with the key does not change the XOR value in the pair, i.e. the expanded XOR stays valid even after the XOR with the key, by the formula:

$(X \oplus K) \oplus (X' \oplus K) = (X \oplus X')$

The output of the S boxes is mixed by the permutation P and the output XOR of the P is the permuted value of its input XOR, by the formula:

$P(X) \oplus P(X') = P(X \oplus X')$

The output XOR of the F function is linear in the XOR operation that connects the different rounds:

$(X \oplus Y) \oplus (X' \oplus Y') = (X \oplus X') \oplus (Y \oplus Y')$

The XOR of pairs is thus invariant in the key and is linear in the E expansion, the Permutation P and the XOR operation [1].

The S-boxes are known to be nonlinear. Knowledge of the XOR of the input pairs cannot guarantee knowledge of the XOR of the output pairs. Usually several output XORs are possible. For any particular input XOR not all the output XORs are possible and the possible ones do not appear uniformly, and some XORed values appear much more frequently than others.

Any S-box has 64*64 possible input pairs and each one of them has an input XOR and an output XOR. There are only 64*16 possible tuples of input and output XORs. Therefore, each tuple results in average from 4 pairs. However not all the tuples exist as a result of a pair, and the existing ones do not have a uniform distribution. We develop a difference distribution table for each S-box which gives the frequency distribution of each output pair XOR for a particular input pair XOR. The Differential attack algorithm on DES is as follows [1]:

1. Choose an appropriate plaintext difference (XOR for DES)
2. Create an appropriate number of plaintext pairs with the chosen plaintext difference (XOR). Encrypt them and keep only the resultant ciphertext pairs.

[1] Biham and Shamir [1] quoted (1993) a time of less than 0.3 sec on a personal computer using 240 ciphertexts for 6 rounds and less than 2 minutes on a PC by analyzing 2^{14} ciphertexts for 8 rounds.

3. For each pair derive the output difference (XOR) from the plaintext difference (XOR) and the ciphertext pair.
4. Derive the possible keys for each pair and for each possible key value count the number of pairs that result with the expected output difference (XOR).
5. The right key value is the (hopefully unique!) key value suggested by all the pairs.

According to Biham and Shamir this technique yields the right key with more than 99.9% accuracy. Biham and Shamir have shown that with reduced number of rounds, DES becomes quite vulnerable to *differential* cryptanalysis attacks. The summary of results by them is shown below.

No. of Rounds	Chosen Plaintexts	Known Plaintexts
4	2^3	2^{33}
6	2^8	2^{36}
8	2^{14}	2^{38}
10	2^{24}	2^{43}
12	2^{31}	2^{47}
14	2^{39}	2^{51}
16	2^{47}	2^{55}

They further showed that these attacks reduced to 10-16 rounds are not affected by the choice of the permutation and thus the replacement of the P permutation by any other permutation cannot make DES stronger. Even the replacement of the order of the 8 DES S-boxes can make DES much weaker. Similarly even a minimal change of one entry in one of the DES S-boxes can make DES easier to break.

3.3 Differential Cryptanalysis Example

We have devised a simple example to illustrate differntial cryptanalysis. Consider a DES like encryption scheme. To keep it simple, suppose the F-function contains only the XORing of the Key with the input and an S-box. We will represent the S-box input by S_I and output by S_O. The absolute input to the system is P. The difference function is XOR. The system works like this: it takes a 4-bit input P and XORs it with the key K. This is passed onto the S-box which outputs a 3-bit result. So we can construct the *difference distribution table* for this S-box since it is independent of the key. The range of possible input XOR differences would be 0-15, since we can have only 16 unique differences with any two 4-bit inputs. Similarly the output XOR differences would range from 0-7. The following table represents one such table. Different S-boxes will have different *difference distribution tables*. Notice that the frequency corresponding to entry (0, 0) is 16 which means that if there is no difference in the inputs, then there will be no difference in the outputs. We have shown only the relevant portions of the table which we will use later on. Notice that the frequencies are not uniformly distributed in a row and there are many zeros meaning that some output XORs are not possible for a certain input.

Input\Output Differences	0	1	2	3	4	5	6	7
0	16	0	0	0	0	0	0	0
1								
...								
4	0	2	4	0	0	8	0	2
...								
9	2	0	2	2	4	0	6	0
...								
15								

Let us first analyze the encryption function. The XORing with the key does not change the value of the input XOR. That is, if we XOR the input to the XORing function, we will get the same input difference as the difference between the output of the XOR function.

