MASTER RESEARCH INTERNSHIP

BIBLIOGRAPHIC REPORT

SCARE for Hardware SPN

Domain: Cryptography and Security

Author:
Paul SAUROU

Supervisor:
Benoît GÉRARD
Margaux DUGARDIN
SDCyb1/EPOC/XCS

Abstract: write your abstract here

Contents

Introduction

Following Kerckhoffs' principle, the security of a cryptographic device shall not rely on the secrecy of its mechanisms [Kerckhoffs 1883]. However in some specific contexts, having a secret implementation can add a layer of security, by increasing the practical difficulty of the attack.

1 State of the art of Side Channel Attacks on SPNs

In this section, we formally define substitution–permutation network ciphers. Then, we introduce some of the most classical and standardized implementations. Finally, we present several Side Channel Analysis classic attacks on these implementations.

1.1 SPNs, Feistel schemes, DES and AES

Symmetric cryptography refers to the protection of a communication between a sender and a receiver who share the same secret key. An algorithm that derives from symmetric cryptography is called a symmetric algorithm. Symmetric algorithms mostly relies on either substitution ciphers, stream ciphers or block ciphers.

Substitution ciphers replace the units of plaintext by ciphertext in a defined manner, with the help of a key. Those are well known in history but can nowadays be easily decrypted using a frequency table or a similar mechanism.

Stream ciphers encrypt the characters (in the form of bytes) of a message one at a time.

Block ciphers encrypt fixed-length blocks of plaintext.

This internship will focus on block ciphers, and especially on substitution–permutation networks.

1.1.1 Block ciphers

A block cipher consists of both an encryption function E and a decryption function $D = E^{-1}$. They both accept two inputs: a block of text of fixed size n and a key of size k. They output a block of size n.

The difference between different block ciphers is the definition of the function E (and D).

Most block ciphers are iterated, which means that E consists in the repetition of a round function R taking a different round key K_i each round i and the output of the previous round as the new input. If M_0 is the plaintext and there are r rounds, then $\forall i \in [1, r], M_i = R(M_{i-1}, K_i)$ and $M_r = E(M_0)$ is the ciphertext. $R(M, K_i)$ is generally written $R_{K_i}(M)$.

Choosing the number of rounds of an iterated block cipher scheme is a compromise between computing efficiency, cryptographic security and side channels leakages. Too few rounds allow for cryptographic attacks and too many rounds make the cipher inefficient and more difficult to implement securely.

We will now define two types of iterated block ciphers: Feistel networks and substitution–permutation networks.

1.1.2 Feistel ciphers

In a Feistel scheme, the plaintext block is divided in two parts (p, q) of equal length $n/2$. With the previous notations, the round function is $R : (p, q) \mapsto (q, p \oplus f(q, K_i))$ where f is called the Feistel function. Both the security and the performance of the Feistel network depend on the Feistel function f. A specificity of Feistel schemes is that f does not need to be invertible. The decryption is almost identical to the encryption, the only difference being the reversal of the round keys order.

1.1.3 DES

The Data Encryption Standard (DES) is a Feistel network scheme that has been highly influential in the advancement of cryptography. It is unsecure due to its too short 56-bit key size which make it vulnerable to brute-force attacks. However its design is interesting. It consists in a 16 round Feistel network and operates on blocks of 64 bits. The Feistel function of DES is shown on

First the 32-bit half-block is expanded to 48 bits using the E expansion permutation function. Then the 48-bit block is xored with the round subkey and goes through a substitution layer. It is divised into eight 6-bit words which are replaced by 4-bit words using a lookup table. Finally the 32 outputs are mixed according to a fixed permutation, the P-box.

1.1.4 SPN ciphers

In substitution–permutation networks (SPN), the round function consists of a substition stage followed by a permutation stage. It splits the plaintext M of size n into l blocks $m_1, m_2, ..., m_l$ of equal length. The substitution stage is non-linear and mixes the round key with the plaintext. It uses a set of substitution-boxes (S-boxes) that do one-to-one substitution to be reversible. The permutation stage rearranges all the bits of the output of the S-boxes each round, as shown on Figure 1.

More formally and with the previous notations, the round function is $R_{K_i} = \lambda \circ \gamma \circ \sigma_{K_i}$ with

- $\sigma_K : M \mapsto M \oplus K$ the key addition layer,

- $\gamma : M = (m_1, m_2, ..., m_l) \mapsto S_1(m_1), S_2(m_2), ..., S_l(m_l)$ the susbstitution layer, where $(S_i)_{1 \leq i \leq l}$ are non-linear substitution functions.

- $\lambda : M \mapsto A \times M$ the linear layer, where $\times$ is the matrix product.

In some SPN implementations, the final round sometimes skips the linear layer and an additional key addition is often performed after the final substitution layer.

