Fault Injection Platform for Block Ciphers

Brian Baldwin[†], Emanuel M. Popovici[‡], Michael Tunstall[††], and William P. Marnane[†]

[†] *Claude Shannon Institute for Discrete Mathematics, Coding and Cryptography. Dept. of Electrical & Electronic Engineering, University College Cork, Cork, IRELAND.*
{brianb,liam}@eleceng.ucc.ie

[‡] *Claude Shannon Institute for Discrete Mathematics, Coding and Cryptography. Dept. of Microelectronic Engineering, University College Cork, Cork, IRELAND.*
e.popovici@ucc.ie

[††] *Dept. of Computer Science, University of Bristol, Merchant Venturers Building, Woodland Road, Bristol BS8 1UB, UNITED KINGDOM.*
tunstall@cs.bris.ac.uk

Abstract — **Block ciphers are typically resistant to direct attacks, such as an exhaustive key search or cryptanalysis, all of which require too many resources to achieve an efficient attack. Many block ciphers are examined for their resistance to less direct attacks that target a given implementation. Of these attacks, fault attacks are amongst the most effective at retrieving information on secret key, and require specific countermeasures to be included in an implementation. In this paper we describe a simple platform for the study of fault injection and analysis in the context of fault attacks block ciphers based on a Feistel structure (e.g. DES). We show that an attacker who can successfully inject faults into a block cipher can reduce the complexity of an attack to derive the secret key. We also present a novel version of the attack than can be applied to Triple-DES by independently injecting faults in the second and third instantiations of DES involved in a Triple-DES computation.**

Keywords — **Feistal, Implementation Attacks, Differential Fault Analysis, DES.**

I Introduction

Traditionally, block ciphers were viewed as a black box abstraction which took in a plaintext and a key and produced an encrypted ciphertext. In this case, the strength of a block cipher is entirely dependant on the secrecy of the key value, and the only ways to find this key value is to try all the different possible key values of find a mathematical weakness in the block cipher.

In cryptography, an implementation attack is any attack based on information gained from attacking an implementation of an algorithm, rather than theoretical weaknesses in the algorithms [8].

The possibility of injecting faults into cryptographic algorithms was first proposed in [7] and implementations of this type of attack have also been reported [2]. This led to the publication of numerous different attacks on other cryptographic algorithms.

In this paper we present a known-plaintext attack in a hardware based smart card or chip-level security processor environment by use of an FPGA development board, where the cryptanalyst has access to a ciphertext and its corresponding plaintext pair (or many such pairs), to discover the secret key. Using differential cryptanalysis and a version of the fault attack method described in [3], we run simulations with fault induced variations of the data lines, and we will show that the time taken to retrieve the secret key value can be greatly reduced by means of these induced faults.

This paper is organised as follows: We begin with a brief introduction of the DES architecture. We then move on to describe differential cryptanalysis. Following this we explain our implementation and the design method, followed by our results, and then we present our differential fault attack on Triple-DES, followed by our conclusions.

II DES Architecture

DES, the Data Encryption Standard algorithm, is the archetypal block cipher — an algorithm that takes a fixed-length string of plaintext bits and transforms it through a series of operations into another ciphertext bitstring of the same length [12]. DES was approved as a federal standard in 1976, and authorised for use on all unclassified data. In 2002, DES was finally superseded

by AES, the Advanced Encryption Standard, but its Feistal scheme forms the basis of many of the current block ciphers in use today. The applications use this block cipher include pay-TV smart cards, prepayment meter tokens, and remote locking devices for cars.

In the case of DES, the block size is 64 bits. DES also uses a key to customise the transformation, so that decryption can only be performed by those who know the particular secret key used to encrypt. The algorithm's overall structure is shown in Figure 1, with the Feistal function shown in Figure 2. There are 16 identical stages of processing, termed rounds. There is also an initial and final permutation, termed IP and FP, which are inverse operations. Before the main rounds, the block is divided into two 32-bit halves and processed alternately; this structure is known as the Feistel scheme [11].

