Combinational Power Analysis on Smart Cards

Balaji Megarajan
Department of Electrical & Computer Engineering,
Oregon State University, Corvallis, Oregon 97331 -USA.
E-mail: megaraba@engr.orst.edu

Abstract— **Every time information is transmitted from an ATM or any other machine to the host the secret key also accompanies the data. If we can clearly differentiate the transactions occurring each time, then by analyzing the data we can get to the secret key that is being used.The attack will have to be divided into first finding what type of transaction information is being transmitted then we can concentrate on deciphering the secret key.Various power analysis are being carried out extensively with the best available hardware and success is being achieved. An attempt will also be made to develop an algorithm which will keep the link busy so that the attacker won't be able to differentiate the actual data from the bits flowing.**

I. Introduction

The attacks on cryptographic systems often rely on the weak points of that system. There can be one or more weak points to a given system. The system often has a secret key and a public key. So the attacks are based on either of the one. And also the attacks are broken down into stages. The implementation of an algorithm in real life also draws out its flaws, which would have been difficult to zero in otherwise. Power analysis, fault insertion, and timing attacks are some of those. A few of these attacks are Differential Power Analysis, Simple Power Analysis and Inferential Power Analysis. Here power analysis implies that the modern devices, which are made of silicon for the transistors leak electromagnetic radiation, which can be analyzed to decipher the secret information. To find out the power consumed a resistance may be placed in the circuit and the value of the voltage across the resistance along with the value of the resistance can give us the power consumption.[1] Some of these attacks rely on the knowledge of the cipher text or the output data or the public key or the power consumed or the knowledge of the encryption algorithm not the particular implementation) by the system. Countermeasures have been implemented like smart cards carrying their own power source, different implementations of the security algorithm The goal of the paper is to present a new attack to the DES algorithm and also suggest a few measures to prevent the same. The paper first describes briefly the different methods that are popular nowadays in section 1 and in section 2 DPA in section 3 the SPA and ways to resist this attack and IPA in section 4 and my idea in section 5 and finally its implementation in section 6.

Author is a graduate student at the Department of Electrical & Computer Engineering, Oregon State University, Corvallis, Oregon 97331. E-mail: megaraba@engr.orst.edu

II. Differential Power Analysis

This attack relies on statistical analysis and error correction techniques to extract information related to secret keys. Some differentials on two sets of average consumption are computed and if an unusual phenomenon occurs then the key bits can be found out.[1] The unusual phenomenon can be because the data values are being manipulated. But here statistical functions can be used to target the algorithm. Naive implementations against can be attacked by high order DPA. Thus during the computation the attacker can get information other than the input and the output. The signals leaking from asymmetric operations are much stronger than those during symmetric operations as the multiplication operations are relatively complex operations.

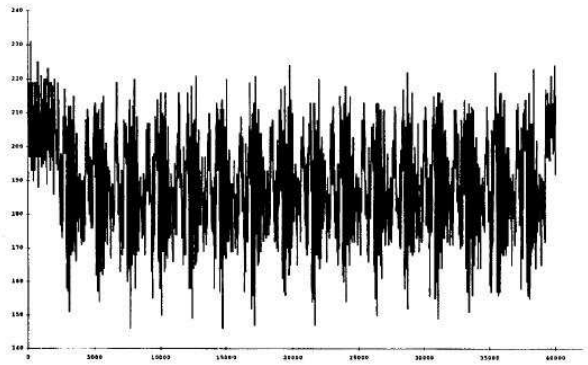

Figure 1: Power Consumption of Smart card

So implementing effective countermeasures against DPA can be challenging task. DPA spots correlation between the data being manipulated and the side channel implementation.[2]The specialty of the DPA attack is that it can find out the secret key of a public algorithm without knowing anything about that particular implementation. DPA can be used to break implementation of any symmetric or asymmetric algorithm. The basic principle of DPA is that the probability distribution of the power consumption given that a certain event occurs can be distinguished from the average probability distribution.[3]The attack is done as follows:

A. DPA Implementation for Data Encryption standard:

One of the most common attack by power analysis is the DPA attack against DES. The interesting aspect here is that this attack is based on the following assumption.

On average the consumption of the card computing with values x such as $x_i = 0^2$ can be distinguished from the consumption where $x_i = 1$.To perform DPA, different power consumption curves of the device must be first collected.[4]DES executes in 16 steps.

