Differential Power Analysis Side-Channel Attacks in Cryptography

A Major Qualifying Project
submitted to the Faculty of
Worcester Polytechnic Institute
in partial fulfillment of the requirements for the
Degree of Bachelor of Science

by

William Hnath

Jordan Pettengill

29 April, 2010

Approved:

Professor Berk Sunar, Project Advisor

Professor William J. Martin, Project Advisor

Abstract

In this project we explore a technique known as differential power analysis and its strength as an attack in the cryptanalyst's arsenal. We begin by looking into side-channel attacks and how they operate in regards to a cryptosystem. Further, we explore the usage of mathematical models known as the Hamming Weight and Hamming Distance which provide a correlation between a given cryptographic protocol and its power consumption. The Data Encryption Standard (DES) and Advanced Encryption Standard (AES-128) are used as targets for the attack and are discussed in theory as well as how they are exploited using power characteristics. Finally, we share our attack and how it fairs when attempting to retrieve DES and AES-128 keys as well some insight as to how they can be improved.

Acknowledgments

We would like to acknowledge the following people for aiding in the success of our project:

- Professors William J. Martin and Berk Sunar, for advising the project and providing guidance in our research as well as our work in cryptography.
- Alan Chen, Brendon Chetwynd, Evan Custodio, Sam Ianni and Gerardo Orlando, for advising our project while at General Dynamics C4 Systems, providing resources to us to complete our project and providing necessary guidance in the completion of our project.
- Dr. Sylvain Guilley of Telecom ParisTech, for providing online support for the DPA contest code repository.

Contents

List of Figures

1 Introduction

We live in an age in which almost all sensitive information is controlled and distributed via computer networks. We have come a long way in protecting this information with a wide array of cryptographic schemes and protocols, but there are still many concerns for systems in which the physical implementations can be accessed. Systems which make use of cryptographic protocols that we use everyday such as ATM's and RFID smartcards are vulnerable to implementation attacks. Given physical access to the targeted device, an attacker can recover sensitive information which is otherwise supposed to be hidden, such as the devices key used for encryption. Due to many of these devices being readily available for physical access and an increasing number of individuals deploying these attacks for personal gain, there is an evident need for raising the robustness of our current cryptographic implementations by introducing countermeasures in both hardware and software.

Besides the need for protecting sensitive consumer information, there is another perspective to cryptanalysis in which we can use the methods of breaking ciphers to develop countermeasures and further strengthen them. Many side-channel cryptanalysis methods exist to attack encryption and authentication schemes running on various platforms. In this work, we will discuss differential power analysis; a branch of side-channel attacks where the attack is based on information gained from the power consumption of the cryptosystem. We do not question the theory of the cipher or its mathematical consistency, but rather look for clever ways to attack the implementations, such as observing the behavior of registers and how power levels vary while various instructions execute. By retrieving the power consumption data from the cryptosystem, we are able to make educated guesses as to how the algorithm on the system is operating. Furthermore, we can statistically correlate the power consumption data obtained using the power models and make guesses about parts of the exact key being used for encryption, once we have a priori knowledge of the protocol involved.

The primary departure point and main resource for this project is the DPA Contest, where power consumption information pertaining to a specific cryptosystem is published. This allows for cryptography researchers around the world to have access to a consistent data source to test and build on their attacks. The DPA Contest (`http://dpacontest.org`) is primarily a combined effort of several researchers at Telecom ParisTech, a technical university located in Paris, France. In the contest, a specific cryptographic protocol is selected as the target for attacks. This protocol is then implemented on a known device such as an FPGA evaluation board with several peripherals. On this board, power consumption characteristics are captured given certain plaintexts or ciphertexts and keys. These power traces are bundled into PostGreSQL tables in a database and are broken into different categories, such as random plaintexts being encrypted with a fixed key or with a random key. This allows for the testing of attacks with different approaches, such as differential power analysis and the more powerful template-based attacks which require the data introduced from more than one fixed key. The first version of the contest uses the Data Encryption Standard (DES) as the cryptographic scheme, whereas the second version uses the 128-bit variety of the Advanced Encryption Standard (AES-128).

The goals of this project were to research side-channel attacks and develop our own attack based on DPA to target the DES and AES-128 cryptosystems. In this work, we begin by introducing the reader to the idea of a side-channel attack in cryptography and the need for the method in cryptanalysis. Further, we introduce the power analysis technique and the motivation for its work. The Hamming Weight model is introduced as a partial means to execute the attack and our work on the DES and AES-128 cryptographic schemes is discussed in detail. Finally, we make conclusions on our findings as well as comment on some Future Work for the project, such as template-based attacks for the DPA contest and further improvements to the current attacks.

2 Background

2.1 Side-Channel Attacks

Side-channel cryptanalysis is a branch of cryptography in which sensitive information is gained from the physical implementation of a target cryptosystem. This is in contrast with other forms of cryptanalysis where the algorithms and their underlying computational problems are attacked. All electronic devices leak information in a multitude of ways. Prominent examples of this are temperature, power consumption, time taken for computations, acoustics and electromagnetic emanations. In general, these types of information leakages may be tied in some way to the types of operations that the cryptographic algorithm is performing. This makes them a very powerful tool for gaining the desired information from the cryptosystem. All of the aforementioned leakages are passive, meaning that we only capture power consumption information as it normally comes off of the target device. Further, this type of attack requires no active manipulation of the device by the adversary. Figure 1 shows a generic model for a side-channel as it relates to the implementation of a cryptosystem.

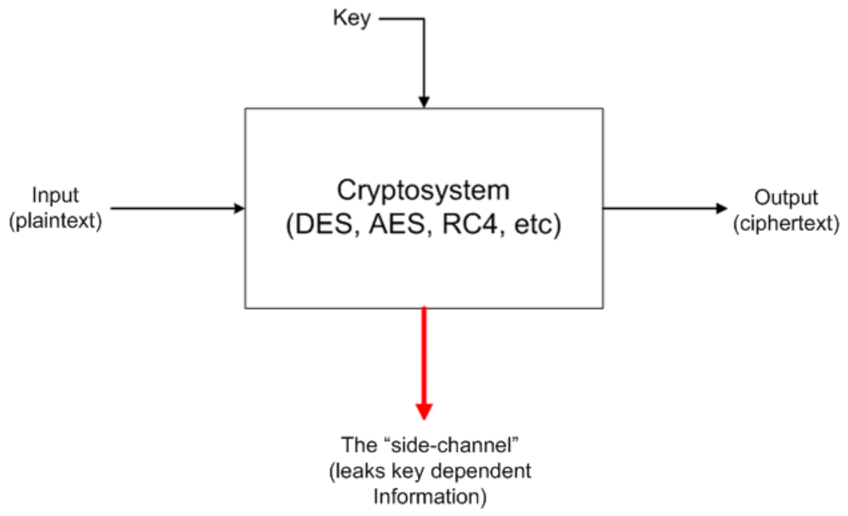

Key
Input (plaintext)
Cryptosystem (DES, AES, RC4, etc)
Output (ciphertext)
The "side-channel" (leaks key dependent Information)

Figure 1: A typical cryptosystem can leak information on various side-channels.

As shown above, we have a cryptosystem which contains an implementation of a cryptographic algorithm within it. Plaintexts come into the system and are encrypted with the key into a given ciphertext. The focal point of this model is the "side-channel", which is where information pertaining to the cryptosystem and key are leaked. This is where the majority of the information will be gathered and used for later analysis in a specific attack. Due to the passive nature of the attack, we only take in information as it is executed and do not introduce any modifications to the cryptosystem. There also exist active attacks, such as fault induction where the target device is altered by the attacker in order to force it into a vulnerable state. Active attacks may be classified as non-invasive, semi-invasive and invasive attacks. Active attacks are categorized according to the level at which the adversary tampers with the target device. Passive side-channel attacks are the primary focus of the project and we will make use of power consumption profiles collected from the target device.

2.2 Power Analysis Attacks

Power Analysis attacks may be defined as a type of side-channel attack where a device's power consumption serves as the leaked information used to exploit the particular device. During the execution of a cryptographic algorithm on a particular device, information pertaining to the low-level instructions the algorithm is using may be revealed in the power traces that are retrieved. This in turn allows the attacker to identify the instructions being used (MOV, XOR, etc), as well as branch statements to ultimately derive the cryptosystem and recover the key. A typical generic setup for a power analysis attack is shown below in Figure 2.

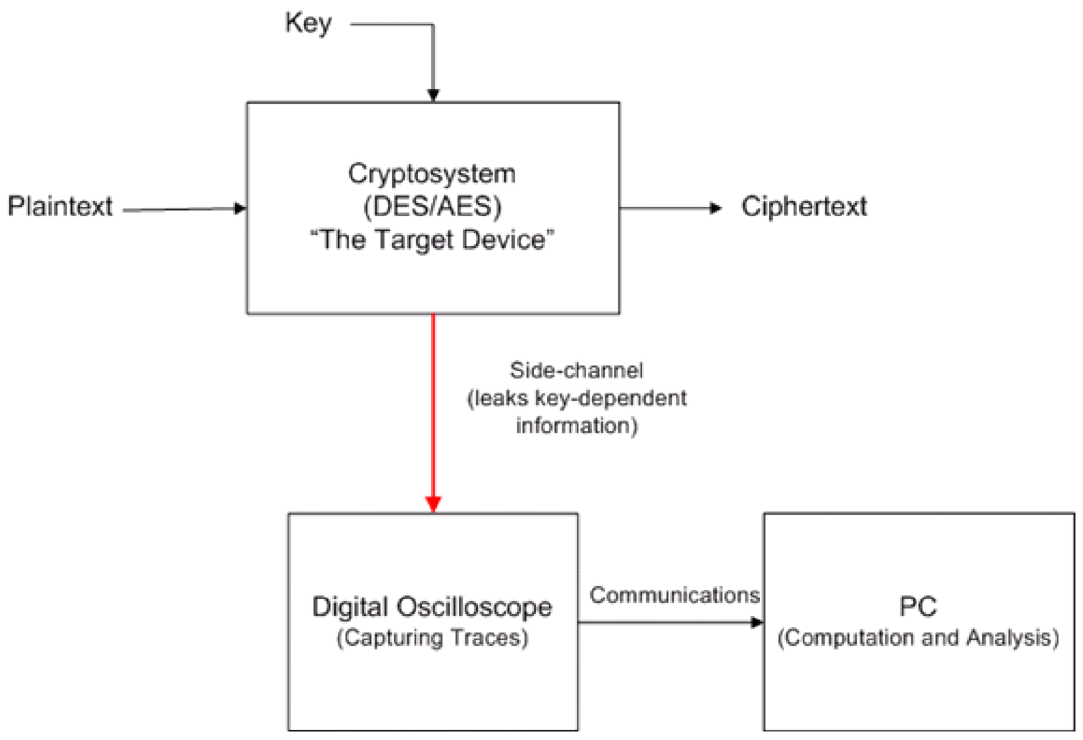


Figure 2: A depiction of a typical side-channel power analysis attack setup.

As we can see, the setup for a power analysis side-channel attack typically consists of a target device, a digital oscilloscope and a computing platform. A target device is considered one which is running some sort of cryptosystem and actively executing the encryption scheme. An oscilloscope is hooked up to the target device via probes and captures the power consumption at a determined sample rate. The collection of points containing the voltage level measured from the circuit is called a *power trace*. Once captured, the traces may be sent to a personal computer running any variant of data processing software (MATLAB, ANSI C code, etc) to analyze and process the traces. During this stage of a power analysis attack, the adversary does not need to know about the cryptographic algorithm being used. However, after an initial set of traces has been accumulated, it is the job of the adversary to used this power consumption data to learn as much about the cryptosystem as possible. Without proper analysis of the cryptosystem from this information, the adversary will not be able to write a successful differential power analysis attack. An example recording of a power-based side-channel is shown below in Figure 3.