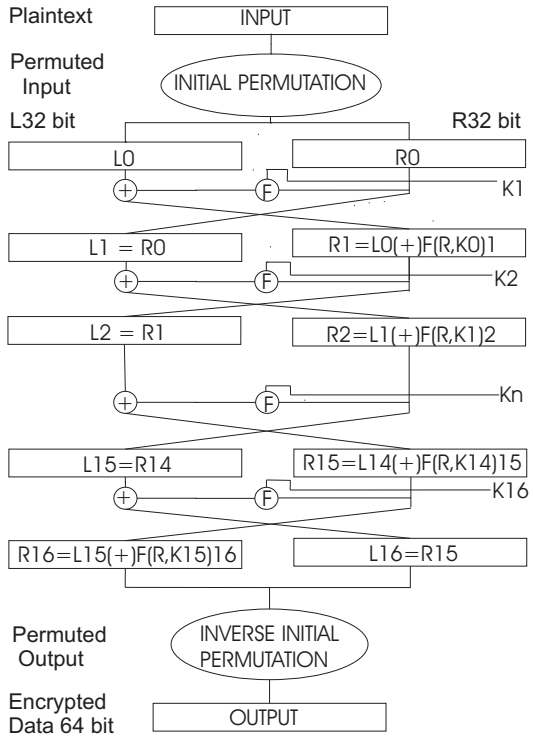

Fig. 1: The DES Algorithm.

The Feistel structure ensures that decryption and encryption are very similar processes; the only difference is that the subkeys are applied in the reverse order when decrypting. The rest of the algorithm is identical. This greatly simplifies implementation, particularly in hardware, as there is no need for separate encryption and decryption algorithms. The alternation of substitution from the S-boxes, and permutation of bits from the P-Permutation and E-Bit selection table provides so-called "confusion and diffusion" respectively, a concept identified by Claude Shannon in the 1940s

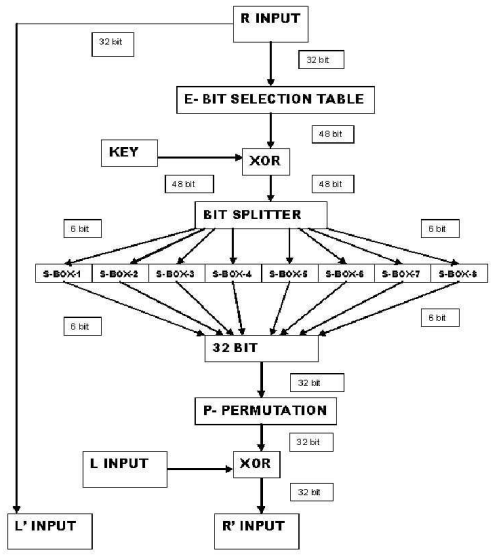

Fig. 2: The Feistel structure (F-function) of DES.

as a necessary condition for a secure yet practical cipher [13].

III Differential Cryptanalysis

Differential cryptanalysis has been, and still is, one of the most influential techniques in block cipher cryptanalysis. It led, in 1991, to the first attack on the full 16-round DES which was faster than exhaustive search [5]. Constructing a differential distinguisher consists of finding a distinctive property in the input and output blocks for the purposes of revealing the secret key value.

In a block cipher like DES, the secret sub-key bits are inserted into the encryption function by XORing them to intermediate data blocks during different round points in the computation. Let X_1 and X_2 be the values of such an intermediate data block for two different plaintexts P_1 and P_2. Assuming that both plaintexts are encrypted with the same key, we can write:

$$\begin{aligned} Y_1 &= X_1 \oplus K_i \\ Y_2 &= X_2 \oplus K_i \end{aligned} \tag{1}$$

and therefore,

$$\begin{aligned} &\Delta Y = Y_1 \oplus Y_2 \text{ and,} \\ &(X_1 \oplus K_i) \oplus (X_2 \oplus K_i) = X_1 \oplus X_2 = \Delta X \ . \end{aligned} \tag{2}$$

This simple equation illustrates the idea of a differential approach: while the adversary cannot compute the values Y_1 and Y_2 without knowing the round key K_i, they can easily determine their difference ΔY, given ΔX. The idea of differential cryptanalysis is to try to extend this property over multiple rounds. The attacker can predict