$(X \oplus K) \oplus (X' \oplus K) = (X \oplus X')$

Similarly if we had used a linear function, the input difference and the output difference for that function would be linearly related. Next we see from the above equation that if we XOR the output of the XOR function and the input, we can get the key.

$(X \oplus K) \oplus X = K$

But many different inputs can have the same input XOR. Thus we have many different possible values for the keys. We keep on finding sets of possible keys until one key is left which is our final key. We will now illustrate this calculation.

Suppose we first want to try the input difference of 4. We take the inputs 1010 (10) and 1110 (14) which have the XOR difference of (0100) 4 and encrypt them to get the ciphertext XOR value of 2. We look in the difference distribution table and see that the frequency is 4 for (4, 2). This means that 4 possible input pairs give this input/output difference pair. Suppose the possible inputs for an input difference of 4 are as shown in the table

Ouput XOR difference for Input XOR of 4	Possible Inputs
1	2, 6
2	3, 7, 8, 12
5	0, 4, 9, 10, 11, 13, 14, 15
7	1, 5

This means that the possible inputs for this are 3, 7, 8 and 12. Each of these exists as a pair i.e. (3 and 7) and (8 and 12) and will lead to a pair of keys. By using the second equation given above we get the following table which means that the key is one of the following.

S box input	Possible Keys
3, 7	9, 13
8, 12	2, 6

To narrow down the keys we try a different input pair of 0111 (7) and 1110(14) for an input pair difference of 1001 (9). Suppose the output XOR is 3.

Output XOR for input XOR 9	Possible Inputs
0	3, 10
2	0, 9
3	5, 12
4	1, 6, 8, 15
6	2, 4, 7, 11, 13, 14

This means that the possible inputs are 5 and 12 and the possible keys for this case are 2 and 11.

S box input	Possible Key
5, 12	2, 11

We can see that the only common key is 2 in both cases. Hence, key number 2 is the one we are looking for.

4 Linear Cryptanalysis

4.1 What is Linear Cryptanalysis?

Linear Cryptanalysis uses linear approximations to model nonlinear steps in the encryption process. Applying the approximations to a vast amount of known plaintext will eventually recover one key bit that is correct with a certain probability. Cipher-specific refinements of this method can recover multiple key-bits.

Matsui and Yamagishi first introduced linear cryptanalysis in 1993 in an attack on FEAL. Matsui later used linear cryptanalysis to attack DES, resulting in the most powerful DES attack known today.

4.2 How does Linear Cryptanalysis work?

Recall from section 2.1 that calculating key bits is easy when all calculations during the encryption process are linear. Assuming some calculations are nonlinear, one can still recover a key bit with high probability by finding good linear approximations for the nonlinear calculation steps. This is exactly the key idea behind linear cryptanalysis: Build equations that linearly approximate a key bit by combining some of the plaintext and ciphertext bits. If these equations have a bias, that is, they hold true for a certain probability $P \neq 0.5$ then this bias can be exploited to mount a known-plaintext attack.

In the known-plaintext attack we simply evaluate the approximations for a large amount of known-plaintext/ciphertext pairs and observe the outcome. After a sufficiently large number of evaluations we examine what bit value resulted in more than half of all the evaluations. This will reveal the true key bit with a certain probability. The weaker the bias of the approximation, the more plaintexts need to be processed for making a correct guess. Note that this approach will reveal only one single key bit, but there are refinements. We will get to these refinements when discussing the linear cryptanalysis of DES.

4.3 Linear Cryptanalysis Example

We have developed a simple yet instructive example to demonstrate the principles underlying linear cryptanalysis. The following table represents a 3x1 substitution function σ with its possible inputs (left column) and some linear function (right column) which will serve as an approximation.

$B_1B_2B_3$	$\sigma\ (B_1B_2B_3)$	$B_1 \oplus B_2$
000	1	0
001	0	0
010	1	1
011	0	1
100	1	1
101	1	1
110	0	0
111	0	0

Assuming that σ is a function between two vector spaces over finite fields $F_2^3 \rightarrow F_2$ and that XOR serves as our additive operation, we observe that σ is non-homomorphic and therefore nonlinear by definition of linearity: $\sigma(011) \oplus \sigma(110) = 0 \oplus 0 \neq 1 = \sigma(101) = \sigma(011 \oplus 110)$

It is therefore not possible to change the order of function applications making it more difficult to reverse-calculate the input. However, observe that we can approximate σ by XORing the first two input bits (third column in the table), and that $P[\sigma(B_1B_2B_3) = B_1 \oplus B_2] = \frac{6}{8} = 0.75$.