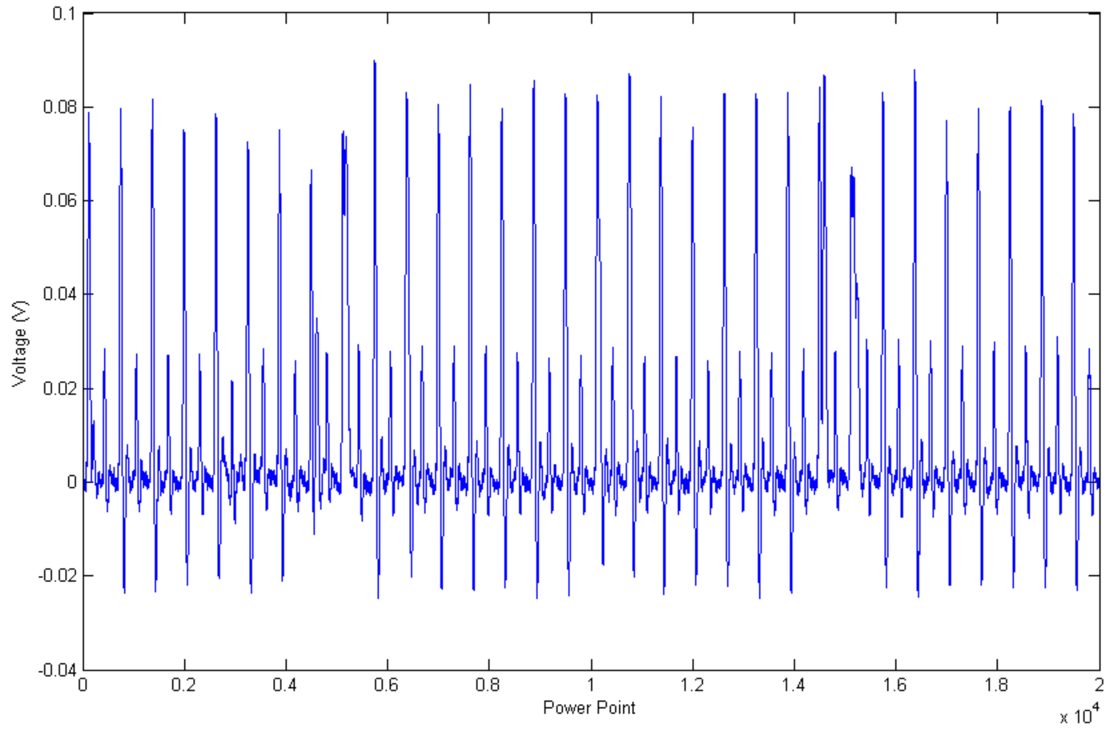


Figure 3: An example of the measured results from a side-channel reading of the power consumption of a DES circuit during encryption.

The above figure shows an example of the power consumption of a circuit during an encryption. The specific encryption used is the Data Encryption Standard (DES), described in Section 5. In order to obtain a noise free power trace for a given encryption, many traces must be taken using the same plaintext input and key. These power traces are then averaged together to remove any potential noise. When it comes to analysis of the power traces, we are presented with three different types of attacks in which to utilize them. These attack types are termed *simple power analysis* (SPA), template-based analysis, and differential power analysis (DPA).

In a simple power analysis attack, the attacker looks at a set of power traces and using their reasoning and knowledge of the system attempts to determine the operations being executed and eventually the cryptographic key. In a template-based attack, the attacker builds up a series of templates with data which has been profiled from the target device and calculates a probability on each template in accordance with a trace. The key used in the encryption is then determined using probability analysis [5]. In a differential power analysis attack, traces are taken and statistically correlated to intelligently make guesses at the key to derive the correct one. Figure 3 is an example of a power trace that belongs to a differential power analysis attack, where multiple traces will be taken and used to discover the key used. Simple power analysis will only work with traces which have little to no noise, where as the template-based and differential power analysis approaches will work on power consumption traces with varying amounts of noise due to their signal processing and error-correcting properties. Differential power analysis attacks are the focus of this project and are discussed in further detail in the later sections of this report.

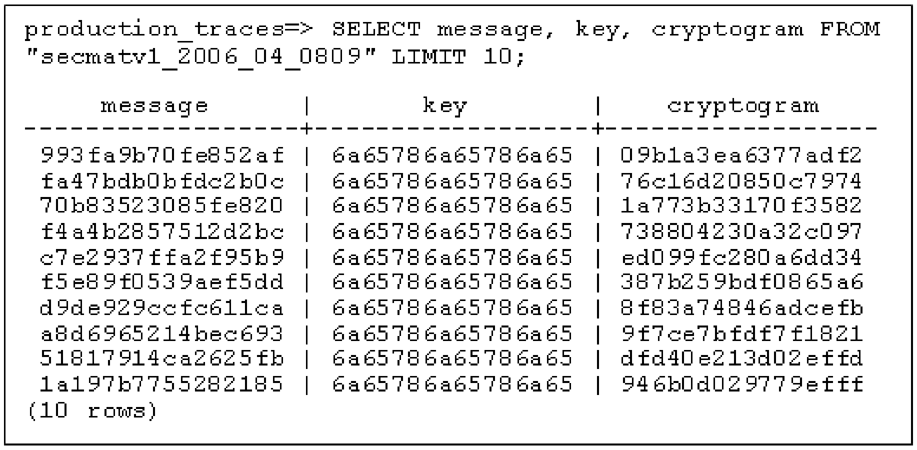
3 The DPA Contest

The DPA Contest4 is an online-based differential power analysis contest organized by the VLSI research group from the COMELEC department of the Telecom ParisTech University. This contest serves as the underlying point of departure for our project due to its publicly available power traces as well as existing repository of DPA attacks. The contest was first announced during the 2008 Workshop on Cryptographic Hardware and Embedded Systems (CHES). The second installment, termed DPA Context v2 was announced during CHES 2009 and builds on the previous contest with a few improvements. The goal of the DPA contest series is to facilitate a community for side-channel attack and cryptanalytic researchers. By providing a common set of data for particular attacks, participants are able to design and execute attacks as well as compare them amongst others in the community for effectiveness. In later contests, the target devices characteristics as well as schematic layouts have also been made available for further side-channel attack analysis. Power traces for the contest are stored in a database to allow for easy retrieval and software integration of the data. The trace database provides a cleanly acquired set of traces for the given cryptosystem to which attack developers can utilize for testing. All plaintexts, ciphertexts and keys are made available in the trace tables to provide an easy way to compare results for correctness. Details on the individual competitions are explained in the following sections.

3.1 DPA Contest V1 - DES

The first installment of the DPA contest focused on a specific class of attacks, mounted on a single cryptographic algorithm with a single implementation. The contest is made this specific to allow for focus to be placed on the attacks instead of the concerns that arise from a more complex cryptographic scheme. As stated before, the class of attacks under consideration for the contest is power analysis type attacks, which include but are not limited to simple, differential and template-based power analysis as well as any combination or variant of the three. The cryptographic algorithm used for the contest is the Data Encryption Standard (DES) because it is one of the more heavily studied encryption algorithms and will provide a simple learning curve for those approaching the attacks. The DES implementation is run on a custom FPGA development board created by one of the contest founders, Dr. Sylvain Guilley, as part of his Ph.D. thesis [9]. The trace acquisition board is described extensively in Dr. Guilley's thesis and serves as the platform from which the traces included in the trace tables are acquired.

The trace tables are made publicly accessible via a PostGreSQL database which is hosted by the COMELEC department at Telecom ParisTech. The contest's traces are organized into separate tables according to when they were acquired and each trace includes the plaintext message, the key used to encrypt the message, the ciphertext and the trace data stored in '.bin' format which contains all of the individual data points for each trace as a function of time. A sample query for traces in a particular trace table as well as its result is shown below in Figure 4.



```
production_traces=> SELECT message, key, cryptogram FROM
"secmatv1_2006_04_0809" LIMIT 10;

     message      |       key        |    cryptogram
------------------+------------------+------------------
 993fa9b70fe852af | 6a65786a65786a65 | 09b1a3ea6377adf2
 fa47bdb0bfdc2b0c | 6a65786a65786a65 | 76c16d20850c7974
 70b83523085fe820 | 6a65786a65786a65 | 1a773b33170f3582
 f4a4b2857512d2bc | 6a65786a65786a65 | 738804230a32c097
 c7e2937ffa2f95b9 | 6a65786a65786a65 | ed099fc280a6dd34
 f5e89f0539aef5dd | 6a65786a65786a65 | 387b259bdf0865a6
 d9de929ccfc611ca | 6a65786a65786a65 | 8f83a74846adcefb
 a8d6965214bec693 | 6a65786a65786a65 | 9f7ce7bfdf7f1821
 51817914ca2625fb | 6a65786a65786a65 | dfd40e213d02effd
 1a197b7755282185 | 6a65786a65786a65 | 946b0d029779efff
(10 rows)
```

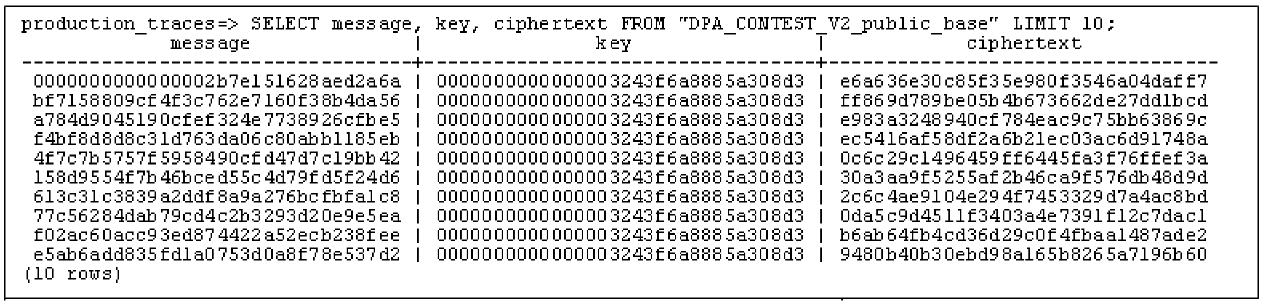
Figure 4: A sample database query for DES traces. For each trace, the message, key, and ciphertext is given. Opening a given trace gives the power information depicting power consumption during the 16 DES rounds.

The programmer may interact with the contest's power trace data in a couple of ways. As shown from the query and result, each trace comes with all of the data required to run an attack program. To utilize this data, the programmer has a couple of options; to use the PostGreSQL API or to download the traces remotely. Using the API, the programmer does not need to worry about anything other than calling the database and selecting the traces. However, this amount of work takes a significant amount of time during an attacks execution. Remotely downloading the traces improves this time, but some overhead is created as well when dealing directly with the power traces. A reference attack written in the Python programming language is available as a starting ground for power trace manipulation and attack. Attacks are accepted in a variety of programming languages, including C/C++, Python and MATLAB.

The contest's sole criterion for attack evaluation is the number of traces required to retrieve the key for the particular trace table. Along with this, it is required that the key guess is stable, meaning that the key guess must remain the same for 100 iterations of the attack. It is also noted that exhaustive search for the attack is not desirable unless it is only used to acquire the last eight bits of the key, where the remaining 48 bits were found using a power analysis attack. For more information pertaining to DPA Contest v1 and its rules and documentation, please refer to [10].

3.2 DPA Contest V2 - AES-128

With the success of the first contest, the DPA Contest v2 was released with several improvements over the previous DES-based contest. Among these changes is a switch to AES-128 as the cryptographic algorithm as well as the consideration for template-based attacks. New metrics have also been introduced for judging the attacks besides trace count; these include memory footprint, success rates and execution time. A different board, titled the "SASEBO GII" has been used to acquire the traces and its full hardware FPGA design is available. Traces are once again stored in tables in the COMELEC trace databaasse. Unlike the DES contest, which offered only public trace tables with fixed keys, DPA Contest v2 has three types of tables for use with the competition. A public trace table is available for standard attack testing and validation. Meanwhile, a private trace table will be used for the judging of attack implementations. Both the public and private trace tables utilize fixed keys, much like the first competition. Another new type of table, termed the profiling base, is available for contest participants wishing to profile the target device given the traces of random plaintexts and random keys for the cipher. An example query to the public trace table, along with its result, is given below.



```
production_traces=> SELECT message, key, ciphertext FROM "DPA_CONTEST_V2_public_base" LIMIT 10;
            message              |               key                |            ciphertext
---------------------------------+----------------------------------+----------------------------------
 0000000000000002b7e151628aed2a6a | 00000000000000003243f6a8885a308d3 | e6a636e30c85f35e980f3546a04daff7
 bf7158809cf4f3c762e7160f38b4da56 | 00000000000000003243f6a8885a308d3 | ff869d789be05b4b673662de27dd1bcd
 a784d9045190cfef324e7738926cfbe5 | 00000000000000003243f6a8885a308d3 | e983a3248940cf784eac9c75bb63869c
 f4bf8d8d8c31d763da06c80abb1185eb | 00000000000000003243f6a8885a308d3 | ec5416af58df2a6b21ec03ac6d91748a
 4f7c7b5757f5958490cfd47d7c19bb42 | 00000000000000003243f6a8885a308d3 | 0c6c29c1496459ff6445fa3f76ffef3a
 158d9554f7b46bced55c4d79fd5f24d6 | 00000000000000003243f6a8885a308d3 | 30a3aa9f5255af2b46ca9f576db48d9d
 613c31c3839a2ddf8a9a276bcfbfa1c8 | 00000000000000003243f6a8885a308d3 | 2c6c4ae9104e294f7453329d7a4ac8bd
 77c56284dab79cd4c2b3293d20e9e5ea | 00000000000000003243f6a8885a308d3 | 0da5c9d4511f3403a4e7391f12c7dac1
 f02ac60acc93ed874422a52ecb238fee | 00000000000000003243f6a8885a308d3 | b6ab64fb4cd36d29c0f4fbaa1487ade2
 e5ab6add835fd1a0753d0a8f78e537d2 | 00000000000000003243f6a8885a308d3 | 9480b40b30ebd98a165b8265a7196b60
(10 rows)
```

Figure 5: A sample database query for DES traces. For each trace, the message, key, and ciphertext is given. Opening a given trace gives the power information depicting power consumpton during the 8 AES rounds.