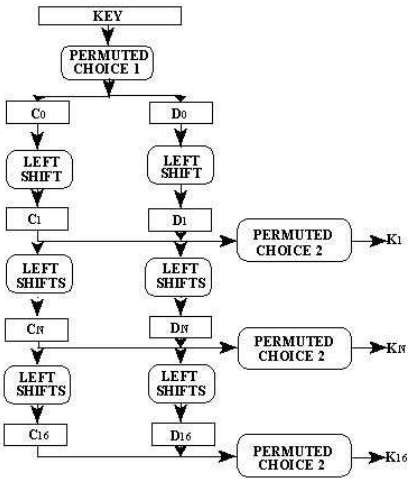

Fig. 3: The key-schedule of DES.

the output difference ΔK by tracing how the input difference ΔY evolves through the cipher, and then this can distinguish the cipher from a random permutation.

In practice however, a cipher does not only consist of key additions (which, as can be seen from above, do not produce a differential); it also contains diffusion components and non-linear substitution tables (S-boxes). Linear diffusion layers can be ignored to a certain extent. Although they do not preserve differences, they do transform them in a predictable way, i.e. if $Y_1 = A(X_1)$ and $Y_2 = A(X_2)$, for some linear function A, then $\Delta Y = A(\Delta X)$.

Unfortunately, this is not true for S-boxes (or any other non-linear component in a block cipher). Unless the difference ΔX at the input of the S-box is zero, the attacker typically cannot determine the output difference ΔY without knowing the actual value of X_1. However, given ΔX and assuming that X_1 are uniformly distributed, one can compute the statistical distribution of possible output differences. In order to proceed, the attacker must pick one of these output differences, compute the probability that their choice was correct, and continue their analysis.

Eventually, they will reach the output of the cipher, and will have described one of the possible ways in which the difference ΔP at the input could have propagated through the cipher. This is called a differential characteristic [9]. The probability that a given pair of plaintexts actually follows this characteristic is the product of the probabilities of all choices that the attacker must make (assuming that these probabilities are independent).

This type of cryptanalysis can be employed is a fault is injected into a device that is computing a block cipher, as the difference between an execution of a block cipher with and without a fault can be compared in the manner described above. Theoretical attacks that can exploit faults at the beginning and end of the computation of a block ciphers have been proposed [6, 9]. This type of attack is termed Differential Fault Analysis (DFA).

IV Implementation

We insert our faults on the 15th round of DES. With this technique, only the fifteenth and sixteenth rounds will vary from a fault free implementation. This was the method described in [3], and conducting the attack several times either at different positions in the fifteenth round with a varying message, we insert the faults over the entire right hand 32-bit data line with a logic high, as shown in Figure 4. In hardware, if the algorithm is a direct implementation, this is achieved by inserting switches to generate stuck-at-one faults into the circuit on the data line in the fifteenth round.

If a loop folding technique is implemented, some monitoring is initially required, either a count of the bitstream passing through on the data line to determine the round number, Figure 5, or some simple power analysis to determine at what point in time the target round is being computed.

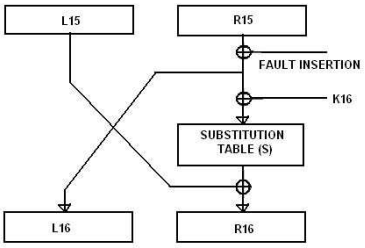

Fig. 4: Simplified DES last round model.

We then run a fault-free encryption using the DES algorithm while choosing to ignore the initial permutation, which is a simple bitwise permutation. If we denote the substitution table as function S, the output of the last round can be expressed in Boolean logic as:

$$\begin{aligned} L_{16} &= R_{15} \\ R_{16} &= S(R_{15} \oplus K_{16}) \oplus L_{15} \\ &= S(L_{16} \oplus K_{16}) \oplus L_{15} \end{aligned} \tag{3}$$

We can then repeat the encryption, using the same plaintext and key pairing, only this time with our fault insertion technique, where R_{15} is changed to R'_{15}. The output is transformed to:

