The Applications of Linear Transformation Methods in the Domain of Side-Channel Analysis

Author: Rauf MAHMUDLU
Supervisors: Dr. Lejla BATINA\(^1\)
Dr. Valentina BANCIU\(^2\)
Dr. Ileana BUHAN\(^2\)

A thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in the
Digital Security Group
Institute for Computing and Information Sciences

August 24, 2016

\(^1\)Digital Security Group, Radboud University, The Netherlands
\(^2\)Riscure BV, Delftechpark 49, 2628 XJ Delft, The Netherlands
Abstract

Faculty of Science
Institute for Computing and Information Sciences

Master of Science

The Applications of Linear Transformation Methods in the Domain of Side-Channel Analysis

by Rauf MAHMUDLU

Side-channel attacks put the security of the implementations of cryptographic algorithms under threat. Secret information can be recovered by analysing the physical measurements acquired during the computations and using key recovery distinguishing functions to guess the best candidate. Several generic and model based distinguishers have been proposed in the literature. In this work we investigate the application of dimensionality reduction methods used as means of Point of Interest (POI) selection for building side-channel templates. We compare the Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) based methods and conclude that the latter is the more preferable option especially in the presence of high amount of noise.

Additionally, we describe two contributions that lead to better performance of side-channel key recovery attacks in challenging scenarios. First, we describe how to transform the physical leakage traces into a new space where the noise reduction is near-optimal. Second, we propose a new generic distinguisher that is based upon minimal assumptions. It approaches a key distinguishing task as a problem of classification and ranks the key candidates according to the separation amongst the leakage traces. We also provide experiments and compare their results to those of the Correlation Power Analysis (CPA). Our results show that the proposed method can indeed reach better success rates even in the presence of significant amount of noise.
Acknowledgements

I would like to take the opportunity to thank my supervisors Ileana Buhan, Valentina Banciu and Lejla Batina for they have addressed the questions that I have had about the technical and theoretical challenges, and offered guidance during the times when the project was not advancing as intended. Their valuable comments have inspired me to explore new ideas and approach problems from multiple angles. I thank the whole Riscure B.V. for welcoming me within their family and providing an academically and technically advanced environment for research. I would like to extend my thanks to the Digital Security group for sharing their knowledge and offering their support.
Contents

Abstract iii

List of Figures ix

List of Tables xi

1 Introduction 1
   1.1 Research Questions ........................................ 1
   1.2 Structure ................................................... 2

2 Preliminaries 3
   2.1 Information Theory ........................................... 3
       2.1.1 Random Variables ..................................... 3
       2.1.2 Joint and Conditional Probability .................... 4
       2.1.3 Shannon Entropy ........................................ 4
       2.1.4 Joint and Conditional Entropy ......................... 4
       2.1.5 Mutual Information .................................... 5
   2.2 Dimensionality reduction ................................. 6
       2.2.1 Principal Component Analysis ......................... 6
       2.2.2 Linear Discriminant Analysis ....................... 7
   2.3 Side-Channel Analysis .................................... 8
       2.3.1 Correlation Power Analysis ............................. 10
       2.3.2 Template Analysis .................................... 11
   2.4 Cryptographic Algorithms ................................ 11
       2.4.1 Data Encryption Standard .............................. 11
       2.4.2 Advanced Encryption Standard ....................... 12

3 Template Attacks in Principal Subspaces 15
   3.1 Experimental Setup ......................................... 15
   3.2 Template Building ......................................... 16
       3.2.1 PCA-Based POI Selection ............................... 17
       3.2.2 LDA-Based POI Selection ............................ 18
   3.3 Template Matching .......................................... 19
   3.4 Results ..................................................... 20

4 (De-)classification of Keys 23
   4.1 Attack description .......................................... 24
       4.1.1 The Leakage Transformation Step .................... 24
       4.1.2 The Distinguishing Step ............................... 25
   4.2 Caveats ..................................................... 28
   4.3 Experimental Validation .................................. 28
       4.3.1 HW Leakage Model .................................... 28
       4.3.2 Identity Leakage Model ............................... 29
       AES Encryption ............................................. 30
DES Encryption ........................................... 30
4.3.3 Computational Complexity ..................... 32

5 Final Words ........................................... 35
  5.1 Conclusion ........................................ 35
  5.2 Future Research .................................. 36

A Supplementary Material for Cryptographic Algorithms 41
  A.1 Supplementary Material for AES ................. 41
  A.2 Supplementary Material for DES ................. 42
    A.2.1 Permutation Tables .......................... 42
    A.2.2 S-Box Functions ............................. 43
List of Figures

2.1 Binary entropy as a function of the probability $p$ .................... 5
2.2 The relation between the mutual information and the conditional entropies ......................................................... 6
2.3 Schematic illustration of a side-channel key recovery ................. 10