Another new development in the latest DPA Contest is a complete overhaul of how attacks are executed and submitted for judging. Similar to the DES-base contest, the AES-

128 contest contains a reference attack which demonstrates how a proper attack might work. The contest also features a modular software breakdown, with components termed the attack wrapper and result computer. The attack wrapper acts as the hub for the entire software package; communicating with the user for parameters, accessing the trace database or filesystem, executing the actual attack and finally shipping the result off to the result computer. The result computer in turn takes the data from the attack wrapper as input and processes it, giving the programmer output of their attack as well as a set of result metrics. The following block diagram depicts the operation of the attack wrapper and the result computer for a given attack.

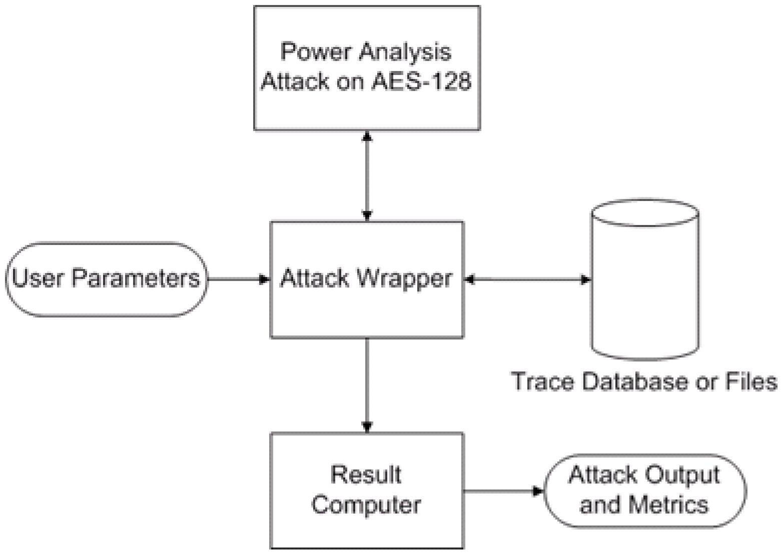


Figure 6: A block diagram outlining the attack wrapper for the DPA Contest AES contest.

The reference attack is written in C++ and it is encouraged that all participants also use a language in which the memory footprint and execution time can be brought to a minimum. These practices are set forth in order to maximize success in the contest with the new rules. However, submissions will still be allowed in a variety of programming languages, including all of the previously accepted languages such as Python and MATLAB. A reference attack is available, written in the Python programming language, which once again serves as a decent starting point for new researchers wishing to build attacks. However, the use of the python attack to submit to the contest is not recommended, as it takes much more time to execute over a C/C++ implementation. For more information on DPA Contest v2 as well as for further description of the rules and policies, please refer to [10].

4 Power Consumption of Cryptographic Devices

4.1 Power Models

A requirement of a differential power analysis attack which utilizes correlation as it's primary method of determining the key is that there be a model which can approximate the power consumption of a circuit during the encryption process. There are several different methods for constructing this power model. An accurate method for describing the power consumption of a device is to simulate the device in a software environment designed to show power consumption. If the target cryptographic device has an architecture that is known, then such a simulation may provide a precise prediction of the power consumption of the circuit during encryption. In instances where the target device's architecture is not entirely known or it is infeasible to simulate it, a more general power model must be used. The power models currently used in these instances are the Hamming weight and the Hamming distance models.

4.2 Hamming Weight Power Model

The Hamming weight model is the most basic power consumption model. It is most applicable to approximate the power consumption of a circuit utilizing a data or address bus, and relies on the basic premise that a bus will consume an amount of power that is proportional to the number of bits that are switched on within that bus. If no bits are switched on, the bus will consume very little power compared to a data bus with all bits switched on. An example is shown in Figure 7.

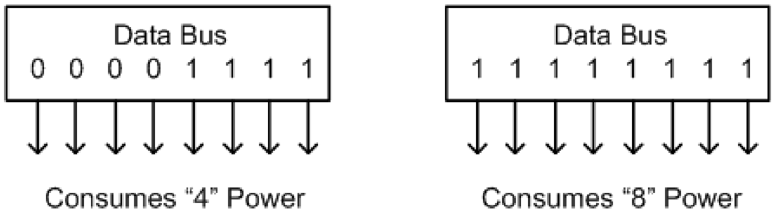


Figure 7: An example showing the Hamming weight model. The Hamming weight of the value on the data bus is taken to be proportional to the power consumption of the bus.

This model can be used when very little information is known about the circuit that the attack is being performed on. It also carries an advantage over the Hamming distance model in that it does not require any information about the data on the bus during the encryption routine. However, the model only weakly describes power consumption within a circuit. While differential power analysis attacks are possible using the HW model, they are not as efficient compared to the Hamming distance model. In general, whenever some information is known about the target circuit's implementation the HD model should be used[3].

4.3 Hamming Distance Power Model

The Hamming distance model is an extension of the Hamming weight model which uses *changes* in logic values in a certain time interval to determine power usage. The change can occur in many different circuit components such as a data or address bus, register, memory, or some other component. Using this model, you can determine the approximate power consumption of a circuit as being proportional to the number of 0→1 and 1→0 transitions made within the circuit. The number of bit transitions is simply the Hamming weight of the exclusive or of the two values. For example, the Hamming distance between two register values (R0 and R1) is given as:

$$HD(R0, R1) = HW(R0 \oplus R1)$$

Several basic assumptions are made when this power model is used. It is assumed that bits which do not change (0→0, 1→1) do not contribute to the power consumption of a circuit. It is also assumed that a 0→1 and 1→0 transition consume an equal amount of power. In most circuits this may be found to not be the case. If more information is known about the specific power consumption of the device, minor enhancements can be made to the model by weighting transitions differently. An example showing how a register updating it's value on a clock edge is shown in Figure 8.

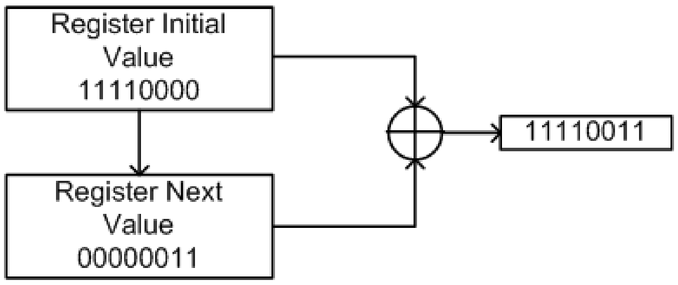


Figure 8: An example showing the Hamming Distance model.

As the register updates from one state to the next, it consumes a certain amount of power. In this example, the Hamming distance shows that the power consumed when the register updates is proportional to 6. Regardless of the weaknesses, the Hamming distance model provides a very convenient method for determining the expected power consumption of a circuit during a certain time frame. In any case where a change in data can be observed, the Hamming distance model should be used over the Hamming weight model.

5 Data Encryption Standard

5.1 Description

The Data Encryption Standard (DES) is a commonly known block cipher employed in many applications dating back to the late 1970's. It was first chosen as an official Federal Information Processing Standard (FIPS) by the National Institute of Standards and Technology (NIST) in the United States in 1976. The standard required NIST to review the algorithm every five years to determine whether it was still approved for use. It was reviewed and reaffirmed for unclassified use in 1983, 1988, and 1993. During this time, DES saw widespread use in multiple industries. The American Bankers Association (ABA) utilized DES as a standard to protect communication in automatic teller machines, point-of-sale terminals, and other wholesale electronic fund transfers (where it was used to protect transfers exceeding $1-2 trillion per day). DES also saw use in the telecommunications industryin the 1980's as a standard for data communcations encryption [11].

As computing power has increased and new attacks have been formulated, DES has become less secure. The FIPS approval in 1993 indicated the intention to replace the algorithm in the near future. In 1999, the standard was reviewed once again, and triple DES was approved for use until a new standard could be developed and approved. Even though it is not considered as secure as it once was, DES and variations of DES are still used in certain applications. It's broad use and implementations make DES an appropriate target for demonstrating power analysis side-channel attacks. Figure 9 depicts the DES encryption scheme with the standard parameters.

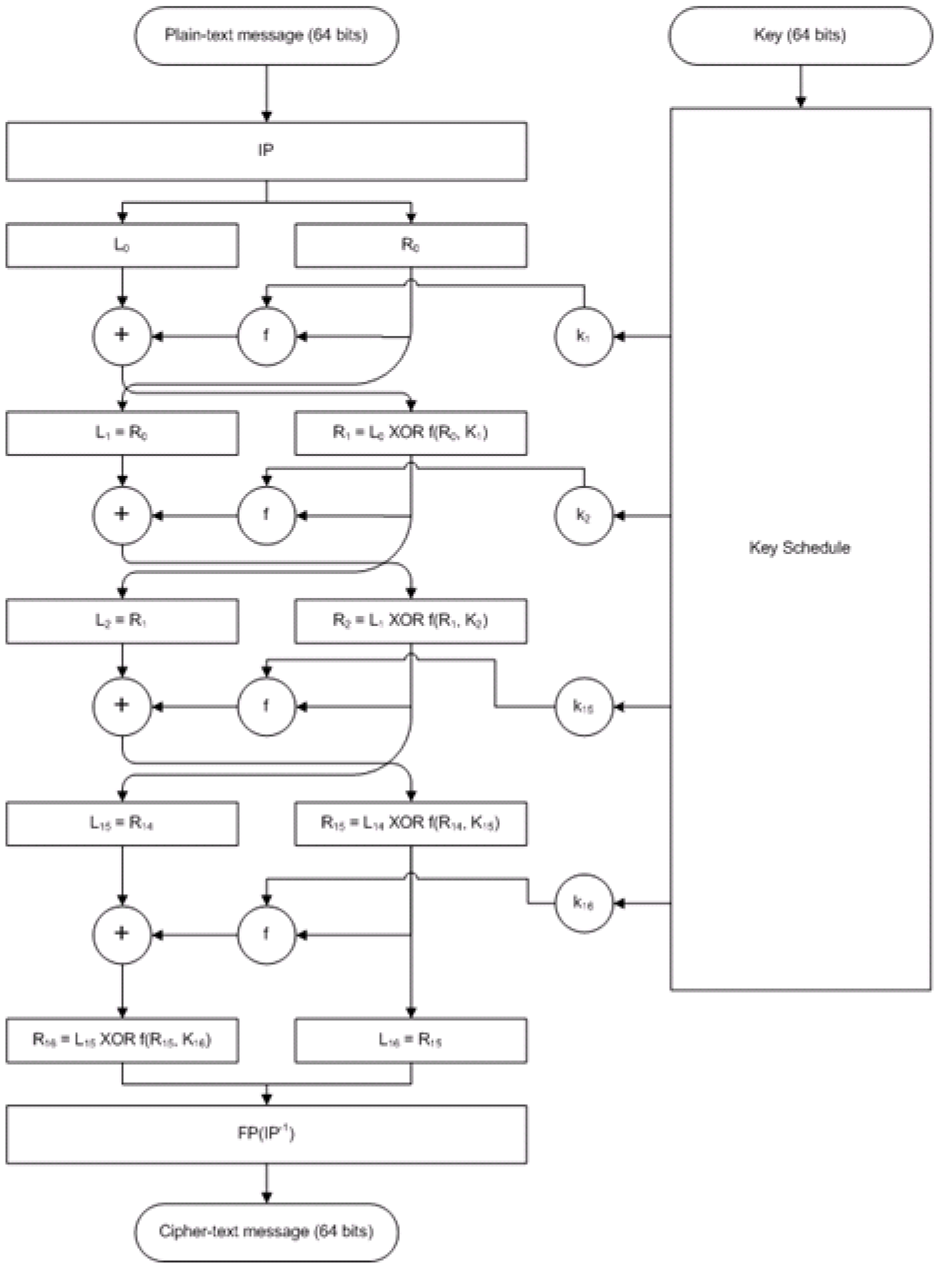

Plain-text message (64 bits)
Key (64 bits)
IP
L_0
R_0
f
k_1
$L_1 = R_0$
$R_1 = L_0$ XOR f(R_0, K_1)
k_2
Key Schedule
$L_2 = R_1$
$R_2 = L_1$ XOR f(R_1, K_2)
k_{15}
$L_{15} = R_{14}$
$R_{15} = L_{14}$ XOR f(R_{14}, K_{15})
k_{16}
$R_{16} = L_{15}$ XOR f(R_{15}, K_{16})
$L_{16} = R_{15}$
FP(IP^{-1})
Cipher-text message (64 bits)

Figure 9: Flow chart depicting the operation of the DES encrpytion

As shown above, DES operates on a 64-bit message which we will term the plaintext and a 64-bit key (8 bits of which are reserved for parity). The 64-bit plaintext is run through a permutation matrix labeled *initial permutation* (IP). From there it is broken into two 32-bit halves and run through what is known as a Feistel network for 16 rounds. Finally the message is passed through the *final permutation* (FP) which is just the inverse of the initial permutation. The resulting 64-bit message is now known as the ciphertext; this constitutes a complete encryption. To decrypt the ciphertext, one reverses the algorithm to recover the plaintext. The 48-bit subkeys are handled by the key scheduler, and this allows for a different 48-bit subkey to be created from the overall 64-bit key for each round of DES operations. For a more complete description and explanation of DES, please refer to [4].