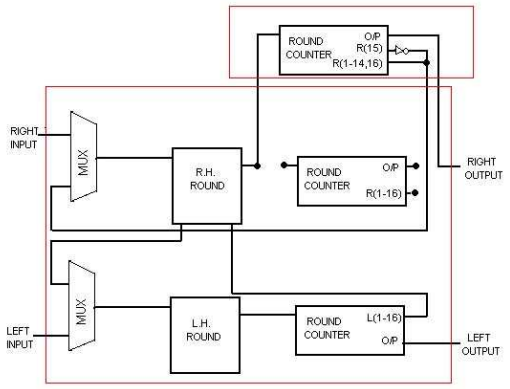

Fig. 5: Round block with fault insertion.

$$\begin{aligned} L'_{16} &= R'_{15} \\ R'_{16} &= S(R'_{15} \oplus K_{16}) \oplus L_{15} \\ &= S(L'_{16} \oplus K_{16}) \oplus L_{15} \end{aligned} \tag{4}$$

Then by using the differential approach and XORing the R_{16} with the fault inserted R'_{16}, we obtain:

$$\begin{aligned} L'_{16} &= R'_{15} \\ R_{16} \oplus R'_{16} &= S(L_{16} \oplus K_{16}) \oplus L_{15} \\ &\quad \oplus S(L'_{16} \oplus K_{16}) \oplus L_{15} \\ &= S(L_{16} \oplus K_{16}) \oplus S(L'_{16} \oplus K_{16}) \end{aligned} \tag{5}$$

This gives a relationship where only the value of the sixteenth subkey (K_{16}) is unknown; all the other variables being given directly as an output of the DES. For each substitution table used in the last DES round this relationship will hold true.

Since we now know the output from the substitution tables, an exhaustive search of the 64 possible values that validate Equation 5 can be conducted for each of the six bits corresponding to the input of each of the eight substitution tables. This will give approximately 2^{24} different hypotheses for the last subkey (determined through exhaustive simulation). We then implement a final exhaustive search through 2^{32} DES keys to find the whole key.

V Implementation Method

The implementation was performed using VHDL and then programmed onto a Digilent D2SB FPGA programmable logic development board. This development board comprises of a 200K gate Xilinx Spartan 2E200 FPGA in a PQ208 package, and associated hardware that provides 143 user I/Os.

Connected to the development board, on two banks of the expansion connectors, B and C, were two DIO4 digital I/O boards. Each I/O board consists of a set of eight switches. Each of the eight switches (per I/O board) were used to independently simulate a logical stuck-at-one fault, for each of the eight least significant bits of both the left and right hand inputs of the permuted input. In this way the inputs of the S-Boxes can be modified to simulate a fault attack.

A Search Unit program was developed and implemented to work in conjunction with the DES block. It was designed using differential cryptanalysis to study the differences of input and output blocks encrypted with the same key. Figure 6 shows the search unit operation.

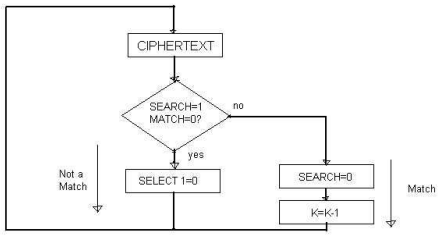

Fig. 6: Search Unit Operation Flow Chart.

The main premise of this is that if the attacker can choose the plaintexts as is the case here, the encryptions of a sufficient amount of plaintext pairs with a fixed difference ΔP can use the probability to distinguish the block cipher from a random permutation. The attacker can then count the number of pairs which produce the output difference predicted by their characteristic. As shown in Figure 7, the outputs from the DES block are fed directly into the inputs of the Search Unit and a key search is initiated. The Search Unit takes a key and a 64 bit block of ciphertext. It decrypts the ciphertext block with the key and checks the resulting block of plaintext. Since we already know the plaintext and are just looking for the key, if the plaintext from this key matches, or partially matches our known block of plaintext, then we have a partial or full recovery of the key. If not, it adds one to the key and repeats. In this way it searches its way through the key-space.