3.1 Acquisition setup .......................................................... 16
3.2 Mean traces associated with each of the key classes for different trace sets. ......................................................... 17
3.3 Visualisation of the leakage traces after their projection into the sub-space determined by PCA. ............................. 18
3.4 Visualisation of the leakage traces after their projection into the sub-space determined by LDA. ................................. 19
3.5 Empirical and binomial probability distributions of the key classes for AES and DES implementations. ................. 20
3.6 Success rates for the POI selected with PCA transformation ..... 21
3.7 Success rates for the POI selected with LDA transformation .... 22
3.8 Success rates for the POI selected as the samples with the highest correlation coefficient ................................. 22

4.1 Schematic illustration of the proposed attack ......................... 25
4.2 The known key correlation of the trace samples with the HW of the intermediate values. ................................. 25
4.3 The first projection direction constructed for the LDA transformation where the traces are classified based on the value of the plaintext. 26
4.4 Visualization of the class separation under different key candidates 27
4.5 GSR when the hypothetical power consumption is predicted according to the HW Model .................. 29
4.6 The SNR before and after the LDA transformation ............... 30
4.7 GSR for AES implementation when the hypothetical power consumption is predicted according to the Identity Model .......... 31
4.8 GSR for DES implementation when the hypothetical power consumption is predicted according to the Identity Model .......... 32
List of Tables

4.1 The analysis of the Mutual Information (MI) between the key and the classification for AES and DES S-box outputs .......................... 31

A.1 AES S-box function ................................................. 41
A.2 Initial Permutation .................................................. 42
A.3 Expansion Permutation ............................................. 42
A.4 P-box ................................................................. 42
A.5 Final Permutation .................................................... 42
A.6 DES S-Box functions ................................................ 43
Acronyms

CPA  Correlation Power Analysis. iii, 1, 2, 10, 24, 26, 28, 29, 31, 32, 36

DPA  Differential Power Analysis. 1, 23

EM   electro-magnetic. 9, 15, 16, 18–20, 28

GSR  Global Success Rates. vii, 15, 20, 21, 24, 28–31, 36

HW   Hamming weight. vii, 16, 20, 23, 25, 27–31

LDA  Linear Discriminant Analysis. iii, vii, 1, 2, 6, 7, 15, 19–24, 26, 28, 29, 35, 36

MI   Mutual Information. ix, 30–32

PCA  Principal Component Analysis. iii, vii, 1, 2, 6, 7, 15, 17–21, 23, 35

POI  Point of Interest. iii, vii, 1, 15, 17, 19–22, 35

SCA  Side-Channel Analysis. 1, 2, 9, 15, 23, 35

SNR  Signal-to-Noise Ratio. vii, 1, 15, 16, 20, 21, 25, 28, 29, 35
Chapter 1

Introduction

Side-Channel Analysis (SCA) attacks have become a powerful tool for extracting secret information from cryptographic devices since the introduction of Differential Power Analysis (DPA) by Kocher et al. [21]. These attacks exploit the relationship between the side-channel measurements and the data-dependent leakage models to reveal some part of the key. The CPA method [7] is among the most efficient distinguishers when the relationship of the leakage and data can be approximated with a linear model. However, due to process variation in nano-scale devices and consequently the increase in the contribution of the leakage component of the power consumption, different leakage models become necessary. Since the performance of the CPA method strongly depends on the assumed (linear) leakage model, imprecise predictions can lead to complete failure of the method.

Another major cause of the suboptimal performance of key recovery attacks is the presence of noise in leakage traces. While the performances of all SCA distinguishers are similar for a large Signal-to-Noise Ratio (SNR) [26], in real world scenarios it is common that the physical leakage measurements contain a significant amount of noise originating from multiple sources such as the power supply, the specifics of the measurement set-up, the clock generator, parallel computations etc. As discussed by Mangard et al. [25], the success of SCA attacks is heavily dependent on the SNR, and thus multiple noise reduction methods such as filtering, PCA [20], LDA [17], singular spectrum analysis [18] etc. have been studied in the domain of SCA attacks.

One can conclude that there are two main directions to improve the methods for SCA key recovery that are ideally also combined. The first includes finding optimal distinguishers for suitable leakage models, and the second focuses on reducing the noise level in the measurements. In this work we address both of these directions.

1.1 Research Questions

In this thesis we look into the applications of linear transformation and feature extraction methods in the domain of SCA. In particular, we analyse the performance of PCA and LDA when they are used as dimensionality reduction and POI selection techniques for template analysis. Furthermore, we are investigating the possibility of utilizing the LDA for key extraction purposes. Within this context, we are addressing the following research questions:

1. How does the application of PCA and LDA as a POI selection method affect the performance of template attacks?
2. Is there a possibility of applying LDA for side-channel key recovery? What are the characteristics of LDA transformation that can be exploited for distinguishing key candidates? How does this technique perform when compared to CPA?