5.2 Power Characteristics

Before any side-channel attack on the DES implementation used by the DPA contest can be mounted, we must first ensure that the power consumption of the circuit can be approximated with one of the models described. If the Hamming weight or Hamming distance model can determine the expected power consumption of the DES circuit, then that opens up the possibility of an attack on the system. To determine whether such a weakness exists, we refer to the DES implementation used by the DPA Contest trace aquisition board, specifically described in [9]. Figure 10 shows how the DES routine is implemented on the trace aquisition board.

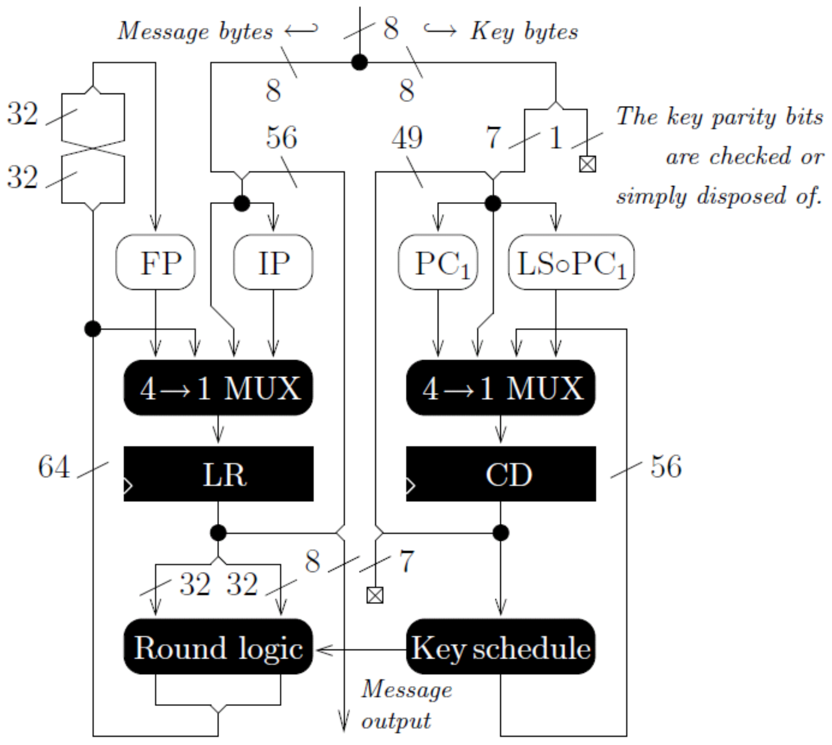


Figure 10: The DPA Contest trace aquisition board DES co-processor layout. Designed for parallel execution scheduled at one round per clock cycle. [9]

From this figure, we can see that there is a single data register (LR) that gets updated each round and which holds the intermediate results of the DES operation. This register is a prime target for the Hamming distance power model. If the model is accurate, then the Hamming distance between the initial value of the register and the next value of the register (after the DES round is complete) will indicate a portion of the power consumption of the circuit. This Hamming distance plays the role of the *sensitive data* in our attack. A single power trace from the DPA Contest database contains the necessary encryption information (plaintext, key, and ciphertext) as well as the power information. Because the plaintext and key are known, we can choose any intermediate round, determine the Hamming distance between the values of this register at the start and end of that round, and use that as our sensitive data. However, in the unkown key situation only the first and last rounds are vulnerable to this basic method of attack. This is because either the initial value or the post-round value of LR must be known to retrieve the key. If we choose an arbitrary round, we would have to guess at both of these values. For our power consumption profiling, we demonstrate the correlation between the first round sensitive data and the power information contained in the power trace. This power profiling was done entirely in Matlab.

The first step in the power profiling is to determine the sensitive data that will have some relationship to the power information. We chose to look at the first round input and output to find this sensitive information. Initially, we know the LR register holds the 64-bit plaintext message. The right 32 bits of this message are called $R1$, and contain the bits that are input into the Feistel function (together with the first round subkey) during the DES encryption. After the first round of DES is complete, the LR register is updated with its new value. The right 32 bits of the now updated register are called $R2$. Only the right 32-bit portion of the LR register is used in the power profiling because it is only these bits which are dependent on the key. This is important for the unknown key scenario where we will need to determine the key bits used. After this first round, we have the two pieces of information that we need in order to perform our power analysis. The Hamming distance between $R1$ and $R2$ becomes the sensitive data for the program.

The next step in the power profiling is to take the power information from the power traces. Each DPA Contest DES power trace consists of 20000 points of power information taken at a rate of 20×10^9 samples/s. We consider a given trace $P_i = (p_i(t_1), p_i(t_2), ..., p_i(t_m))$ to be a collection of such points indicating the voltage of the circuit at time t_i. By a *power trace*, we will mean this collection of points. Just by looking at the trace, the 16 power peaks corresponding to the 16 rounds of DES can be identified (refer to Figure 3). We take the local maximum point in the first round peak to be our power information, p_{max}. This point is the maximum point between t_{5700} and t_{5800}. The first goal in our power profiling is to show that there is a clear relationship between the power information and the sensitive data. We ran through 4000 power traces, retrieving the first round power consumption peak (p_{max}) and the first round sensitive data for each trace ($HW(R1 \oplus R2)$). For each sensitive data value (from 0 to 32), we take the average of the power consumption peak values across all power traces which contain that sensitive data value. The resulting plot is shown in Figure 11.

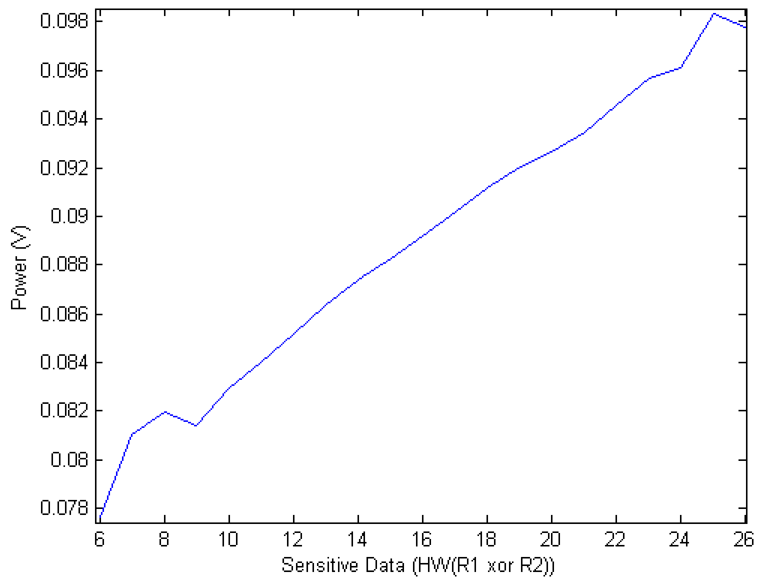


Figure 11: Power consumption versus Hamming distance.

The results show a clear relationship between the sensitive data and the power consumption of the circuit during the first DES round. In general, a higher Hamming distance results in more power consumed in the DES encryption process. This verifies the utility of the Hamming distance model for this specific DES implementation. Based on this information, it is possible to determine a sample correlation between the power consumption and the Hamming distance data. We calculate the correlation between the power trace and the sensitive data for the given trace using Pearson's sample correlation coefficient, given by:

$$r_{sb} = \frac{\sum_{i=1}^{n}(h_i^{(sb)} - \overline{h^{(sb)}})(p_i(t_j) - \overline{p(t_j)})}{(n-1)\sigma_h \sigma_{p(t_j)}} \tag{1}$$

The mean value across all traces of a point t_j is given as $\overline{p(t_j)}$ and the standard deviation of these values at a point t_j is given as $\sigma_{p(t_j)}$. For each trace i, there exists sensitive data which can be exploited. The sensitive data for a trace i is denoted as h_i, with the mean sensitive data over all traces $\overline{h_i}$, and standard deviation σ_h. Using this equation to correlate the power information to the first round sensitive data, there should exist a spike in correlation around the first round. To test this, we calculate this coefficient across every point (j = 1 to 20000) in each power trace for 1000 total power traces (the correlation improves as more traces are used, 1000 traces is sufficient to produce high correlation values). The results are shown in Figure 12.

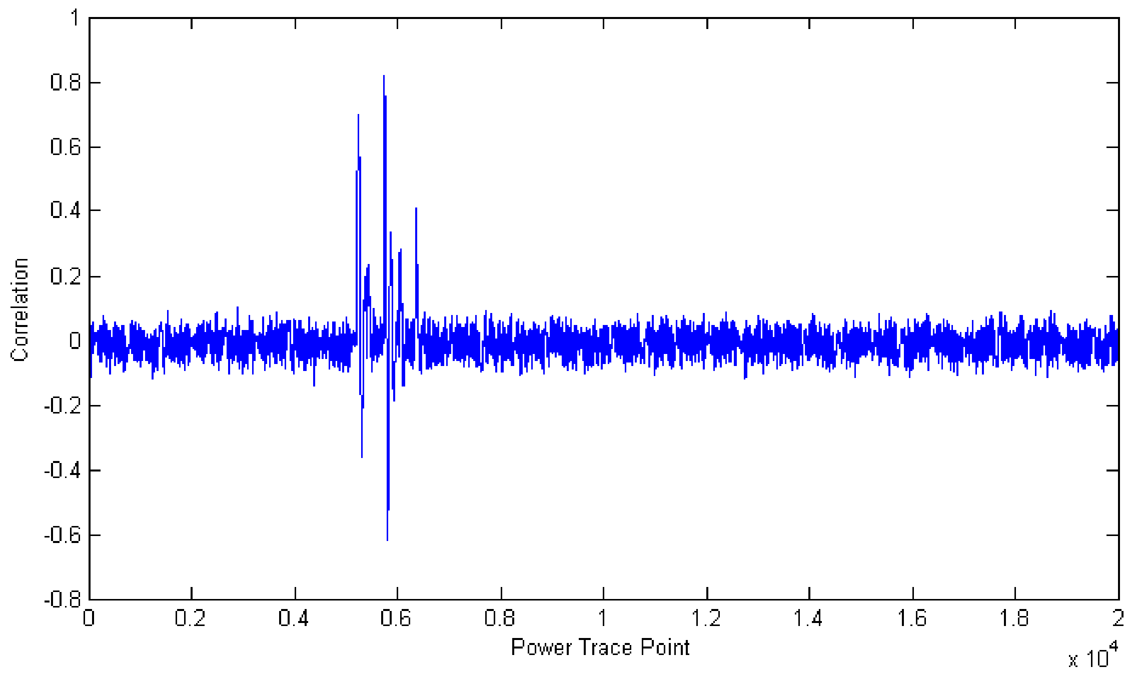


Figure 12: The correlation between the first round sensitive data and the power traces for each point j in the power traces.

From this figure, it is apparent that there is a strong correlation between the sensitive data and the power trace centered around j = 5750. This further verifies the validity of the Hamming distance model in relation to the DES implementation used. However, it is still not completely adequate to say that an attack is possible based on this information. To further ensure that we can recover key bits, it is necessary to show that the correlation is still strong given only one half-byte of sensitive data (corresponding to a single S-box output). To do this, we repeat the correlation calculation over all of the traces using only the first S-box output as the sensitive data. Even with a reduction in sensitive data, there is still a strong correlation between the data and the power information, as shown in Figure 13.