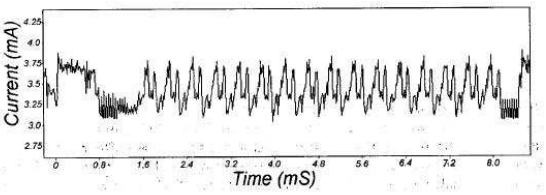

Figure 2: 16 rounds of DES

In each step transformation F is performed on 32 bits. F uses eight non-linear transformations (S boxes) from 6 to 4 bits.[4]
Step1: Measure input values and consumption values for 1000 DES computations. Also find the mean curves MC for these 1000 values.
Step2: Assume b is the value of the first output bit of the first S box. It depends on 6 secret key bits. A guess is made about these 6 bits and the expected output bits are calculated. Then the input data are separated into two categories, giving b = 0 and b = 1.
Step3: Then the mean MC' corresponding to the input for b= 0 is traced. If MC and MC' show a perceivable difference then the chosen values for the 6 bits are considered correct. Otherwise Step 2 is repeated for a new set of 6 bits.
Step4: With target b in the second S box the Steps 2 and 3 are repeated until the eighth S box.
Step5: The last 8 bits are found by exhaustive search.

The DPA attack does not require knowledge of position of instructions nor the position of each instruction. This attack is based on the assumption that there is an intermediate variable for which knowing a few key bits allows to decide whether two inputs will give the same output or not.

B. Preventing DPA:

The algorithm can be reinforced by introducing random shifts so that computed means do not match with the consumption of that instruction or by replacing some of the critical instructions by different assembler instructions whose consumption traces are difficult to analyze or giving an explicit way for computing the algorithm so that DPA is not effective on that implementation.[4] Also wide usage of modulus exponent modification processes in public key schemes can be used to prevent attackers from accumulating data across large number of operations. It should also be possible to remove any cover-ups that are just facial while testing so that the actual algorithm can be put to the test. Using constant execution path code or choosing operations that will leak lesser information and, of course physical shielding of the devices can also reduce the signal size thereby reducing the chances of intrusion. Finally the design assumptions have to be verified so that they are binding to the physical characteristics of the actual device.

III. SIMPLE POWER ANALYSIS

This attack is much more straightforward than DPA. What happens in this attack is that the attacker makes use of some quantity measurable at the outside of a cryptographic device to detect instruction being executed inside it.In this from the values of the consumption the particular instruction that is being executed is found and the values for the corresponding inputs and outputs are guessed to find the secret key.[4] SPA can yield information about the device's operation as well as the key. Typically the measurable quantity is the power consumption or radiation.[1] A trace is measurements made across the cryptographic consumption. As higher resolution trace for SPA are viewed small variations between rounds can be viewed. These small variations are caused by conditional jumps based on key bits and computational intermediates. These differences in the higher resolution traces are as a result of different power consumptions for different microprocessor instructions. Thus the SPA can reveal sequence of instructions executed and so it can be used to break cryptographic implementations in which execution path depends on the data being processed.

A. Preventing SPA:

Constant execution path code for micro code, which has large operand dependent power consumption features, can be vulnerable to SPA. The other method is to avoid procedures that use secret key or intermediates for conditional branching operations, which will mask SPA characteristics. Another effective way to protect against this type of attack is to program the cipher as a fixed sequence of instructions. This also means a constant execution time effectively sealing up the timing leak.[1] Also most hard wired implementations of symmetric cryptographic algorithms have small power consumption variations for which SPA can not yield keys.

IV. INFERENTIAL POWER ANALYSIS

This attack is also based on power analysis. This is done in two steps : profiling stage and key extraction stage. The profiling stage involves comparisons of repeated parts of a selected cryptographic operation. This comparison can be done on a single cryptographic module requiring many measured operations, which will result in a profile that can be used to extract keys from other modules. This attack also does not require the knowledge of the inputs or outputs. This attack enables to extract keys from other modules after a attack on just one of the modules. Lately cards are being provided with both the supply and the processing ability to process the secret key, so that the key is never exposed to the outside world. But if the cardholder wants to know the secret key then he can either subject the card to unusual conditions like out of range supply voltage, clock frequency, extreme temperatures to induce errors or physically dissect the card and reset the protection bits

or directly read the electrical signals that travel between the processor and the data bus. The cardholder can also measure the currents, voltages or the execution times and find a correlation to the actual secret key. The current drawn by the card gives lots of revealing information. If the smart card uses a hardware implementation for modular exponentiation algorithm, the current consumptions for the squaring and other operations are different and can be used to trace the actual implementation of the algorithm.
Implementation:
Here a large number of executions of the cryptographic algorithm are caused and with different plaintexts and the amount of power consumed for each step is recorded. The assumption is made that the algorithm does not have implementations in it to prevent power attacks and that the execution of the algorithm is deterministic function of the plaintext and key.[5]