While we are conducting our research on the side-channel measurements obtained from the software implementations of Advanced Encryption Standard (AES) and Data Encryption Standard (DES), the research methodology and conclusions can be applicable to other cryptographic algorithms and also to hardware implementations.

1.2 Structure

The preliminaries on information theory, dimensionality reduction and SCA are provided in Chapter 2. In Chapter 3 we describe the experimental setup and analyse the application of PCA and LDA in the domain of template analysis. We also compare their performance by visually inspecting the leakage transformation and calculating the success rate of correctly matching the traces to pre-built templates. In Chapter 3 we are proposing a novel key recovery method that relies on the LDA transformation as a pre-processing technique and distinguishes key candidates based on the classification. We discuss the conclusions in Chapter 5 and investigate the possibilities of further research.
Chapter 2

Preliminaries

2.1 Information Theory

In 1948 Claude Shannon demonstrated how information could be numerically quantified, analysed and expressed as a mathematical formula [33]. Although the paper was mainly targeting the field of communications, its concept and methodology has attracted a lot of attention and lead to the creation of a new scientific field – Information Theory. Nowadays, the theory has become ubiquitous and is used in different areas such quantum computing, physically unclonable functions, pattern recognition, anomaly detection etc. It also serves multiple purposes in the field of cryptography. The concepts of entropy and conditional entropy are used as the measures of secrecy of cryptosystems during their design. The same measures together with the mutual information can also be used by adversaries for side-channel key recovery purposes [3].

2.1.1 Random Variables

A random variable [11] is a quantity that associates a unique numerical value with every outcome of a non-deterministic experiment. The lack of predictability in such experiments leads to the variation of the random variable even when the same process is repeated. Throughout this paper, random variables and their values are denoted with upper-case and lower-case letters correspondingly. Calligraphic letters are used to denote the set of their possible outcomes. For every random variable $X$ drawn from a set $\mathcal{X}$ the following always holds.

$$\sum_{x \in \mathcal{X}} \Pr[X = x] = 1.$$ \hspace{1cm} (2.1)

When the space $\mathcal{X}$ is a continuum the probability of getting exactly $x$ is always zero and the above-mentioned equation becomes invalid. For the purposes of this work we disregard these cases and only assume discrete random variables where the space $\mathcal{X}$ is finite.
2.1.2 Joint and Conditional Probability

It frequently happens that a single experiment produces multiple outcomes. In these cases the joint or conditional probability of the outcomes are often of the interest. Let \( X \in \mathcal{X}, Y \in \mathcal{Y} \) be random variables. Then, the joint probability of \( X \) and \( Y \) is expressed as \( \Pr[X = x, Y = y] \) and the Equation (2.2) holds true.

\[
\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} \Pr[X = x, Y = y] = \sum_{x \in \mathcal{X}} \Pr[X = x] = 1 \quad (2.2)
\]

The conditional probability is denoted as \( \Pr[X = x|Y = y] \) which can be read as the probability of \( X \) given \( Y \). It expresses the probability of \( X \) being equal to \( x \) when the value of \( Y \) is known. The conditional probability can be expressed in terms of the joint probability as shown in Equation (2.3).

\[
\Pr[X = x|Y = y] = \frac{\Pr[X = x, Y = y]}{\Pr[Y = y]} \quad (2.3)
\]

When the random variables \( X \) and \( Y \) are independent from each other, the joint probability becomes equal to the product of individual probabilities

\[
\Pr[X = x, Y = y] = \Pr[X = x] \Pr[Y = y], \quad (2.4)
\]

which leads to \( \Pr[X = x|Y = y] = \Pr[X = x] \) from the Equation (2.3).

2.1.3 Shannon Entropy

Shannon Entropy quantifies the uncertainty one has about a random variable. For a random variable \( X \in \mathcal{X} \) the Shannon entropy \( H(X) \) is calculated as:

\[
H(X) = \sum_{x \in \mathcal{X}} \Pr[X = x] \log_2 \frac{1}{\Pr[X = x]} \quad (2.5)
\]

According to Shannon’s formula the entropy of a random variable is maximised if it has a uniform probability distribution. In a special case when the size of the outcome set is 2, i.e \( \mathcal{X} = \{0, 1\} \), with the probabilities \( \Pr[X = 0] = p \) and \( \Pr[X = 1] = 1 - p \), the binary entropy function \( h(p) \) is calculated as shown in Equation (2.6).

\[
h(p) = p \log_2 \frac{1}{p} + (1 - p) \log_2 \frac{1}{1 - p} \quad (2.6)
\]

It should be noted that \( \lim_{x \to 0} x \log_2 \left( \frac{1}{x} \right) = 0 \), therefore, the expression \( p \log_2 \left( \frac{1}{p} \right) \) can be said to be equal to 0 when \( p = 0 \).