The S-boxes $(S_i)_{1 \leq i \leq l}$ can be chosen identical. All the S-boxes do not have the same cryptographic properties. One is considered secure if changing one input bit changes almost half of the output bits on average.
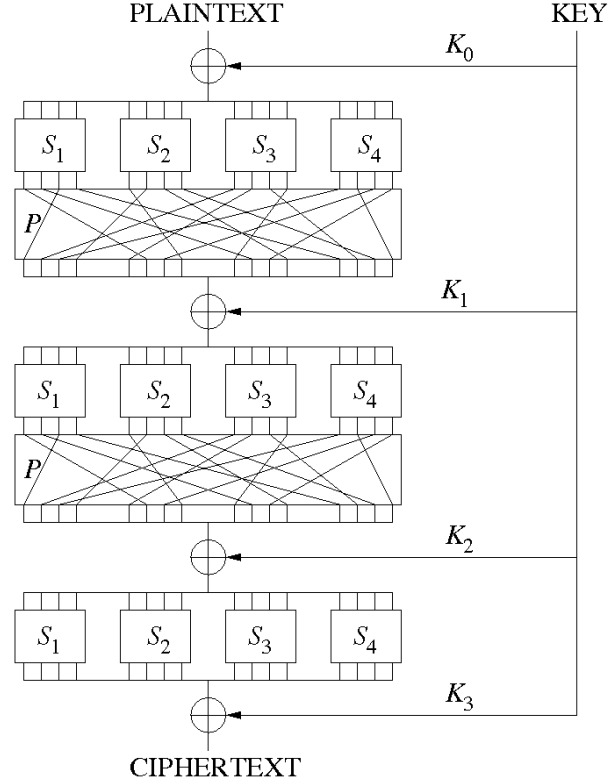
2

Figure 1: A sketch of a substitution–permutation network with 3 rounds, encrypting a plaintext block of 16 bits into a ciphertext block of 16 bits. The S-boxes are the Si, the P-boxes are the same P, and the round keys are the Ki. Image and caption from wikipedia.org

1.1.5 AES

The Advanced Encryption Standard (AES) is the replacement of DES chosen by the NIST in 2001. The algorithm, also called Rijndael, is a SPN. It exists in three variants:

- 10 rounds and 128-bit keys

- 12 rounds and 192-bit keys

- 14 rounds and 256-bit keys

The 128-bit block plaintext is arranged in a $4{\times}4$ byte matrix $\begin{pmatrix} m0 & m4 & m8 & m12 \\ m1 & m5 & m9 & m13 \\ m2 & m6 & m10 & m14 \\ m3 & m7 & m11 & m15 \end{pmatrix}$ The round keys are derived using the AES key schedule. All the 16 S-boxes are the same and it is chosen to have no fixed points. The linear layer is divided into a ShiftRows and a MixColumns operation. The ShiftRows step cyclically shifts the bytes in each row by a certain offset (from 0 to 3):

$$\begin{pmatrix} m0 & m4 & m8 & m12 \\ m1 & m5 & m9 & m13 \\ m2 & m6 & m10 & m14 \\ m3 & m7 & m11 & m15 \end{pmatrix} \rightarrow \begin{pmatrix} m0 & m4 & m8 & m12 \\ m5 & m9 & m13 & m1 \\ m10 & m14 & m2 & m6 \\ m15 & m3 & m7 & m11 \end{pmatrix}$$

The MixColumns step then multiplies each column in place by the matrix $\begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix}$ so each byte of the column influences the output of the four other bytes.

The details of the AES standard are defined here [Standards and Technology 2001].

3

1.2 Side Channel Analysis classic attacks

All the implementations of cryptographic algorithms in real world devices provide more information to an attacker than just the plaintext and the ciphertext. These side-channel leakages can consist in the timing of operations, power consumption, electromagnetic emanations, etc. They are not taken into account by the mathematical models and very little side-channel information is enough to break many common ciphers.

Non-invasive attacks and corresponding countermeasures have been studied extensively over the past decades.

In the following, side channel attacks consist in associating the power consumption of a cryptographic module with the data processed by the module at the same time. Indeed, the physical properties of semiconductors link the data processed by a chip to its power consumption.

To model the link between data transitions and power consumptions, the Hamming weight and Hamming distance models are often used.

The **Hamming Distance** of $a \in \mathbb{F}_2^n$ and $b \in \mathbb{F}_2^n$ is $H(a \bigoplus b) = \sum_{i=0}^{n-1} \mathbb{1}_{a_i \neq b_i}$ where a_i (resp. b_i) is the ith bit of a (resp. b).

It counts the number of transitions when $a \rightarrow b$. For consecutive values in the same register, this metric correlates the Hamming distance with the power consumption. However the attacker needs to obtain consecutives data values to use this model. A simpler solution is to consider that $a = 0$. The model then becomes the following.