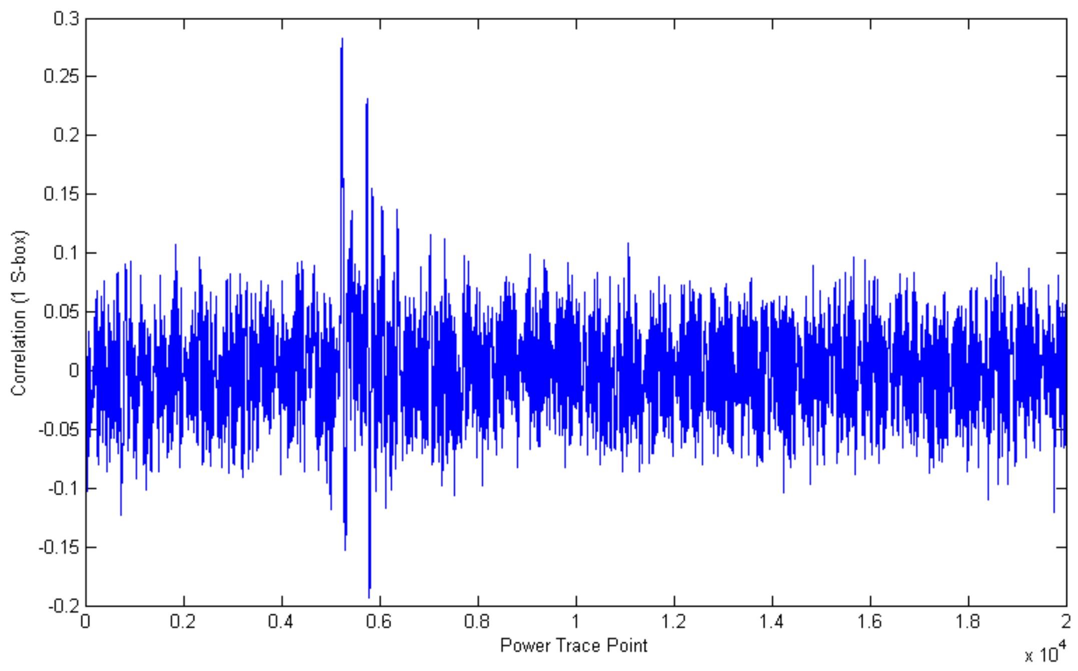


Figure 13: The correlation between the first round sensitive data and the power traces for each point j in the power trace.

Based on the information gained from this power profiling, we can conclude that a basic differential power analysis attack is possible on the DES system used to create the DPA Contest power traces. The next section outlines the details of how to retrieve key bits from the power traces using this information.

5.3 Basic DPA Attack

The basic differential power analysis attack described here relies on two properties of the DES encryption. The first is the fact that the individual DES S-box outputs produce sensitive data that can be correlated to the power information recorded from the target device. The second fact is that each DES S-box input uses only 6 bits of the 48-bit subkey used for that round, which is low enough that they may be exhaustively searched to find the highest correlation. The purpose of the attack implemented is to retrieve the first round subkey used. Once enough of this subkey is determined, the full key used in the DES encryption can be found by exhaustive search methods (if desired). The entire attack program was written in Matlab.

The DPA attack program is set up similarly to the power characteristics program, however this time we must guess the key bits used and select those that produce the highest correlation for each S-box. For each S-box, we select the 6 bits from the plaintext that are used which is dictated by the initial permutation and the expansion permutation inside the Feistel function. For example, in the case of the first S-box these bits are (7, 57, 49, 41, 33 25) of the plaintext. Because the output of the S-box is dependent on the unknown 6-bit key, we simply produce an array of 2^6 potential S-box outputs, one for each key guess (from 0 to 63). For each potential 4 bit S-box output, we calculate the Hamming distance between that output and the corresponding bits in the plaintext (e.g. for key guess 12, $HW[R2_{12}(1:4) \oplus R1(1:4)]$, where we have used $R2_{12}$ to denote the potential value of the register using key guess 12). The result is an array of 64 sensitive data values, one for each potential key. Using the correlation coefficient equation (equation 1), we calculate the correlation between each one of the 64 sensitive data values and the power information. This process is repeated using many power traces to build up the correlation coefficient. After a sufficient number of traces, there will exist a peak in the correlation which corresponds to the correct key. The results for a single S-box after 500 power traces are seen in Figure 14. This shows the correct key guess as a spike in the correlation at guess 14 (which tells us the correct subkey for this S-box is 13).

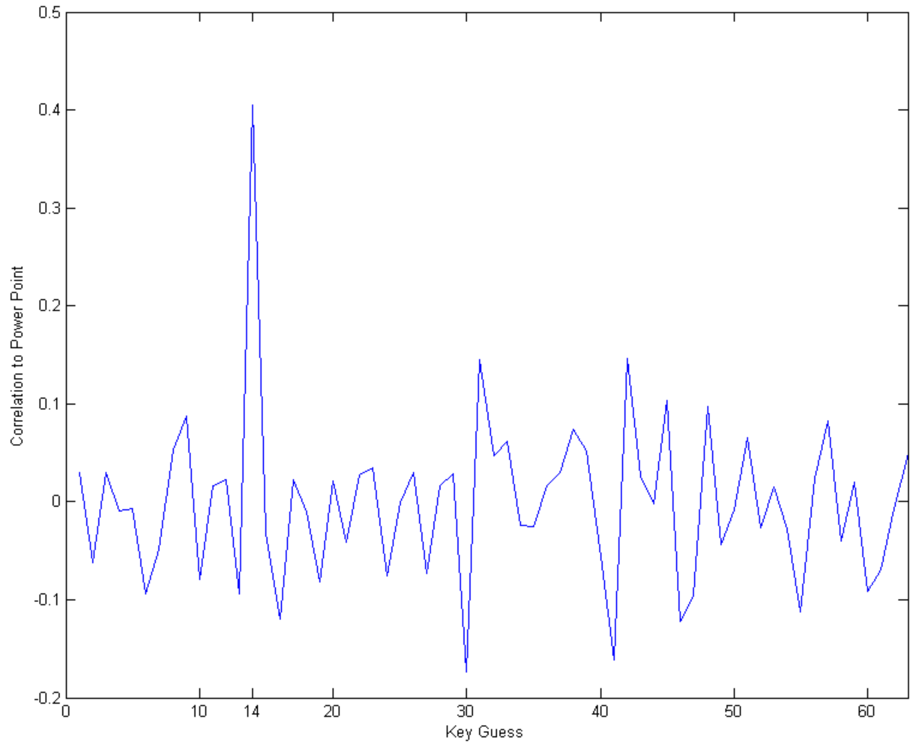


Figure 14: The correlation between the power information and the sensitive data for each 6-bit key guess in a single S-box.

The process is repeated for each S-box to create a table of correlation coefficients containing 64 entries for all 8 S-boxes. For each S-box, the key guess with the maxmimum correlation is the correct key. An example plot for all 8 S-boxes is shown in Figure 15. This shows the correct 6-bit key values for the first round subkey (56 11 59 38 0 13 25 55) as peaks in the correlation.

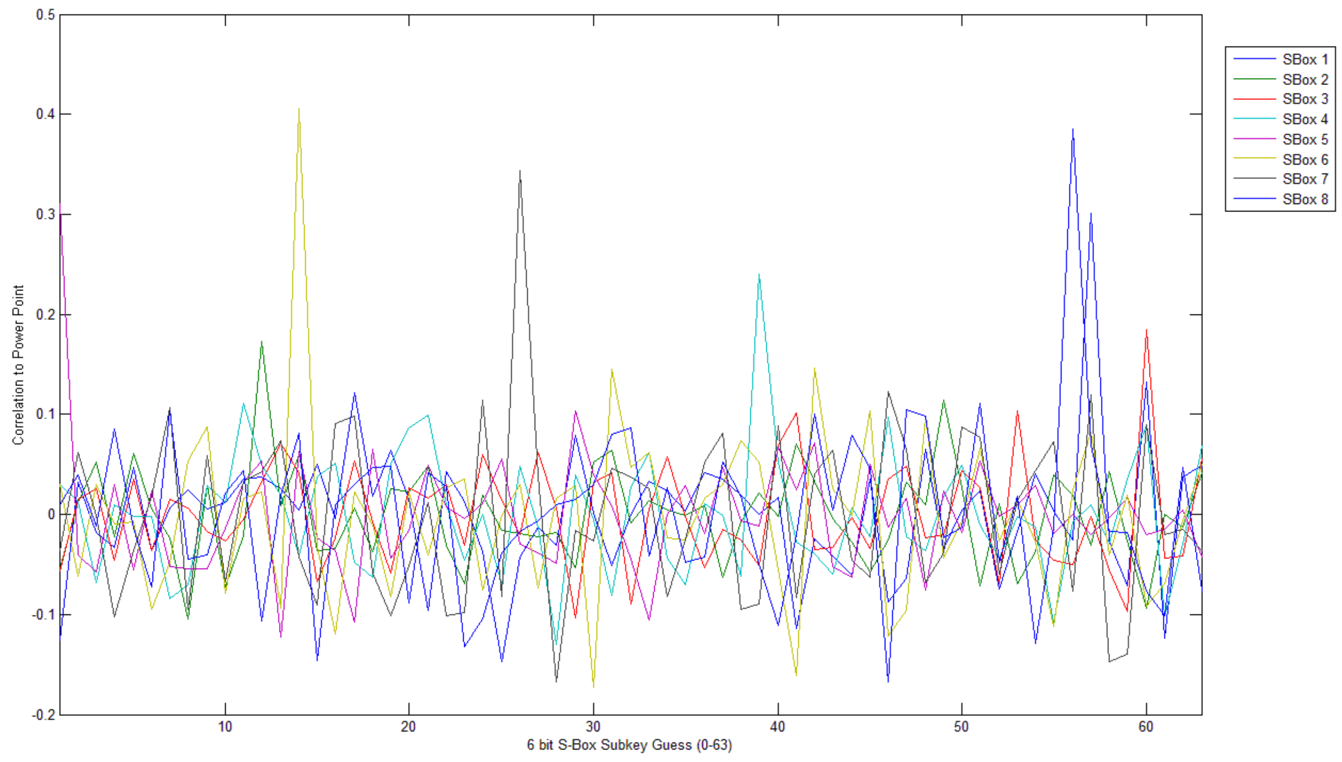
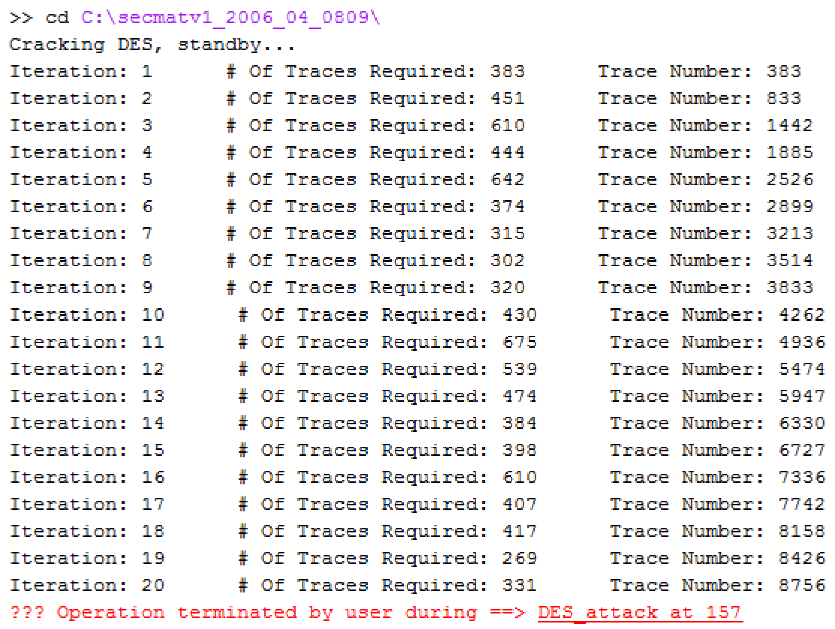


Figure 15: The correlation between the sensitive data and the power information for all 8 S-boxes.