VI Results

The development board was designed with a DES algorithm/search unit pair. The board was left to run with the inserted faults and the partial key value was recovered. Modifying one S-box table in the fifteenth round will approximately change the inputs for on average 3.2 of the S-boxes in round sixteen [1]. Therefore, when we implement the attack a number of times, some secret key information is released. The process is then repeated to

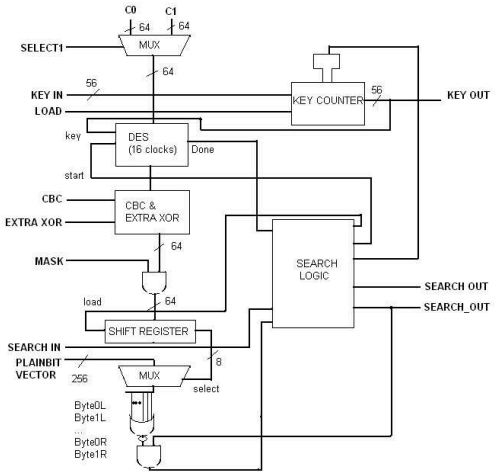

Fig. 7: Search Unit Operation Flow Chart.

Search Units on the FPGA	1
Clock speed (Hz)	5×10^7
Clocks per key (typical)	16
DES keys per search (unit per second)	3.125×10^6
Total DES keys per search	5×10^7
Search size (fault free)	7.21×10^{16}
Seconds per result (fault free)	14.41×10^8
Days per result (fault free)	16678 (around 45 years)
Search size (fault inserted)	268.5×10^6
Seconds per result (fault inserted)	288.2×10^3
Days per result (fault inserted)	3.33 (around 80 hours)

Table 1: Search unit results

obtain more information. An exhaustive search was then performed on the remainder of the key.

As can be seen from Table 1, when the key size has reduced from 56 to 32 bits, we can recover a complete key in approximately 80 hours.

VII DFA on Triple DES

Differential Fault Analysis can also be applied to triple DES by independently injecting faults in the second and third instantiations of DES involved in a triple DES computation. A method for attacking triple DES is given in [6], which involves first deriving the last subkey of the last round, and then the penultimate subkey, in order to derive the key used for the third instantiation of DES. This approach then allows faults in the second instantiation to be exploited. This assumes that the fault injection is perfect; a more robust and novel approach would be to use an extended version of the algorithm presented in Section IV, and this approach is part of the contribution of this work.

This approach involves independently injecting faults into the fifteenth round of the second and third instantiations of DES. Let the correct ciphertext block (generated from message block M) be C, a ciphertext block derived as a result of a fault in the fifteenth round of the third instantiation of DES be C', and a ciphertext block derived as a result of a fault in the fifteenth round of the second instantiation of DES be C''.

An attacker can compare C and C' to generate a set of possibilities for the key used in the last instantiation of DES. For each key hypothesis in this set, C and C'' are deciphered, to give a hypothesised state between the second and the third instantiations of DES. These candidate deciphered values can then be compared to yield information on the last subkey of the second instantiation of DES. The set of candidate keys resulting from this process can then be searched through, using the key used to decipher C and C'' to encipher M. When the key for the second instantiation of DES is found, the key hypothesis used to generate the input and the output of the second instantiation of DES is also validated.

For one attack on the fifteenth round of the second and third instantiation of DES, the total number of hypotheses generated will be $2^{32} \times 2^{32} = 2^{64}$, i.e. the expected size of the set of possibilities produced by combining C and C', each of which will produce a set of the same expected size.

This is a significant reduction when compared to the exhaustive search of size 2^{112} that is required to attack triple DES by exhaustively searching all the possible key values. This can be improved upon by acquiring more data; if two faulty ciphertext blocks are acquired for both the second and third instantiation of DES; the number of hypotheses generated becomes $2^{14} \times 2^{14} = 2^{28}$.

This can be further reduced if we take into account the fact that some deciphered values of C and C'' will not produce a valid keyspace. If C and C'' are deciphered with an incorrect key, then the intermediate states produced will be random values. When these values are compared, the relationship between L_{16} and R_{16} in the generated intermediate state will not depend on any bits of the key. The values generated will therefore no longer be distributed with the frequencies given in [4], but will be uniformly distributed across all the possible combinations.

If an impossible differential (i.e. a difference that could not occur because of the structure of the block cipher) is created when analysing the deci-

phered C and C'', then the key hypothesis used for deciphering C and C'' can be discarded.