The **Hamming Weight** of $a \in \mathbb{F}_2^n$ is $H(a) = \sum_{i=0}^{n-1} \mathbb{1}_{a_i=1}$ where a_i is the ith bit of a.

It counts the number of bits set to 1 when storing the variable a. It is the most widely used leakage model.

Single power analysis (SPA) involves directly interpreting power consumption measurements collected during cryptographic operations. SPA can yield information about a device's operation as well as key material.

Differential Power Analysis (DPA) involves statistical tools to study multiple power consumption leakages from a cryptographic module. It aims to remove noise by using signal processing and error corrections techniques.

Template attacks (or profiling attacks) creates a profile of the attacked device with many generated traces and uses it to later find the secret of the victim with a very small number of traces.

First described in [Chari, Rao, and Rohatgi 2003], it uses a copy of the protected device to record a large number of power traces using many different plaintexts and keys. The noise is modeled using multivariate gaussian distributions, one for each possible subkey. If a subkey is a byte, then the attack would need 256 different templates. The samples are used to identify the n points where the differences between averaged signals are the largest. These are points of interest and the model will only focus on their values. The "templates" are then created for each subkey on these points of interest by combining the mean signal M_i and the noise covariance matrix Σ_{N_i}. The attack then consists in computing the probability that the trace from the target device originates from each one of the template. Selecting the highest probability is optimal. Depending on the training set, a few or even one leakage can be enough to identify a subkey. The original paper also calculates the error probability of their test and explain how to improve the attack by pruning the least probable hypotheses.

Mutual Information Attacks (MIA) uses entropy and mutual information between the leakage and the predicted data. It does not require any prealable knowledge of the studied device and

can be applied in any context.

The reference paper [Prouff and Rivain 2009] presents a theoretical analysis of the assumptions made in side channel attacks. It studies MIA in the Gaussian leakage model and in the case of masked implementations. By making less assumptions about the model, MIA has worse performance than other techniques using correlations between traces on devices following the Hamming weight model. However it could retrieve the key in a context where the trace is not linearly linked to the Hamming weight of the manipulated data.

Over the last years, **Deep Learning based Attacks)** (DL) have reached the state of the art of side channel attacks. Deep Neural Networks (DNNs) are able to efficiently break implementations of AES that resisted to other types of SCA [Maghrebi, Portigliatti, and Prouff 2016].

These are the main side channel analysis techniques. They aim to retrieve the key, or some other secret component, involved in a known algorithm. Template attacks even use a replica of the attacked device to "prepare" the attacks. Nonetheless it is also important to verify the security of cryptographic implementations against weaker opponents that do not know the exact implementation or its details. For cryptographic systems that choose to use proprietary algorithms or a specific set of parameters for a known standard, the architecture itself is a secret component that can be determined through SCA. The following section is about Side Channel Analysis Reverse Engineering (SCARE) and aims at assessing the gains or losses in security of choosing secret parameters for symmetric cryptography algorithms.

2   SCARE attacks

2.1   SCARE attacks on non AES ciphers

[Novak 2003] presents a side-channel attack on substitution blocks with a demonstration on a SIM card using COMP-128 cipher.

[Daudigny et al. 2005] presents a SCARE attack on DES and propose new methods to exploit the power measurement information.

[Guilley et al. 2010] presents two SCARE attacks on the parameters of a LFSR and DES.

2.2   SCARE attacks on AES-like ciphers

[Tiessen et al. 2015] presents an integral cryptanalysis of an AES with a secret S-box and less rounds. It is not a SCA but is still closely related to our subject.

[Rivain and Roche 2013] presents a generic SCARE attack against a wide class of SPN block ciphers.

FIRE (injection fault attempts) and SCARE attacks to recover the full set of secret parameters of an AES-like software implementation, even with masking and shuffling [Clavier et al. 2015].

3   Application to hardware implementations

3.1   Differences between software and hardware cryptographic implementations

- The sequential execution becomes parallel

- There is even more noise

- SCA leaks cannot be determined as precisely

- It takes more time to implement, develop and attack

3.2 Attacks on hardware design

[Réal et al. 2008] presents a SCARE attack on a general Feistel scheme with an hardware design. SAKURA board reference

[Guilley et al. 2010] presents two SCARE attacks on the parameters of a LFSR and DES, implemented on FPGAs.

SCA attacks on FPGAs ([Peeters et al. 2005] or more relevantly [Standaert, Ors, and Preneel 2004] or something more recent)

4 Conclusion

References

References

Chari, Suresh, Josyula R. Rao, and Pankaj Rohatgi (2003). "Template Attacks". en. In: *Cryptographic Hardware and Embedded Systems - CHES 2002*. Ed. by Burton S. Kaliski, çetin K. Koç, and Christof Paar. Vol. 2523. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 13–28. ISBN: 978-3-540-00409-7. DOI: 10.1007/3-540-36400-5_3. URL: http://link.springer.com/10.1007/3-540-36400-5_3.