In general, we would calculate linear combinations of every subset of the input bits and examine which linearly combined subset of the output bits it approximates best. In the example above we just took the first two bits as our bits subset and did not examine different output subsets, since σ has only one bit output.

The bias β of this approximation is defined as the difference to an unbiased function and always is in the range of $[0; 0.5]$, i.e. in this example $\beta = 0.75 - 0.5 = 0.25$.

Now, if we evaluate the approximation formula say 20 times then with probability $P_{20,0.75}(K \geq 11) \geq 0.986$ more than half of the outcomes will equal the actual key bit. We calculated the probability using a binomial distribution. In reality the approximations that can be found have a bias orders of magnitudes smaller. A bias of the magnitude 10^{-12} is a realistic example. However, the general approach does not change, we just need to compensate for the weak bias by evaluating a vast amount of known plaintext/ciphertext pairs. For example, $2^{50} \approx 10^{15}$ is a typical number of evaluations needed given a realistic bias. In general the amount n of evaluations needed to make a

guess that holds with probability ρ is expressed as $P_{n,\beta}(K \geq n/2) \geq \rho$. We suggest to approximate the binomial distribution using a Gaussian for large n.

Extension to Multiple Rounds

By assuming independence between different rounds, we can multiply our one-round approximations to receive an approximation for multiple rounds [2]. Unfortunately, this also means that the bias of the resulting approximation can only deteriorate when adding rounds. This also accounts for the fact that many attacks are feasible only for ciphers implemented with a reduced number of rounds.

4.4 Linear Cryptanalysis of DES

The only nonlinear components in the one-round design of DES are the eight S-Boxes. These S-Boxes have been designed with differential cryptanalysis in mind but not linear cryptanalysis. In fact, DES is much more vulnerable to linear cryptanalysis than to differential cryptanalysis even after 16 rounds. Specifically, S-Box 5 has a one-round bias of about 0.2 which is so strong that the following refinement can be adopted:

- Use a 14-round linear approximation for rounds 2 through 15
- For the first and last round guess the 6 key bits relevant to S-Box 5

Altogether this will recover 1+6+6=13 key bits. Because of the symmetric design of DES this can be doubled by interchanging the role of plaintext and ciphertext. In other words we can reuse each ciphertext as a plaintext and vice versa and run the two variants in parallel. Altogether this will result in 2*13=26 key bits. The original complexity is thus reduced by 2^{26} leaving 2^{30} for brute-force key-space search. For a 85% success rate, 2^{43} known-plaintexts need to be evaluated for this attack.

Assuming that obtaining enormous numbers of known-plaintext pairs is feasible, linear cryptanalysis provides the most effective attack against DES known today. A software implementation of this attack recovered a DES key in 50 days using 12 HP9735 99 MHz workstations [6]. However, this attack discarded known plaintexts after processing, resulting in negligible storage requirements (generating the plaintexts took 40 of the total 50 days). We want to remark that in a practical environment, 2^{43} known plaintexts pairs of $2 * 64 = 128$ bits each would have to be intercepted and stored, resulting in 128 Terra bytes of data necessary to mount the attack. The only conceivable way to get this amount of data is by stealing magnetic tapes, which makes the attack quite unattractive. Altogether, neither linear nor differential cryptanalysis are considered being a threat to DES in a practical sense.

4.5 Linear Cryptanalysis of AES candidate LOKI97

So far there is only one instance which can clearly be seen as a successful linear cryptanalysis of a submitted AES candidate. It is not surprising to see few successful attempts, because modern cipher design by default involves cryptanalysis in the first place. Other ciphers have also been attacked but their weaknesses are more related to weak keys or key schedule problems rather than to linear cryptanalysis directly. The attacked cipher we will focus on is LOKI97, submitted by Brown and Pieprzyk [12] and cryptanalyzed by Knudsen and Rijmen [13].