Once the program has settled on a key for a certain number of power traces (this threshold is arbitrarily set to 100 for the DPA Contest [10]), the program will output to the Matlab console the subkey, the amount of time taken to determine the correct subkey, and the number of traces required to determine the key. An example of the program output is shown in Figure 16. The DPA Contest power trace database consisted of 80000 trace files. Running over a large number of traces, our program was able to find the subkey using an average of 455 traces. The maximum number of traces required to find the key was 826, and the minimum number of traces required was 269. The DPA Contest reference attack had an minimum average trace count of 485. The more advanced entries in the first DPA Contest were able to reduce the average trace count to as little as 141 traces using more advanced statistical methods. The average time to discover the subkey was between 3 and 4 minutes. This is especially fast compared to the fastest exhaustive search on a DES encrypted message: reportedly, 22 hours and 15 minutes by Deep Crack in 1999 [7].



```
>> cd C:\secmatv1_2006_04_0809\
Cracking DES, standby...
Iteration: 1      # Of Traces Required: 383      Trace Number: 383
Iteration: 2      # Of Traces Required: 451      Trace Number: 833
Iteration: 3      # Of Traces Required: 610      Trace Number: 1442
Iteration: 4      # Of Traces Required: 444      Trace Number: 1885
Iteration: 5      # Of Traces Required: 642      Trace Number: 2526
Iteration: 6      # Of Traces Required: 374      Trace Number: 2899
Iteration: 7      # Of Traces Required: 315      Trace Number: 3213
Iteration: 8      # Of Traces Required: 302      Trace Number: 3514
Iteration: 9      # Of Traces Required: 320      Trace Number: 3833
Iteration: 10      # Of Traces Required: 430      Trace Number: 4262
Iteration: 11      # Of Traces Required: 675      Trace Number: 4936
Iteration: 12      # Of Traces Required: 539      Trace Number: 5474
Iteration: 13      # Of Traces Required: 474      Trace Number: 5947
Iteration: 14      # Of Traces Required: 384      Trace Number: 6330
Iteration: 15      # Of Traces Required: 398      Trace Number: 6727
Iteration: 16      # Of Traces Required: 610      Trace Number: 7336
Iteration: 17      # Of Traces Required: 407      Trace Number: 7742
Iteration: 18      # Of Traces Required: 417      Trace Number: 8158
Iteration: 19      # Of Traces Required: 269      Trace Number: 8426
Iteration: 20      # Of Traces Required: 331      Trace Number: 8756
??? Operation terminated by user during ==> DES_attack at 157
```

Figure 16: Example output from the DES DPA attack program.

The generic DPA attack described in this section carries with it a number of advantages. The attack can be mounted in a known-plaintext only or known-ciphertext only scenario, which is innately better than attacks that require plaintext-ciphertext pairs. The DPA attack is also extremely quick compared to exhaustive search methods, taking mere minutes as opposed to several days for exhaustive key searches. The downside to the DPA attack is that it requires physical access to the device and requires a reliable method of aquiring relatively noise free power traces from the device. Noise can be reduced in the power trace aquisition by taking many samples using the same plaintext and key and averaging them, as described in [3]. In the next section, we describe the Advanced Encryption Standard (AES) and discuss how the same basic method of attack can be utilized to discover key bits used in AES encryptions.

6 Advanced Encryption Standard

6.1 Description

The Advanced Encryption Standard (AES) is a collection of three separate block ciphers; AES-128, AES-192, AES-256 and is the successor to DES. It was selected as an encryption standard by NIST as a Federal Information Processing Standard in 2001 after a lengthy competition. The cipher was originally developed by Vincent Rijmen and Joan Daemen, two Belgian cryptographers, and was first published in 1998. The three separate block ciphers are ordered in terms of the key size that can be used for encryption and decryption.

AES has become the popular replacement for its predecessor and is currently used in a variety of applications where security is a concern. These properties make AES another good candidate for work with power analysis side-channel attacks. The following flow chart depicts a typical AES-128 cipher in operation.

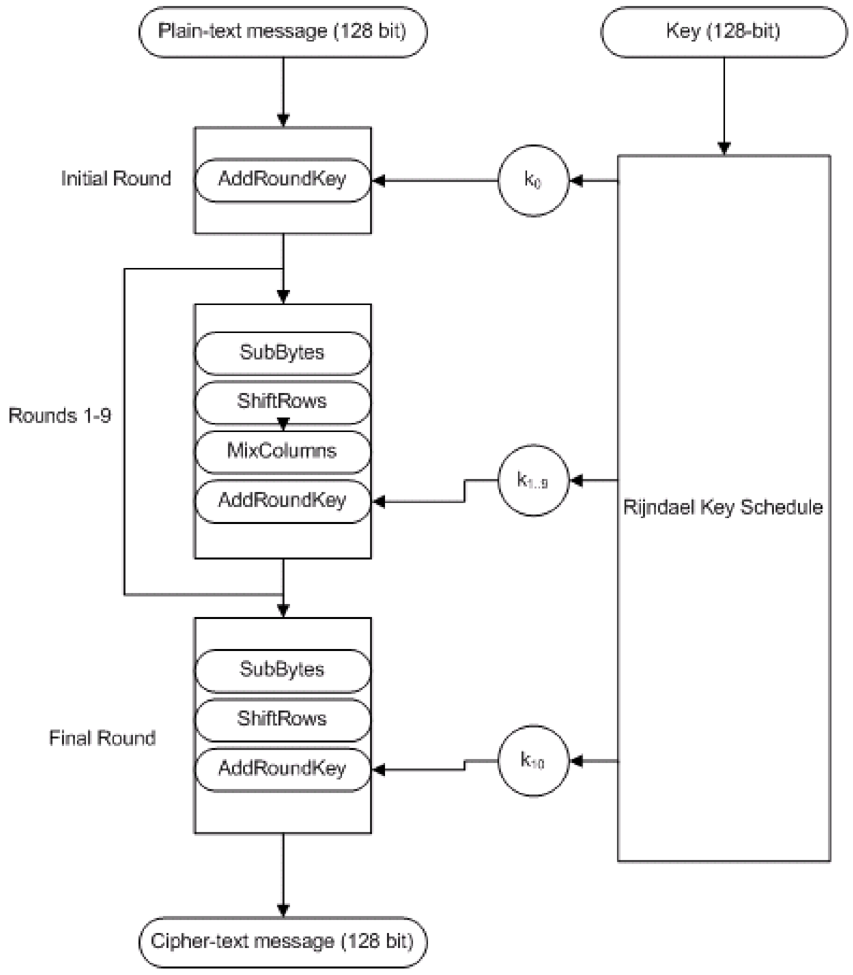


Figure 17: A flow chart showing the operation of the AES-128 encryption.

As shown above, an AES-128 cipher starts with a 128-bit block of plaintext along with a 128-bit key and executes ten AES rounds to compute a 128-bit ciphertext. Unlike its predecessor DES which is based on a Feistel network, AES is based on a substitution permutation network. Each stage of each individual AES round is computed on a 4x4 byte array known as the state. Within each round, AES consists of four main stages; SubBytes, ShiftRows, MixColumns and AddRoundKey. AES uses the Rijndael key schedule for its key scheduling to allow for a different sub-key to be generated for each round. For a more complete description and explanation of AES, please refer to [6]. The next section outlines how a standard AES implementation is vulnerable to the same basic DPA attack that we used on DES.

6.2 Power Characteristics

Similar to DES, in order to determine whether a differential power analysis attack on the AES system is possible we must first verify that the power consumption can be approximated using some power model. Before even implementing any programs to determine whether the Hamming distance power model will apply to AES we can make several assertions about the power consumption. First, based on the implementation given in Figure 18, it is clear that the plaintext is exclusive or'd with the first 16 bytes of the expanded AES key before being loaded into the register. This effectively limits our ability to attack the first round using the plaintext with the Hamming distance model, as the model relies on seeing a switch in register values where one side of the switch is known.

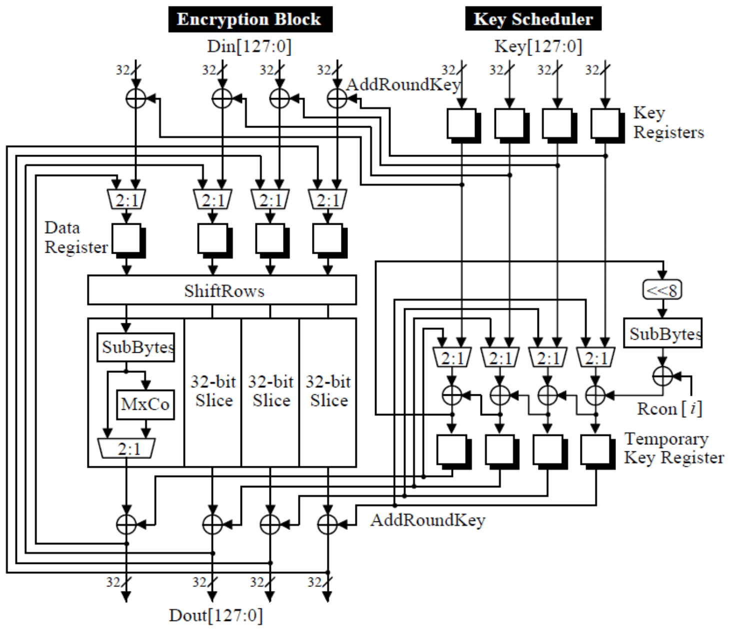


Figure 18: The implementation of AES-128 on the SASEBO aquisition board. [8]

Another initial issue seen in this implementation is that the power consumption caused by the data registers updating will contribute significantly less to the overall power consumption of the circuit than in the DES implementation. This is because an AES round will consume a fair amount of power (especially during the SubBytes step) compared to a DES round. This means the correlation computed using the Hamming distance model should be considerably lower than the correlation computed in the DES power characterization. Based on this information we can say that our attack must be run on the final round of encryption and that our attack will take significantly more traces than the DES attack. An example power trace taken from the DPA Contest trace database is shown in Figure 19. This clearly shows the 10 rounds of AES-128 encryption.

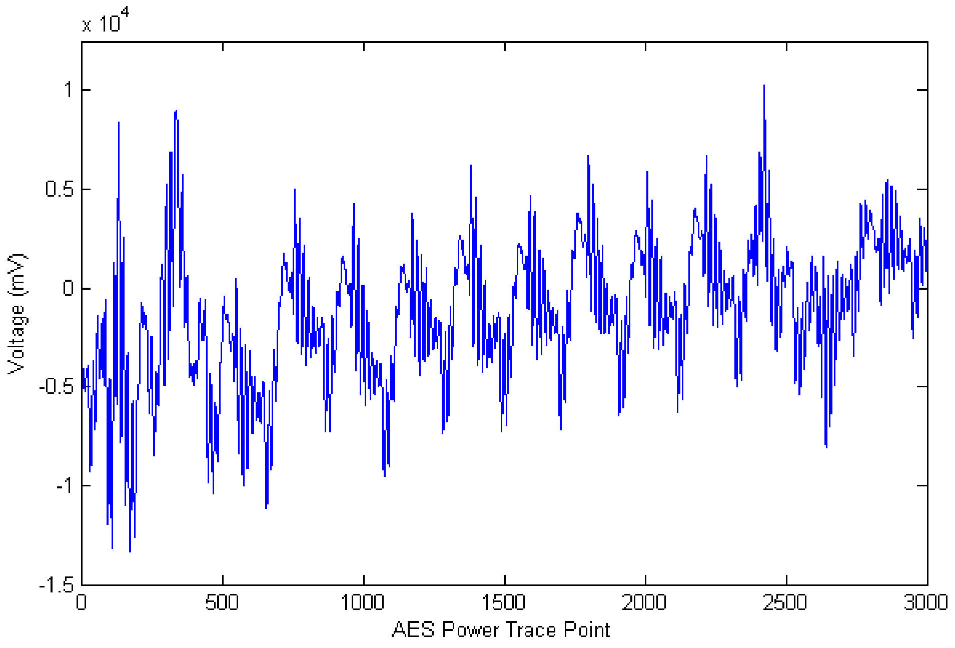


Figure 19: A sample AES power trace showing the power consumption over all rounds of an AES-128 encryption.

The first step in power characterization is similar to that in DES. We must show that there is some relationship between the Hamming distance of the bits in the data registers before and after the 10th round of encryption changes and the power consumption of the circuit. Initially, the 4 data registers hold the output of round 9 of the AES encryption. Because we know the key, we can determine the value of the data in these registers at this time ($R9$). After the 10th round of encryption, the value in these registers is updated to the final ciphertext ($R10$). The Hamming distance between the R9 and R10 values becomes the sensitive data which we will correlate to the power consumption. The process for showing this relationship was the same is in the DES power characterization. First, we run through many AES traces and determine the sensitive data value for each trace. We also take the average power consumption of the AES circuit during the final round for each trace. For each sensitive data value, we plot the corresponding average power value. Figure 20 shows the results over 19000 traces.