A. Profiling stage:

This stage locates and identifies the key bits as they are used. Here the card is made to execute the algorithm a large number of times and it is assumed that the executions are all with the same key but different plaintexts. Then the traces are averaged to remove the effects of the varying data bits while keeping the effects of the constant key bits. This is done after all the traces are aligned. The result is a single average trace containing the power consumed during the execution. If there are n rounds and K_i is the sub key used in round i. The repeating structures are generated using the same source code being executed over and over. The average trace is chopped in to rounds to obtain traces R1,R2..Rn. then these are averaged to obtain a single super average. Next the differences are taken between each rounds R1,R2..Rn with the super average R to obtain the round i difference trace. Then these difference traces are squared and averaged to obtain the peaks that reveal the key bit location. As we know the algorithm we know the number of key bits in use.

B. Key bit identification:

In DES the key identification is difficult as the sub key consists of 48 bits used as inputs to 8 S boxes. The operations of the S boxes do not depend on each other in a round so all 8! orderings of the S boxes are possible. Also the 6 key bits loaded in an S box do not have to be in order. After the key identification the next step is key scheduling, which specifies the pattern of in which individual key bits move from one S box position to another in consecutive rounds. Then finally there is the table of locations and also the corresponding key bit identity.

C. Fast Key extraction:

The bits obtained from profiling depend only on the software implementation and not on the key that was used.[5] So the table will be valid for other implementations of the same software running on same hardware and so the key bits can be easily found for any such instance. To do the key extraction from a new instance of the same implementation a single power trace is needed and it is chopped into rounds and then measuring the power consumed at the locations specified by the table.

D. Advantages and Defenses:

IPA does not need the knowledge of either the plain text or the cipher text. It can look at any part of the algorithm, which can be useful where some implementations are done at some part of the algorithm. Also IPA has the ability of fast key extraction after a lengthy profiling stage, thereby making it feasible in the long term.[5] The IPA can be avoided by not handling the key bits one at a time. Also the execution of the codes and the representation of the data have to be randomized and by limiting the number of times the key can be used without confirmation of legitimacy. The addition of noise prevents key extraction from a captured trace and limit on the key probes will also discourage attacks.

V. The Big Idea

This attack is based on the comparison of the repeated parts of an algorithm. When the smart card is swiped through the card reader it gives the secret key stored in the card and the reader identification number to the chip in the smart card. This attack is being tested on the DES algorithm because of its wide usage. In each of the 16 rounds of the DES it uses 8 S boxes, which take as input 6 bits. This attack needs the knowledge of the input unlike DPA. The figure shows (like DPA pg 3) the differences in the power consumption for different instruction execution. Besides the instruction set being used by that processor in the smart card also largely influences the power consumption curves. The curves are finally aligned and averaged to get a final waveform, which will clearly show the sixteen peaks depicting the 16 cycles in the DES execution.

VI. The Combinational attacks

First I will put down the assumptions for this deadly attack. It assumes that the attacker knows the input values and that the smart card uses the DES cryptographic algorithm. The knowledge of the secret key would far shorten the attackers job. Besides the algorithm has to be implemented in the normal fashion without any application specific modifications in it. If any modifications were made to the algorithm then another Combinational attack can work on it, but the limit of space here forces me to restrict my explanation. The attack needs that the user repeatedly enters the input data. This can be achieved by introducing a thin, clear and rigid plastic sleeve on the card reader. What this simple idea does is that the card reader can't read the card and would repeatedly ask the user to enter the pin number. This way the smart card also has to execute its cryptographic algorithm repeatedly. While this is taking place we need to take continuous waveforms of the power consumption by the hardware. Here I say power because for the same resistance if the voltage change then there has to be a corresponding change to the current through the

resistor and this changing current across the resistor cause power dissipation.
After achieving all the waveforms they are aligned and then averaged together to remove the effect of any noise and to also remove the effects of varying data bits. The waveforms are each squared and then added to each other. What this does is it takes care of the varying data bits and only allows the effect of the constant bit to remain. The number of waveforms that have been added then divides the final waveform and a square root of this final waveform is taken. This final waveform very clearly shows the 16 peaks depicting the sixteen rounds of the DES algorithm. These highs may represent either the load or store operation or a branch operation or a jump execution or a modular multiplication.
The next step is to locate the key bits in the waveforms and their values. The DES algorithm uses 8 S boxes and 6 key bits for the calculation of the secret key bits. We take a single bit of the waveform and then try and conjecture on the bits involved. This way we have 64 waveforms for the 26 bits of the secret key. Each of the 64 waves is squared and then all of these are added to each other. This final waveform is then averaged and then the square root value of this composite waveform is calculated. This procedure is repeated for all the 8 S boxes. What is finally achieved is 48 bits of the DES secret key. The knowledge of the instruction set of the smart chip will far simplify the decoding of the waveforms as we will clearly know the particular instruction being executed and thereby we can decipher the steps in the cryptographic algorithm. This is because as the card is inserted into he reader the reader is powered up and sets up the card to receive software commands.