LOKI97 has a classical DES-like design with 16 rounds and is the successor of the Australian encryption standard LOKI established in 1991. The main weakness of LOKI97 is that certain S-Box inputs are determined by the round key alone. That means inevitably, that these S-Box inputs will vary from round to round but not from plaintext block to plaintext block since they are independent of the plaintext. We observe that this violates the effectiveness requirement for confusion explained in section 2.1. The result is an unbalanced S-Box output enabling linear cryptanalysis. Specifically, it is possible to derive a key bit approximation with a $\frac{1}{16} = 0.06$ bias. Furthermore, LOKI97 exhibits a so-called type II linear relation, that is, the mentioned bias does not change from round to round but only every second round. This means, however, that only 8 one-round approximations for calculating a 16-round full LOKI97 approximation are needed. The resulting 16-round bias has a magnitude of 10^{-8} and will reveal one key bit using 2^{56} plaintexts. The authors do not give further details of the complexity of the attack, but similar

refinements such as in DES are certainly possible, because the single-round bias is comparably strong. The relevant S-Box has 13 inputs in LOKI97. We estimate that guessing and interchanging ciphertext and plaintext would result in (1+13+13) x 2 = 54 key bits leaving 2^{74} for exhaustive search and an overall required computational effort of 2^{101} steps (74+1+13+13 = 101).

5 Comparison of Differential and Linear Cryptanalysis

Both linear and differential cryptanalysis have been discovered while analyzing DES. Linear cryptanalysis is newer (Matsui 1993) and further improvements of the technique are likely. Structurally, there are strong similarities between differential and linear cryptanalysis. Both methods capture structural properties of a single round by building up large tables. The one-round results are then concatenated to model the cipher's behavior for multiple rounds. In both techniques, the quality of capturing the multiple round behavior will decrease when the number of rounds increases.

Probably the most important structural similarity is the notion of a characteristic [3]. For differential cryptanalysis, a characteristic is defined as a input difference that causes a certain output difference. The notion of a characteristic can be adopted in a natural way to linear cryptanalysis: In this case, we define a characteristic as a subset of input bits represented by a bit pattern B, so that $B_i = 1$ if and only if bit number i is a member of the input subset, else $B_i = 0$. The same definition holds for representing subsets of output bits.

Of course, there are also differences distinguishing the two methods: differential cryptanalysis will primarily lead to a chosen-plaintext attack, whereas linear cryptanalysis primarily enables a known-plaintext attack. Specifically, a ciphertext-only attack can not be derived from differential cryptanalysis, but from linear cryptanalysis it can. Chosen-plaintext attacks are irrelevant in linear cryptanalysis because characteristics are depending on single inputs or outputs only and not on specific input or output pairs called differentials in differential cryptanalysis. Hence, the notion of differentials does not extend to linear cryptanalysis, only the notion of characteristics does.

The following table summarizes the relationship of linear and differential cryptanalysis in terms of the authors and publishing years, the type of related attacks, the examined input and output, the key distillation process and the actual information being recovered and finally, the effectiveness against DES.

	Differential Cryptanalysis	Linear Cryptanalysis
First Published	1990 Biham and Shamir	1993 Matsui
Type of attack	Chosen-Plaintext. Known Plaintext also possible	Known Plaintext, Ciphertext only also possible
Examined Input	Difference of inputs (XOR in DES)	Linear combination of subsets of input (XOR in DES)
Input Properties	Differentials; refer to input pairs. The internal structure of each single input is irrelevant	Subsets specifically refer to the internal structure of a single input
Examined Output	Difference of pairs of output (XOR in DES)	Linear combination of subsets of output
Key Distillation	Repeated Evaluation of input-pair differences	Repeated evaluation of key bit approximation $K_i = InputSubset \oplus OutputSubset$
Analysis Result	Sub-Key (48 bits)	One Specific Key bit
Known-Plaintext in DES	$\frac{2^{55}}{2^{55}}$	$\frac{2^{43}}{2^{43}}$
Chosen Plaintext in DES	$\frac{2^{47}}{2^{47}}$	$\frac{2^{43}}{2^{43}}$

Research on the relationship of linear and differential cryptanalysis is still going on, especially because of their importance in cipher design. Clearly, they are considered to be dual rather than orthogonal concepts. Hybrid methods and generalizations have already been proposed, but their effectiveness has yet to be shown.

6 Cipher Design by Cryptanalysis

Interestingly, DES is almost perfectly resistant against differential cryptanalysis. This is no coincidence: both IBM and NSA knew about this method without making it public to the research community and designed the S-Boxes accordingly. However, DES is not specifically designed to preclude linear cryptanalysis. Either the authors did not know about it or they put priority on protecting against some other kind of attack we do not know about. Results from linear cryptanalysis have shown that by carefully rearranging the S-boxes DES can be improved to resist both linear and differential cryptanalysis [2].