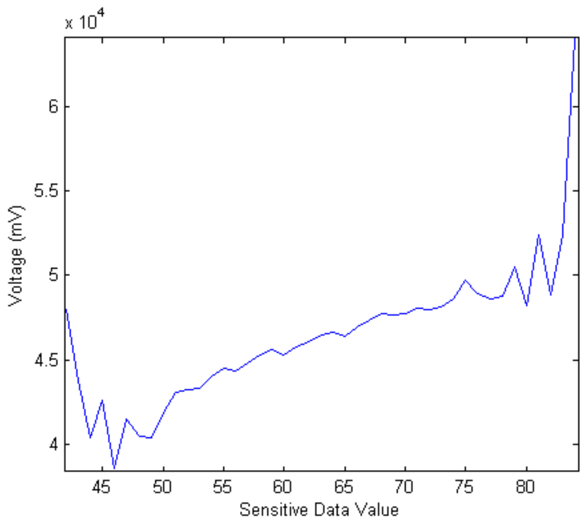


Figure 20: The relationship between the sensitive data and the power usage for the AES-128 circuit, including all points.

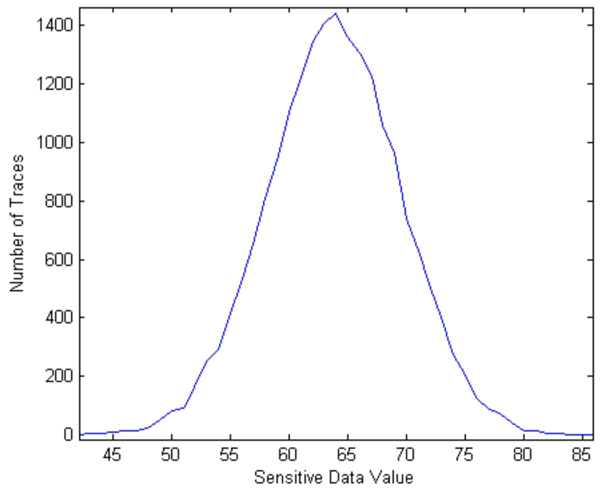


Figure 21: The distribution of sensitive data in the AES traces, showing the number of power traces at each sensitive data value.

Towards the edges of the graph the values appear erratic, while towards the middle there is a clear trend upwards. The edges don't follow the same trend because there are relatively few power traces which have sensitive data values that are lower than 50 and higher than 77 (with none below 42 or above 84, as shown on the distribution graph in Figure 21). This results in points which do not have enough power trace values. This results in a greater effect on the average value by more extreme power values at those sensitive data points. By replotting the data using only those sensitive data values with large numbers of power traces, we can see more clearly the general trend. We chose a lower bound of 50 and an upper bound of 77 (both contain at least 80 power traces) and replotted the resulting data, shown in Figure 22. This shows a clear upward trend in the relationship between sensitive data and power consumption, and implies that an attack on the circuit utilizing correlation with the Hamming distance model may be possible.

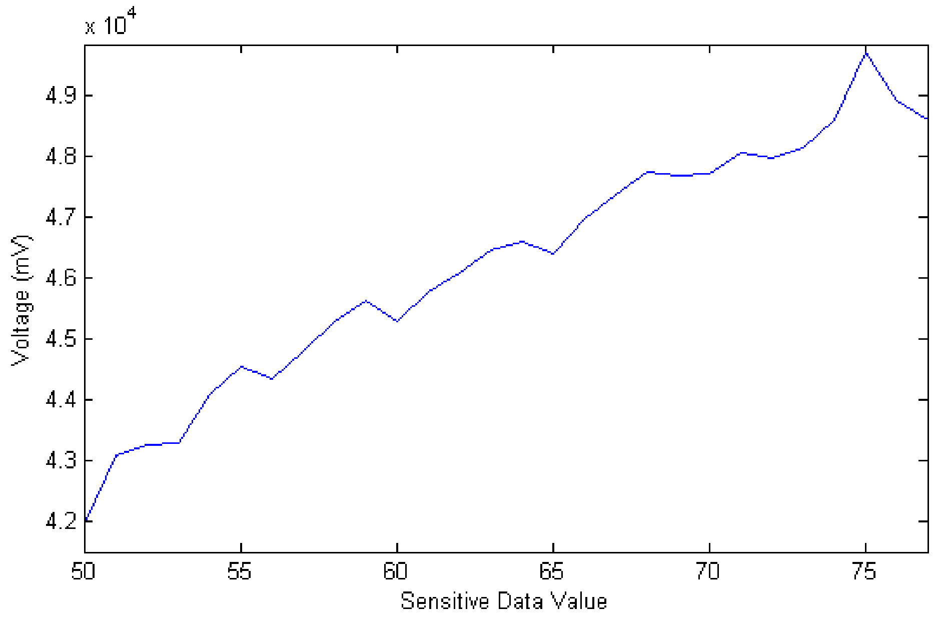


Figure 22: The relationship between the sensitive data and the power usage for the AES-128 circuit, including only those points with large numbers of values to average.

The next step in our power consumption profiling is to build up a sample correlation between the sensitive data and the power consumption using a similar method that was used in the DES power characterization. Using the Pearson sample correlation coefficient equation (Equation 1), we can determine this correlation using the AES-128 power traces provided by the DPA Contest database. For the entire 16 bytes of the data register updating, there is a strong correlation between the data and the power usage in the final round, as seen in Figure 23. Compared to the correlation found in the DES system, the peak of the correlation is much smaller. However, because there is a definite peak compared to the rest of the plot, we are not concerned about this.

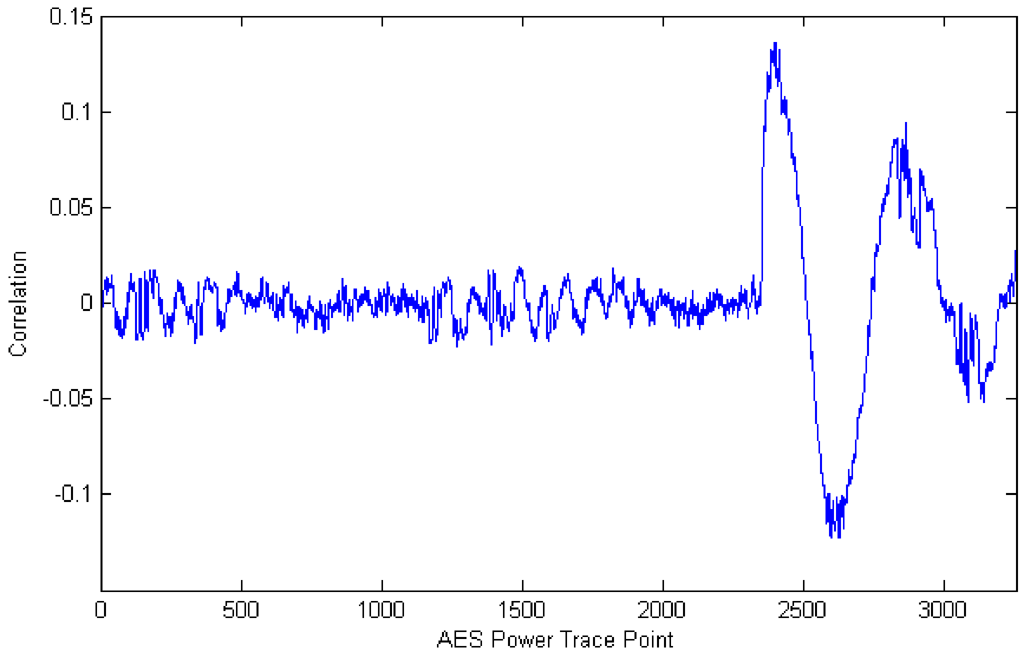


Figure 23: The correlation of sensitive data and power consumption for all 16 bytes of a message.

This figure also shows that there is a broad region on the power trace where the power consumption correlates strongly to the data (compared to the DES trace in Figure 12 where there is a spike in correlation rather than a region). This will become important when we need to determine which point (or collection of points) on the power trace to correlate to the sensitive data for the DPA attack. There are several options which we can use in the attack scenario. We can use:

- The peak of the power consumption.
- The integral of the power consumption over some number of power trace points.
- The sum of squares of the power consumption over some number of points.
- The sum of absolute values of the power consumption over a certain number of points.

While all of these choices may work, one may slightly reduce the number of traces required for a successful attack over the others. [3]

To have the ability to determine the subkey used in the final round of encryption, we must see a clear correlation between the sensitive data and the power information for only one byte at a time in the data register. As one byte updates, it needs to have a significant enough effect on the power consumption of the circuit as a whole that there is an increase in the correlation coefficient. This is shown to be true in Figure 24, where we can see that even with a reduction in the amount of sensitive data we can still see a peak in the correlation around the time of the final round.

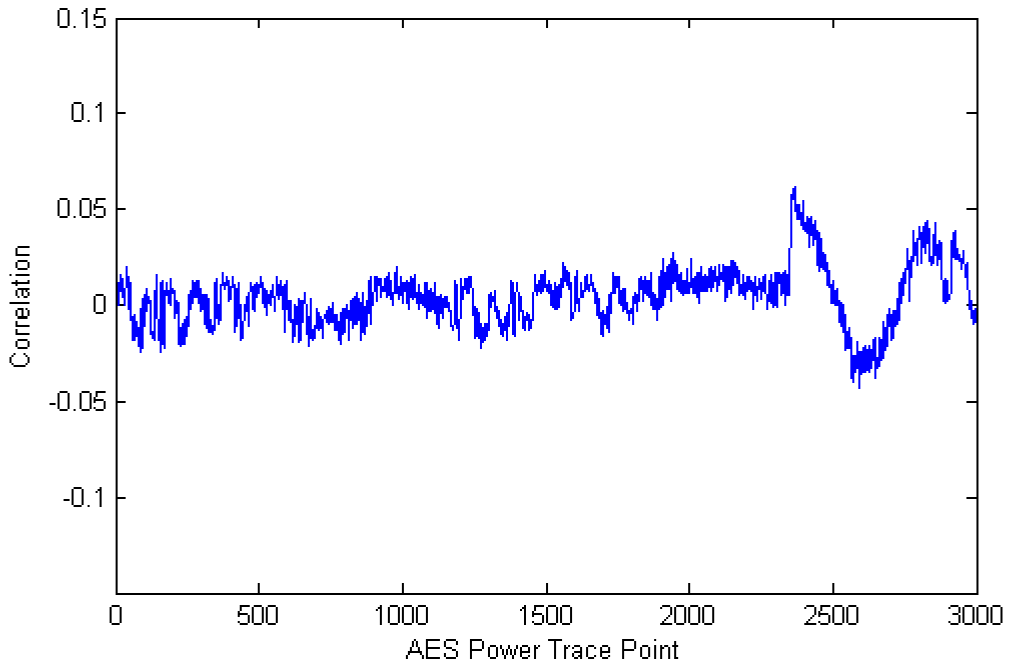


Figure 24: The correlation of sensitive data and power consumption for a single byte of a message over 10000 traces.

Our initial intuition that the AES circuit will require far more traces to successfully discover the key becomes more apparent here. Our other initial claim that the Hamming distance model would not as accurately describe the power consumption of the circuit is also supported. However, although the correlation peak is far less than that in the DES example (Figure 13), it still stands out enough from the rest of the power trace and therefore can be used in an attack. The next section describes how we discover the key using the Hamming distance power model.

6.3 Basic DPA Attack

Based on our knowledge that each individual byte in the 16 byte message of the AES-128 encryption can show a peak in the correlation to the power traces; we developed an attack program in Matlab. This program could succesfully determine the key used in the final round of encryption. One benefit of attacking the final round in AES is that the MixColumns step is skipped in this round, making the attack faster in terms of computation. The attack is similar to the one used to break the DES system. However it targets the ciphertext side rather than the plaintext side of encryption.

The first step in the AES attack program is to split the 128-bit ciphertext message into byte long blocks. The AES-128 algorithm operates on each byte individually, which allows us to guess the 8-bit portion of the round key used for each byte individually. Focusing on one byte at a time, we take each byte of the ciphertext and run through the AES decryption method 256 times (once for each possible key). The result is an array of 256 potential values for the final round input. For each potential byte value, we take the Hamming distance between it and the known ciphertext byte that corresponds to the same location in the data register (these are shifted as a result of the ShiftRows step in the AES algorithm). We are left with an array of 256 Hamming distances, which approximately model the power consumption of the circuit for each possible key value. The final step is to correlate these Hamming distance values to the actual AES power traces to determine which key is actually correct. An example of the resulting correlation for a single byte is shown in Figure 25. The graph shows that there is a peak in the correlation at key guess 160, which tells us that the 8-bit key used for that byte was 159. Matlab array indexes start at 1, so the key is taken as the index value minus one.