The probability that all eight S-boxes produce a valid contribution to the keyspace is the product of the fraction of possible differentials for all of the S-boxes. This gives a probability of 0.125 (derived by exhaustive simulation), so the keyspace generated by this attack becomes $2^{32} \times (2^{32} \times 0.125) = 2^{61}$ for one faulty ciphertext block from a fault in the fifteenth round of the second instantiation, and one faulty ciphertext block from a fault in the third instantiation of DES. This can be improved to $2^{14} \times (2^{14} \times 0.125^2) = 2^{22}$, if two faulty ciphertext blocks are produced for both the second and third instantiations of DES.

This attack can be extended to triple DES with three independent keys, in which case faults will be required in the fifteenth round of each of the three instantiations of DES. Following the same reasoning as for two-key triple DES leads to a keyspace of size $2^{32} \times \left(2^{32} \times 0.125 \times \left(2^{32} \times 0.125\right)\right) = 2^{90}$ for one faulty ciphertext block from a fault in the fifteenth round of each instantiation of DES. This can be reduced to $2^{14} \times \left(2^{14} \times 0.125^2 \left(2^{14} \times 0.125^2\right)\right) = 2^{30}$ if two faulty ciphertexts are generated by injecting faults into each instantiation of DES.

VIII Conclusion

In this paper we presented a simple platform for the study of fault injection attacks on a Feistal algorithm, and proposed a novel approach for a simple attack scenario on the expanded version of that algorithm. Our assumption is that faults can be inserted in known locations, but that the number of faults one can inject is limited. The platform is built in such a way that it allows the study of fault attacks on the Feistal structure. We do not take into account the various countermeasures used to stop these types of attacks such as dummy random cycles, and hardware redundancy [3], it stands to reason that the more difficult it is for an attacker to exploit these fault attack methods, the more secure this hidden key will become.

We have demonstrated that with current technologies it is easy to attack block ciphers in reasonable time, provided that the ciphers do not include specific countermeasures, and that fault injection technology is readily available. The higher the number of injected faults into known locations, the shorter the attack time will be. The issue is to find the locations where fault attacks can be most easily exploited, so that these vulnerabilities can be reduced in the future either by finding ways to reduce the access to the locations, or to strengthen them against future attacks.

IX Acknowledgements

This material is based upon works supported by the Science Foundation Ireland under Grant No. 06/MI/006.

References

[1] F. Amiel, C. Clavier, and M. Tunstall. Collision fault analysis of DPA-resistant algorithms. In L. Breveglieri, I. Koren, D. Naccache, and J.-P. Seifert, editors, *Fault Diagnosis and Tolerance in Cryptography 2006 — FDTC 06*, volume 4236 of *Lecture Notes in Computer Science*, pages 223–236. Springer-Verlag, 2006.

[2] C. Aumüller, P. Bier, P. Hofreiter, W. Fischer, and J.-P. Seifert. Fault attacks on RSA with CRT: Concrete results and practical countermeasures. In B. S. Kaliski, C. K. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems — CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 260–275. Springer-Verlag, 2002.

[3] H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan. The sorcerer's apprentice guide to fault attacks. *Proceedings of the IEEE*, 94(2):370–382, 2006.

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

[5] 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. Springer-Verlag, 1992.

[6] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In B. S. Kaliski, editor, *Advances in Cryptology — CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer-Verlag, 1997.

[7] D. Boneh, R. A. DeMillo, and R. J. Lipton. On the importance of checking computations. In W. Fumy, editor, *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 37–51. Springer-Verlag, 1997.

[8] C. Giraud and H. Thiebeauld. A survey on fault attacks. In Y. Deswarte and A. A. El Kalam, editors, *Smart Card Research and Advanced Applications VI — 18th IFIP World Computer Congress*, pages 159–176. Kluwer Academic, 2004.

[9] L. Hemme. A differential fault attack against early rounds of (triple-)DES. In M. Joye and J.-J. Quisquater, editors, *Cryptographic Hardware and Embedded Systems — CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 254–267. Springer-Verlag, 2004.

[10] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In M. J. Wiener, editor, *Advances in Cryptology — CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer-Verlag, 1999.

[11] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[12] NIST. *Data Encryption Standard (DES) (FIPS–46-3)*. National Institute of Standards and Technology, 1999.

[13] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, 1949.