Today, cryptanalysis has become a general tool not only to assess the security of different ciphers but to design secure ciphers in the first place. We call this cipher design by cryptanalysis. For example, by exhaustively examining characteristics and differentials we can optimize the design of nonlinear components in the cipher by choosing those that reveal the weakest statistical bias. Apart from that research has pointed out the importance of the number of inputs and outputs for S-Boxes. We note that increasing the number of output bits n will decrease the bias of any differential that could be exploited in differential cryptanalysis. Recall that a bias is derived from a relative frequency and as the total number of possible outcomes increases with n, the relative frequencies will decrease. However, increasing n not only increases cost as larger substitution tables are required, but it can as well increase the possibility of a successful linear cryptanalysis: For an S-Box with m inputs and n outputs, there definitely exists a linear relation if $n \geq 2^m - m$, and moreover if $n \geq 2^m$, then this results in a linear relation of only the output bits. These results are derived from advanced mathematical studies of Boolean functions and bent functions. See [5] for the very technical and rather complicated mathematical background.

As a summary, differential cryptanalysis suggests large numbers of output bits for substitutions, whereas linear cryptanalysis provides an upper bound.

The art of designing a good cipher is to balance security requirements and the involved trade-offs, such as speed, memory-requirements, flexibility and simplicity. The ideal cipher has the smallest possible key, requires the smallest amount of memory and is fast and only few rounds are required. We have briefly illustrated how cryptanalysis can serve as a means for making good decisions in cipher design. IDEA and DFC are two examples for ciphers that have been designed based on a strong cryptanalysis theory background. Both are provably secure against differential and linear cryptanalysis. IDEA has never been eligible to become a DES successor though, because it is patented. DFC, however, is a current AES candidate [27]. Cipher design by cryptanalysis has become common practice. The people who submitted the AES candidate algorithms had to provide their own cryptanalysis results together with the algorithm's specifications. These results help increase the trust in the cipher and its design, especially if attempts by others to achieve better cryptanalysis results fail.

7 Advanced Encryption Standard (AES)

7.1 AES Motivation

What really is considered a security threat concerning DES is the fact, that exhaustive search has become feasible at reasonable costs: Paul Kocher recovered a 56-bit key in 56 hours investing $250,000 in specialized hardware [34] and Wiener showed how a $1,000,000 computer could break DES in 3.5 hours. Triple-DES would certainly improve security, but at high cost since it is very slow. DES was primarily designed for hardware implementations which also explains the bit-oriented operations in its design. Only after DES had been established it become clear that most implementations would run in software and DES is clearly not optimized for that.

According to NIST requirements, each submission for candidate algorithms was required to be accompanied by a complete written specifications of the algorithm including [30]

- all necessary mathematical equations, tables, diagrams and parameters
- the design rationale for all these
- the estimated computational efficiency in hardware and software

- a series of Known Answer Tests (KATs) and Monte Carlo Tests (MCTs)
- a statement of the expected strength of the algorithm
- an analysis of the algorithm with respect to known attacks (known and chosen plaintext)
- a description of the advantages and limitations of the algorithm

7.2 AES official procedures

AES is the block cipher intended to become a Federal Information Processing Standard (FIPS) replacing DES. While reports over the last few years of the demise of DES have been greatly exaggerated, most people now agree that DES has come to the end of its useful life. AES is expected to be used for much into the 21st century. DES will not be reaffirmed as a federal standard after 1998. On January 2, 1997 the AES initiative was announced and on September 12, 1997 the public was invited to propose suitable block ciphers as candidates for the AES. On August 20, 1998 at the First AES Candidate Conference, the 15 official AES candidates were announced. Currently NIST is accepting public comments on the AES candidates. The public comment period closes on April 15, 1999. After that the public comments will be made public. The Second AES Candidate Conference will be held in Rome, Italy in March 22-23, 1999, where the public evaluation will be discussed (See [33] for the latest status). The NIST performance criteria for AES requires that the AES candidate should have a strength equal to or better than that of Triple DES with significantly improved efficiency over DES. The minimum acceptability requirements specified by NIST for the candidate algorithms include [30]:

- must implement symmetric (secret) key cryptography
- must be a block cipher
- shall be capable of supporting key-block combinations with sizes 128-128, 192-128 and 256-128 bits.