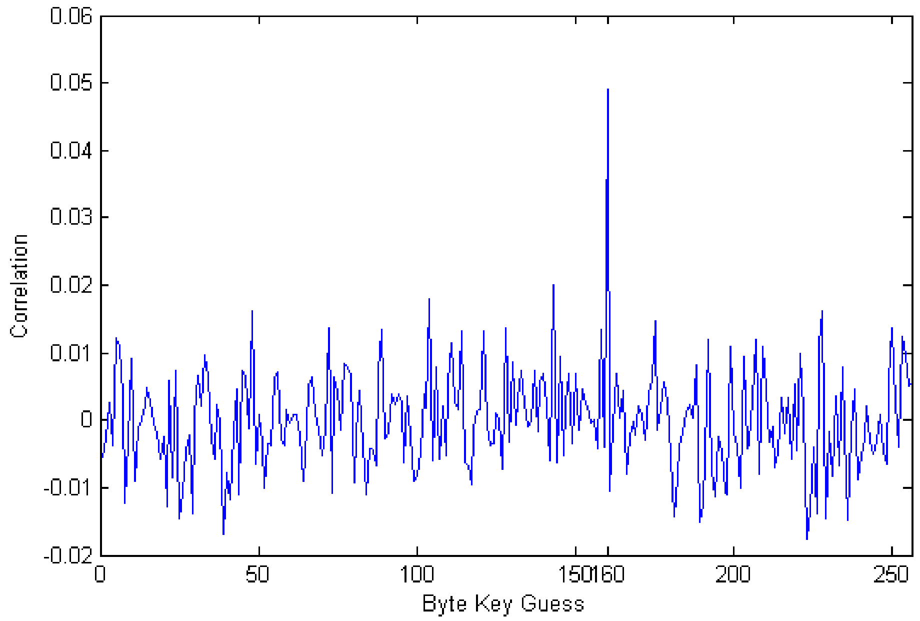


Figure 25: The correlation between the sensitive data and the power consumption for the 256 key guesses for a single byte.

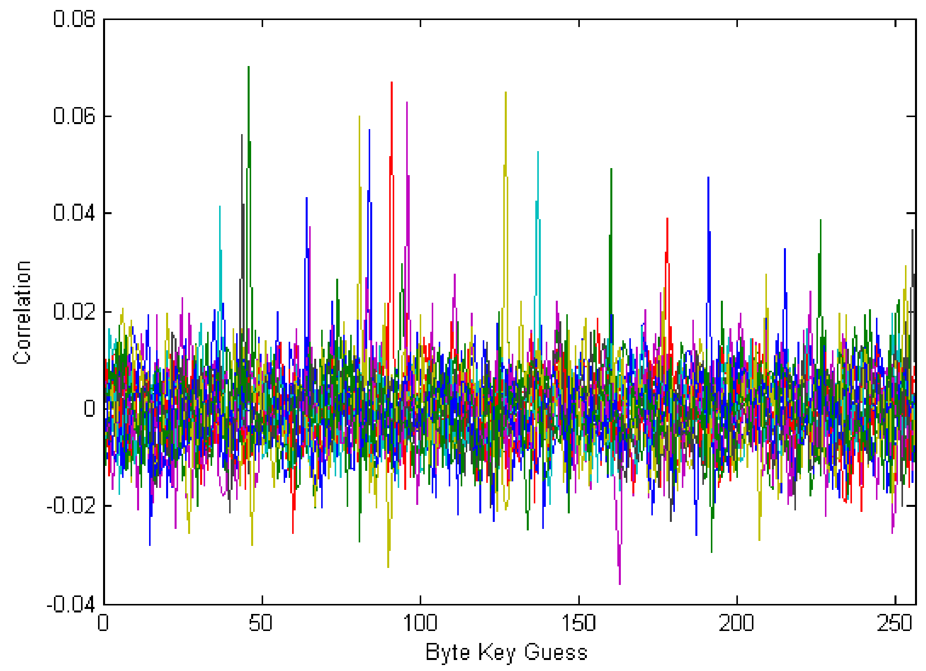
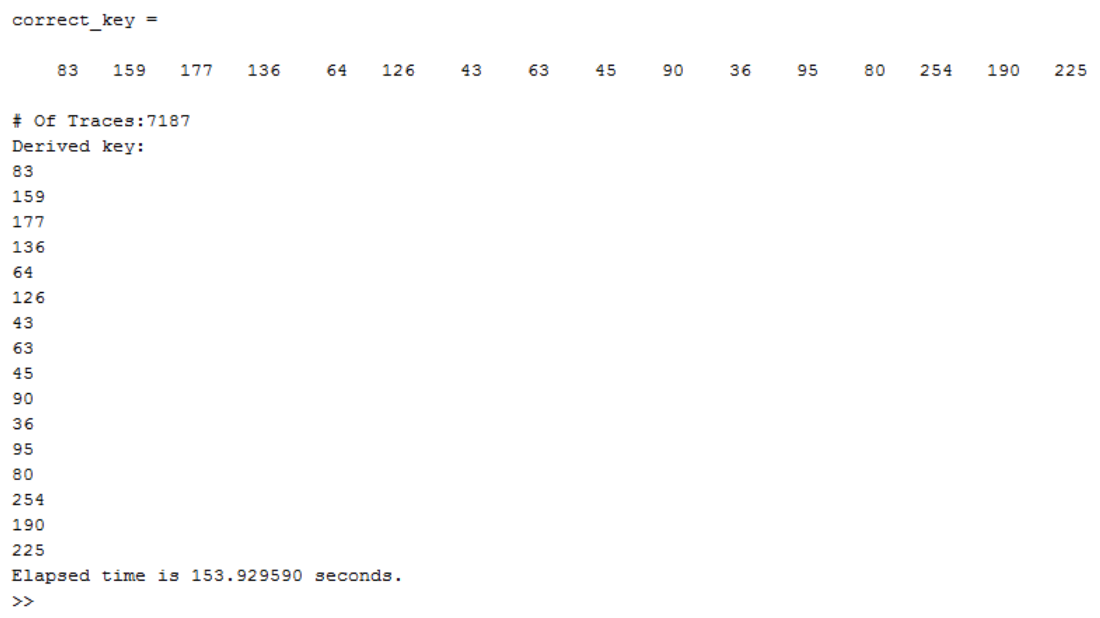


Figure 26: The correlation between the sensitive data and the power consumption for the 256 key guesses for all 16 bytes.

The same process is repeated 16 times, once for each byte in the message. The result is a table of 256 correlation values, one for each potential key, for each byte (16x256 total). The peak correlation for each byte is taken to be the correct key used in the final round of encryption. The final plot showing all the peaks in the correlation across all 16 bytes in the message is shown in Figure 26.

Overall, our attack program could successfully discover the key bits used by the AES encryption (in the final round) using only the ciphertext and the power trace information. The amount of computation was much greater than in the DES DPA attack, as 256 possible keys needed to be accounted for in 16 bytes as opposed to 64 possible keys for 8 S-boxes. However, the jump in computation number only caused an increase of a few minutes in the overall run time of the attack. To run the DPA attack through all 20000 traces that each AES-128 key produced took only 7-8 minutes (on a 3.0 Ghz computer). The correct key was aquired after an average of around 10000 traces, with 5700 being the minimum number of traces required, and 12000 being the maximum. An example output of our program on the first set of AES power traces is shown in Figure 27.



```
correct_key =

    83   159   177   136    64   126    43    63    45    90    36    95    80   254   190   225

# Of Traces:7187
Derived key:
83
159
177
136
64
126
43
63
45
90
36
95
80
254
190
225
Elapsed time is 153.929590 seconds.
>>
```

Figure 27: An example of the output of the Matlab AES DPA program.

The differential power analysis attack used to determine the first round subkey used in the DES algorithm was shown to also apply to the AES-128 algorithm with minor changes. The next logical step in the development of the attack would be to improve it to use as few power traces as possible. This is in line with the goal of the 2nd DPA Contest competition. While minor improvements were made by changing which points were used from the power traces, more substantial improvements should be made. Some potential areas for improvement include developing a more accurate power model for the AES system or looking at the power information in the frequency domain. Alternatively we could use a more powerful template attack.

7 Conclusions

In completing the project we were successfully able to recover the DES and AES-128 cryptosystem keys by mounting a DPA attack. The AES-128 implementation proved to be more difficult and the attacks shortcomings are explained further in this section. For the DES implementation the first round subkey was successfully recovered with 455 traces on average and an average execution time of less than 5 minutes. For AES-128, we were able to recover the subkey used with average of 10000 traces and an average execution time of 9 minutes.

Since our attack was designed with the DES implementation in mind, there were few traces required for recovery. This is accounted for by low resolution correlations which led to fairly accurate key guesses during the attacks runtime. However, running the attack on the AES cryptosystem resulted in very low correlations even after multiple executions. This is in part due to the attack not taking into consideration many of the more intricate parts of the AES implementation, such as the AddRoundKey and MixColumns stages. Due to some similarities between the two algorithms, we were still able to recover the final round subkey, but with a far greater amount of traces compared to the DES implementation.

While we only calculate the first and final round subkeys, to recover the rest of the fixed key for the DES implementation one would need to either rerun the attack or simply exhaustive search the keyspace once enough subkeys were found. However, when our attack considers an AES implementation it is noted that the full key may not be able to be recovered unless the key expansion can be reversed somehow. After a complete execution of the attack on a DES cryptosystem, we recovered 48 out of the 56 bits used for the fixed key. A simple exhaustive search algorithm would be able to recover the full key in a short amount of time.

From the success of our attack on the power traces from the DPA contest, it is clear that these types of side-channel attacks are very powerful when it comes to breaking a cryptosystem. What makes a DPA attack so powerful is that it can make plaintext or ciphertext only attacks, which greatly increase the versatility when attacking a target device. DPA-based attacks also perform much faster than other techniques such as exhaustive search. However, this type of side-channel attack requires access to the physical hardware in order to obtain the traces required for its execution.

8 Future Work

Given the success of the attack in DES and AES-128 implementations, there are a few directions in which this project could go towards future work. In current research, some of the only reliable models for power consumption to lead to a cryptosystems key have been the Hamming Weight and Hamming Distance models. One excellent way of improving our attack could come directly from a new power model or better utilization of the existing power models. This would allow higher correlations in the power traces and lead to more correct key guesses, resulting in the key space being narrowed down further to make exhaustive search techniques feasible. The way power traces are statistically correlated after the power model has been utilized is another source for additional work. More complex statistical methods could be explored in order to improve the correlations even further and allow for more information to be obtained from the power traces.

An alternative method to explore power analysis side-channel attacks for breaking cryptographic implementations would be template-based attacks. Template-based attacks are considered to be one of the most powerful types of side-channel attacks. This is because the statistical functions involved capture a large amount of data from the power traces and utilize probability density functions in order to obtain correct key guesses. For more information on template-based attacks, please see [5]. These types of attacks are now feasible with the introduction of a profiling trace table in the DPA Contest v2. This table contains approximately one million power traces, each with a random plaintext and key as well as the corresponding ciphertext. Along with the FPGA design for the AES-128 implementation, the profiling table makes it possible to analyze the system and build up templates which could be later used in a template-based attack.

One final possiblity for future MQP work in the area of differential power analysis attacks is to create an AES implementation on an FPGA, record power traces, and attempt to attack these traces. An example of AES trace aquisition on a Spartan 3E FPGA development board is given in [12]. It would give more insight into the physical aspects of the DPA side-channel attack.

References

[1] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In M. Wiener, editor, *Advances in Cryptology - CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388-397, Berlin, Hiedelberg, New York, 1999. Springer-Verlag.

[2] Yuichi Komano, Hideo Shimizu, and Shinichi Kawamura. Built-in Determined Sub-key Correlation Power Analysis. 2009.

[3] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security)*. Springer, 2007.

[4] NBS FIPS PUB 46, Data Encryption Standard. National Bureau of Standard, U.S. Department of Commerce, Jan. 1977.

[5] Suresh Chari, Josyula Rao, and Pankaj Rohatgi. Template Attacks. *Proceedings of CHES 2002, LNCS*, vol. 2523, pp. 13-28, 2002.

[6] Advanced Encryption Standard, Federal Information Processing Standards Publication 197. National Institute of Standards and Technology, November 2001.

[7] DES Challenge III, 1999. `http://www.rsa.com/rsalabs/node.asp?id=2108`.

[8] SASEBO-AES Specification Ver. 1.3 for JCMVP (Japanese), 2007.

[9] Sylvain Guilley. *Geometrical Counter-measures to Side-channel Attacks.* PhD thesis, TELECOM ParisTech, 2007.

[10] Sylvain Guilley. Dpa contest 2008/2009, 2008. http://dpacontest.org/.

[11] David Leech and Michael Chinworth. The Economic Impacts of NIST's Data Encryption Standard (DES) Program, 2001.

[12] Rajesh Velegalati and Panasayya Yalla. Differential Power Analysis Attack on FPGA Implementation of AES, 2008.