Clavier, Christophe et al. (Mar. 2015). "Complete reverse-engineering of AES-like block ciphers by SCARE and FIRE attacks". en. In: *Cryptography and Communications* 7.1, pp. 121–162. ISSN: 1936-2447, 1936-2455. DOI: 10.1007/s12095-014-0112-7.

Daudigny, Rémy et al. (2005). "SCARE of the DES: (Side Channel Analysis for Reverse Engineering of the Data Encryption Standard)". en. In: *Applied Cryptography and Network Security*. Ed. by John Ioannidis, Angelos Keromytis, and Moti Yung. Vol. 3531. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 393–406. ISBN: 978-3-540-26223-7. DOI: 10.1007/11496137_27. URL: http://link.springer.com/10.1007/11496137_27.

Guilley, Sylvain et al. (Aug. 2010). "Defeating Any Secret Cryptography with SCARE Attacks". In: vol. 6212, pp. 273–293. ISBN: 978-3-642-14711-1. DOI: 10.1007/978-3-642-14712-8_17.

Kerckhoffs, Auguste (Jan. 1883). "La cryptographie militaire". In: *Journal des sciences militaires*, pp. 5–38.

Maghrebi, Houssem, Thibault Portigliatti, and Emmanuel Prouff (2016). "Breaking Cryptographic Implementations Using Deep Learning Techniques". en. In: *Security, Privacy, and Applied Cryptography Engineering*. Ed. by Claude Carlet, M. Anwar Hasan, and Vishal Saraswat. Vol. 10076. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 3–26. ISBN: 978-3-319-49444-9. DOI: 10.1007/978-3-319-49445-6_1. URL: http://link.springer.com/10.1007/978-3-319-49445-6_1.

Novak, Roman (2003). "Side-Channel Attack on Substitution Blocks". en. In: *Applied Cryptography and Network Security*. Ed. by Jianying Zhou, Moti Yung, and Yongfei Han. Vol. 2846. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 307–318. ISBN: 978-3-540-20208-0. DOI: `10.1007/978-3-540-45203-4_24`. URL: `http://link.springer.com/10.1007/978-3-540-45203-4_24`.

Peeters, Eric et al. (2005). "Improved Higher-Order Side-Channel Attacks with FPGA Experiments". en. In: *Cryptographic Hardware and Embedded Systems – CHES 2005*. Ed. by Josyula R. Rao and Berk Sunar. Vol. 3659. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 309–323. ISBN: 978-3-540-28474-1. DOI: `10.1007/11545262_23`. URL: `http://link.springer.com/10.1007/11545262_23`.

Prouff, Emmanuel and Matthieu Rivain (2009). "Theoretical and Practical Aspects of Mutual Information Based Side Channel Analysis". en. In: *Applied Cryptography and Network Security*. Ed. by Michel Abdalla et al. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 499–518. ISBN: 978-3-642-01957-9. DOI: `10.1007/978-3-642-01957-9_31`.

Réal, Denis et al. (2008). "SCARE of an Unknown Hardware Feistel Implementation". en. In: *Smart Card Research and Advanced Applications*. Ed. by Gilles Grimaud and François-Xavier Standaert. Vol. 5189. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 218–227. ISBN: 978-3-540-85892-8. DOI: `10.1007/978-3-540-85893-5_16`. URL: `http://link.springer.com/10.1007/978-3-540-85893-5_16`.

Rivain, Matthieu and Thomas Roche (2013). "SCARE of Secret Ciphers with SPN Structures". en. In: *Advances in Cryptology - ASIACRYPT 2013*. Ed. by Kazue Sako and Palash Sarkar. Vol. 8269. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 526–544. ISBN: 978-3-642-42032-0. DOI: `10.1007/978-3-642-42033-7_27`. URL: `http://link.springer.com/10.1007/978-3-642-42033-7_27`.

Standaert, François-Xavier, Berna Ors, and Bart Preneel (Jan. 2004). "Power Analysis of an FPGA: Implementation of Rijndael: Is Pipelining a DPA Countermeasure?" In: vol. 3156, pp. 30–44. DOI: `10.1007/b99451`.

Standards, National Institute of and Technology (2001). "Advanced Encryption Standard". In: *NIST FIPS PUB 197*.

Tiessen, Tyge et al. (2015). "Security of the AES with a Secret S-Box". en. In: *Fast Software Encryption*. Ed. by Gregor Leander. Vol. 9054. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 175–189. ISBN: 978-3-662-48115-8. DOI: `10.1007/978-3-662-48116-5_9`. URL: `http://link.springer.com/10.1007/978-3-662-48116-5_9`.