7.3 AES candidate algorithms

There is considerable interest in the AES initiative and 15 candidates were accepted for consideration in the first round from around the world. Among these were close variants of some of the more popular and trusted algorithms currently available, such as CAST, RC5 and SAFER-SK. Other good candidates from well-respected cryptographers were also submitted. The reason for close variants being proposed rather than the original ciphers is that one of the criteria for the AES submission is the ability to support 128-bit blocks of plaintext. Most ciphers were developed with an eye to providing a drop-in replacement for DES and, as a result, were often limited to having a 64-bit block size. The final 15 candidates are, CAST-256, CRYPTON, DEAL, DFC, E2, FROG, HPC, LOKI97, MAGENTA, MARS, RC6, RIJNDAEL, SAFER+, SERPENT and TWOFISH.

8 Evaluation of AES Candidates

8.1 Evaluation Criteria

In April 1997, NIST held a workshop in which they discussed the evaluation criteria for the AES submissions. In November 1997 they announced the following evaluation criteria based on security and performance. Security was accepted to be the most important criteria while performance and flexibility are given an equal weightage. Security criteria include

- Comparison of the algorithm to other submitted algorithms for the same key and block sizes
- Extent to which the algorithm output is indistinguishable from a random permutation of an input block.
- Soundness of the mathematical basis of security

- Other security factors raised by public, including any attacks and their practicality.

For performance, it has specified the following items of computational efficiency and memory requirements for comparing the algorithms [30]:

- Computational efficiency of both Hardware and Software implementations, specifically on key-block size combination (128-128).
- Memory requirements for both hardware and software implementations.
- Testing will be performed by NIST using mathematically optimized implementations provided for different platforms and environments.

Each candidate is also required to provide two mathematically optimized implementation of the candidate algorithms in ANSI C and Java programming languages. Using these NIST will assess the calculation of the time required to perform:

- Algorithm Setup
- Key Setup
- Key Change
- Encryption and Decryption

NIST will perform the tests on a single platform: IBM-compatible PC, with an Intel Pentium Pro Processor, 200MHz clock speed, 64MB RAM, running Windows 95. The compiler used by NIST will be the ANSI C compiler in the Borland C++ Development Suite 5.0 and the Java Bytecode compiler and virtual machine provided in Java Development Kit (JDK) 1.1 by Javasoft.

For estimating hardware performance of AES submissions, NSA will provide technical support to NIST. They will fully describe the hardware necessary to implement each of the algorithm submissions and will model and analyze the algorithms in CAD tools to report the following metrics:

- chip area
- throughput
- transistor count
- inputs/outpus required
- Key setup time
- algorithm setup time (tables etc.)
- time to encrypt/decrypt one block
- time to switch keys

8.2 AES candidate security ranking

According to the official AES regulations, security is evaluated for 128-bit keys. That means that cryptanalysis of any of the candidates is successful if the complexity of the attack is considerably less than a brute-force attack with complexity 2^{128}. It is, however, not expected that the attack is immediately practical. The authors submitted their own cryptanalysis results for their algorithms, making it quite improbable that severe security flaws will be found by others. What really matters is if the author can maintain his claims about the cipher's security. Even a completely infeasible attack might well deteriorate the confidence in the author and the design of his algorithm, if the author did not find the attack himself. On the other hand, confidence in the author's own design might well increase when he or she is able to successfully cryptanalyse a competitor's algorithm.

The winning candidate in terms of security can not be identified at this stage. However, a clear distinction can be made between losers, candidates that have been attacked successfully, and front-runners, candidates that have resisted all security concerns so far. There is also a second row of candidates, that have not been successfully cryptanalyzed but that exhibit small security margins or have been subject to other security criticism. The following table summarizes the three categories and gives brief comments for the second row.

DFC and RC6 have small security margins, that is, the difference between the proposed number of rounds and the number of rounds that would make the cipher vulnerable is rather small. This can be seen as an increase in security risk.

Hasty Pudding not only has an unorthodox name but also a very unorthodox design. The problem is that standard cryptanalysis methods do not apply directly to the design which might seem very attractive at first, but it also means that currently nobody is able to assess or quantify the cipher's security.

MARS has not been attacked successfully, but it was shown, that given a large key (160 bits and 1248 bits) an equivalent shorter key can be found in reasonable time. This does not at all affect the security as a cipher but limits its applicability. We observe for example, by definition MARS can not be used as a one-way hash-function, since it is not collision-free. Finally, there is SAFER+. SAFER (as well as IDEA) was designed by James Massey (ETH Zurich) for Cylink Corporation, a company that has well-known relations to the NSA. That makes us suspicious about possible backdoors.