A. Securing against Combinational algorithm:

There are very few methods against this attack. Among them I can state is minimizing the instruction set, as this will give similar waveform for most of the operations, which will be difficult to decipher. Another way is to use smart card with detached power supply.

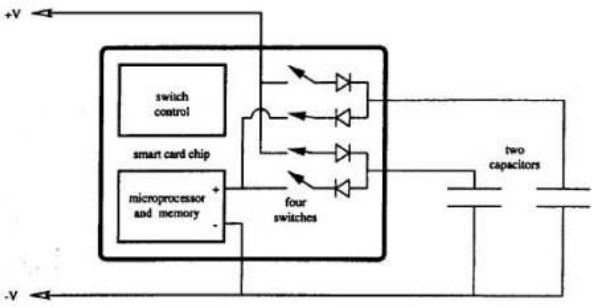

Figure 3: Smart card with detached power supply

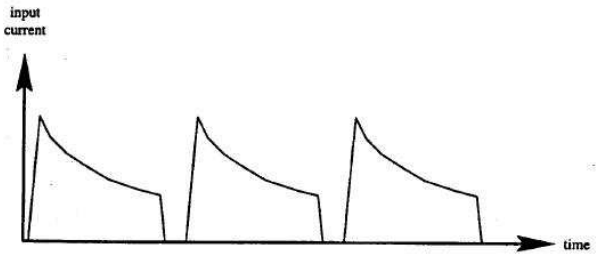

Figure 4: Power Consumption

[6]Introducing variations in the execution of the algorithm. Also introducing random shifts in the frequency with which key is used will cause unknown periods in the output waveforms. The simple way of avoiding the sleeve in the card reader is to run your finger along the card slot before you put your card in. The sleeve has a couple of tiny prongs that the attacker needs to get the sleeve out of the slot, and you will be able to feel them. Another method of fooling the attacker will be to overwhelm the reader with dummy data so that the attacker will not be able to differentiate between the actual execution of the algorithm and this dummy data.

VII. Conclusions

I have concluded that the Combinational attack is much faster

	Time To Crack	Time To Crack
	56-bit DES	1024-bit RSA
Combinational Attack	4.18 μs	0.128ms
DPA	5.57 μs	0.14 ms
IPA	6.03 μs	0.153ms

Table 1: Smart Card Market by Industry(Millions of Units).

than the other algorithms that have been presented here and the industry should also take into consideration the Combinational attack procedure before devising any new counter attack methods.

References

[1] Joshua Jaffe Paul C. Kocher and Benjamin Jun, "Differential power analysis," in *Advances in Cryptology - CRYPTO'99*, Micheal Wiener, Ed., 1999, pp. 387–397.
[2] Nora Dabbous Christophe Clavier, Jean-Sbastien Coron, "Differential power analysis in the presence of hardware countermeasures," in *Cryptographic Hardware and Embedded Systems — CHES'00*, Ç. K. Koç and C. Paar, Ed., 2000, Lecture Notes in Computer Science, No. 1965, pp. 252–263.
[3] Michael Peters Joan Daemen and Gilles Van Assche, "Bitslice ciphers and power analysis attacks," in *Fast Software Encryption Workshop 2000*, Bruce Schneier, Ed., 2000, pp. 134–149.
[4] Louis Goubin and Jacques Patarin, "Des and differential power analysis," in *Cryptographic Hardware and Embedded Systems - CHES 1999*, Ç. K. Koç and C. Paar, Eds., 1999, Lecture Notes in Computer Science, No. 1717, pp. 158–172.
[5] Paul N. Fahn and Peter K. Pearson, "Ipa: A new class of power attacks," in *Cryptographic Hardware and Embedded Systems - CHES 1999*, Ç. K. Koç and C. Paar, Eds., 1999, Lecture Notes in Computer Science No. 1717, pp. 173–186.
[6] Adi Shamir, "Protecting smart cards from passive power analysis with detached power supplies," in *Cryptographic Hardware and Embedded Systems*, "Ç. K. Koç and C. Paar, Eds., Lecture Notes in Computer Science, No. 1965, pp. 71–77.