Losers	Second Row	Comments	Frontrunners
DEAL	DFC	Small margin 5/8	CAST-256
FROG	Hasty Pudding	Unorthodox design	CRYPTON
LOKI97	Mars	Large key problem	E2
Magenta	RC6	Small margin 20/16	RIJNDAEL
	SAFER+	NSA-tainted	SERPENT, TWOFISH

It is clear that although security officially is the most important criteria, no final decision can be made based on security alone. Secondary aspects like speed, flexibility and simplicity will play the deciding role for determining the winner within the group of frontrunners in security.

8.3 AES Candidate Speed Ranking

8.3.1 Software Ranking

As already described in Section 7.1 each author has given the timing analysis for specific implementations of the algorithms on the NIST reference architectures. This gives a very easy method of comparing the speed of the algorithms. However, this ONLY means that a particular algorithm is faster than another on the reference system and not in general on any system. Most candidates have claimed that by using some other hardware or compiler, their performance improves significantly. For example, MARS claims that using DJGPP compiler on the same machine, they can encrypt/decrypt 2.4 times faster. Thus using just one machine or platform for judging the candidates seems to have a bias against certain algorithms.

Another problem in analysing the performance based on the speed is that different designers have chosen different number of buffer rounds i.e. iterations of the main encryption function over the minimum number of rounds after

which the cipher is considered safe. The larger the buffer, the greater are the chances of a cipher surviving attacks discovered later on, thus tending to have a longer life. But this adversely affects the speed of the cipher making their comparison even harder. Biham [4] has suggested that the minimum number of *'safe'* rounds should be used for comparing the speeds.

Furthermore, some are much well-suited and designed for certain architectures than others. Like being tested on a 32 bit computer rather than a 64-bit computer.

Directly comparing the individual data that the different candidates have provided has its own implications. Although almost everyone has submitted the required results for the reference platform, there is still a great deal of error involved. Different people have quoted different reference values for DES (rsaref) system and compared their speedups relative to that. So in all rating candidates in this manner seems to be full of mistakes. To add to the difficulty, different people have been rating the candidates based on their own tests and metrics and have been coming with different rankings.

In order to get a *relatively* fair comparison we decided to combine the rankings given by Brian Gladmanand [31] and El-Biham [4] and then come up with an overall ranking and performance categorization of the algorithms. Basically our ranking gives anyone who is good in most rankings a good ranking and thus we easily divide the candidates into two sets of algorithms; front runners and runners up.

The following table is based on the Pentium Pro 200 MHz reference machine for the ANSI C Borland C++ 5.0 compiler implementation of the algorithms. We restrict our analysis to the ANSI C implementation with 128 it key blocks and block lengths. We also include only the top 11 candidates as everyone seems to have made up their minds that the others are sure losers. The first ranking is based on the algorithms alone by Brian. The second reduces the rounds in each algorithm to the number of rounds at which the algorithm is safe. The third one is based on Biham's data analysis. The final column is our own ranking based on an average of their ranking.

Algorithm	Brian's ranking	Brian with min # of secure rounds	Biham	Overall
CAST-256	6	7	8	7
CRYPTON	5	5	3	5
DFC	11	11	11	11
E2	7	8	7	8
Hasty Pudding	9	10	9	9
MARS	2	1	5	3
RC6	1	2	4	1
RIJNDAEL	4	4	2	4
SAFER+	10	9	10	10
SERPENT	8	6	6	6
TWOFISH	3	3	1	1

Based on this, it can be concluded that RC6, TWOFISH, CRYPTON, MARS and RIJNDAEL seem to be the front-runners with good throughput and low setup costs while the others seem to have been left out with lower throughputs and higher setup costs.

8.3.2 Hardware Ranking

The hardware ratings are harder to come up with since very few have provided the designs or given quantitative results that can be compared and evaluated. Only data for CRYPTON, E2, MARS, SAFER+, SERPENT and TWOFISH is available. Everyone claims that the algorithm can be implemented on all chips like on smartcards taking a very small chip area and also giving a significant speedup. In the absence of proper modelling of these algorithms in VHDL or similar languages, with optimised hardware design, passing any judgment on the performance of the algorithms in hardware would not be reliable. Not only that it will be round 2 when the NIST assesses the hardware performance of the candidates and therefore we should wait till that data is made available.

8.4 Overall Ranking

Combining the overall security-performance trade-off we would place the algorithms in the following categories. TWOFISH, E2, RIJNDAEL seem to be in the front in both aspects. However it is too early to choose runners-up from front-runners conclusively since the discovery of new attacks can change the position of an algorithm in the table. Let's wait and see how AES moves from here.

Speed\Security	Runners-Up	Front-Runners
Runners-Up	DFC, SAFER+, Hasty Pudding	E2, SERPENT, CAST-256
Front-Runners	MARS, RC6	TWOFISH, CRYPTON, RIJNDAEL

9 Conclusion

Although cryptanalysis only rarely leads to practical attacks it has greatly inspired design principles for secure ciphers in general. Differential and linear cryptanalysis help to make good design decisions to balance security, performance and flexibility requirements. By examining the AES candidate algorithms we find that today the design of a new cipher automatically entails cryptanalysis. However, we also observe that the design of a secure cipher still requires a great deal of cryptographic expertise and that some of the submitted candidate algorithms do have security weaknesses.

About one third of the algorithms are very unlikely to exhibit any security flaw within the official AES time schedule. In general these are improved versions of algorithms which have been subject to public cryptanalysis attempts for years. Hence, secondary criteria i.e. performance will play a deciding role in determining the winner of the competition. Unfortunately, the evaluation of secondary criteria is difficult, because speed and memory requirements are greatly influenced by the compiler and the underlying computer architecture.

Although the official evalutation criteria have been fixed last year already, we feel they do not capture the details and difficulties of evaluating secondary criteria sufficiently. Keeping in mind that these criteria will play a major role in determining the winner, we expect a very controversial debate on this issue during the following weeks.

References

[1] Biham and Shamir, *Differential Cryptanalysis of the Data Encryption Standard.*

[2] Schneier, B. (1996): *Applied Cryptography*, Wiley & Sons Inc.

[3] Biham, E. (1994): *On Matsui's Linear Cryptanalysis*,Technion CS Department Technical Report CS0813-1994

[4] Biham, E.(1998): *Design Tradeoffs of the AES Candidates*, ASIACRYPT 98

[5] Chaubaud, F. (1994): *Liens entre la cryptanalyse differentielle et al lineaire,* ENS, Paris in EUROCRYPT 94 Proceedings

[6] Menezes et al: *Handbook of Applied Cryptography*, CRC Press

[7] Schneier, B.: *A Self-Study Course in Block-Cipher Cryptanalysis*, 1998

[8] Schneier, B: *Security Pittfalls in Cryptography*

[9] C. Adams, *The CAST-256 Encryption Algorithm*

[10] R. Anderson, E. Biham, L. Knudsen, *SERPENT*

[11] E. Biham, A. Biryukov, N. Ferguson, L. Knudsen, B. Schneier, A. Shamir, *Cryptanalysis of MAGENTA*

[12] L. Brown, J. Pieprzyk, *Introducing the new LOKI97 Block Cipher*

[13] V. Rijmen, L.R. Knudsen, *Weaknesses in LOKI97*

[14] Burwick, Coppersmith, D'Avignon, Gennaro, Halevi, Jutla, Matyas Jr., O'Connor, Peyravian, Safford, Zunic, *MARS - a candidate cipher for AES*

[15] M-J. Saarinen, *Equivalent keys in MARS*

[16] Cylink Corporation, *SAFER+*

[17] J. Daemen, V. Rijmen, *AES Proposal: Rijndael*

[18] D. Georgoudis, D. Leroux, B.S. Chaves, *The "FROG" Encryption Algorithm*

[19] D. Wagner, N. Ferguson, and B. Schneier, *Cryptanalysis of Frog*

[20] L. Knudsen, *DEAL: A 128-bit Block Cipher*

[21] S. Lucks, *On the Security of the 128-bit Block Cipher DEAL*

[22] Nippon Telegraph and Telephone Corporation, *The 128-Bit Block Cipher E2*

[23] C. H. Lim, *CRYPTON*

[24] R. Rivest, M.J.B. Robshaw, R. Sidney, Y.L. Yin, *The RC6 Block Cipher*

[25] Schneier, Kelsey, Whiting, Wagner, Hall, Ferguson, *Twofish: A 128-bit Block Cipher*

[26] R. Schroppel, *The Hasty Pudding Cipher*

[27] S. Vaudenay, *DFC*

[28] Cryptanalysis Reports, http://www.ii.uib.no/~larsr/aes.html

[29] Essays by Bruce Schneier, www.counterpane.com

[30] http://csrc.nist.gov/encryption/aes/aes_9709.htm

[31] http://www.seven77.demon.co.uk/aes.htm

[32] http://www.santafe.edu/~hag/crypto/node22.html

[33] http://csrc.nist.gov/encryption/aes/aes_home.htm.

[34] http://www.rsa.com/rsalabs/faq/html/2-4-5.html