2.1.4 Joint and Conditional Entropy

Shannon entropy can also be extended to multivariate random variables. The joint entropy of the random variables \( X \in \mathcal{X} \) and \( Y \in \mathcal{Y} \) is defined similarly as in
2.1. Information Theory

Analogical to the conditional probability, conditional entropy is the amount of unevenness about a random variable $X \in \mathcal{X}$ when the outcome of $Y \in \mathcal{Y}$ is known. It can be calculated as

$$H(X|Y) = \sum_{y \in \mathcal{Y}} \Pr[Y = y] \sum_{x \in \mathcal{X}} \Pr[X = x|Y = y] \log_2 \frac{1}{\Pr[X = x, Y = y]}$$

(2.8)

which leads to the following relation between the conditional and joint entropy:

$$H(X, Y) = H(X|Y) + H(Y)$$

$$= H(Y|X) + H(X)$$

(2.9)

The joint entropy of 2 random variables is not larger than the sum of their individual entropies, with the equality holding iff they are independent from each other. Similarly, it is not less than the entropies of individual random variables, with the equality holding iff one of the random variables is completely dependant on the other.

$$\max[H(X), H(Y)] \leq H(X, Y) \leq H(X) + H(Y)$$

(2.10)

2.1.5 Mutual Information

Mutual information measures the information overlap between random variables. It can also be understood as the measure of the full correlation between them. If and only if the random variable are independent from each other, the mutual information between them becomes 0. For random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ it is
Chapter 2. Preliminaries

\[
I(X; Y) = \sum_{x \in X} \sum_{y \in Y} \Pr[X = x, Y = y] \log_2 \frac{\Pr[X = x, Y = y]}{\Pr[X = x] \Pr[Y = y]}
\]  

(2.11)

By substituting the joint probability according to the Equation (2.3) the mutual information can also be written as the functions of entropies:

\[
I(X; Y) = H(X) + H(Y) - H(X, Y)
\]

\[= H(X) - H(X|Y) \]

\[= H(Y) - H(Y|X) \]

(2.12)

2.2 Dimensionality reduction

Let \( X \) denote a random variable over a space \( \mathcal{X} \), and let \( x \) be a realization of \( X \). Then, \( X^d \) is a \( d \)-dimensional random row vector \((X_1, X_2, \ldots, X_d) \in \mathcal{X}^d \) with a realization \( x^d \). When the value of \( d \) is very large it becomes computationally very complex to perform statistical analysis on the data set. Indeed, recent advances in technology have made it possible to acquire very high-resolution measurements from the subjects under test. This, in its turn has triggered a higher need for the usage of dimensionality reduction techniques. Alongside the speed-up due to the reduction in the number of dimensions, such techniques usually also increase the performance of the classification/clustering.

The most widely used dimensionality reduction techniques are based on orthogonal projections. Amongst these the PCA and LDA have been thoroughly studied in the domain of side-channel analysis.

2.2.1 Principal Component Analysis

PCA [20] is one of the oldest multivariate statistical techniques that has been applied in numerous scientific fields. Its goals are to analyse a set of observations of random variables and to express them in terms of low-dimensional orthogonal variables that hold maximum information about these observations. To achieve these goals, the principal components are calculated as the linear combinations of
the original observations. The components are required to be ordered such that the amount of information held by them decreases from the first to the last one.

The calculation of the principal components is performed in a few steps:

- The mean of all the $d$-dimensional observations $x_i^d$ are calculated.

$$\mu = \frac{1}{n} \sum_{i=1}^{n} x_i^d \quad (2.13)$$

- The calculated mean is subtracted from the observation to obtain mean centred observations $\bar{x}_i^d$.

$$\bar{x}_i^d = x_i^d - \mu \quad (2.14)$$

- The covariance matrix $\Sigma$ of the centred observations is calculated. Let us denote the matrix that populates $n$ $d$-dimensional observations in its rows $X_{n,d}$. Then, $X_{n,d}$ is the matrix of centred observations. The covariance matrix can be calculated as:

$$\Sigma = \frac{1}{n} \bar{X}_{n,d} \left( \bar{X}_{n,d} \right)^T \quad (2.15)$$

- At the last step, the covariance matrix is eigendecomposed to obtain the matrix of eigenvectors $U$ and the diagonal matrix of eigenvalues $\Delta$.

$$\Sigma = U \Delta U^{-1} \quad (2.16)$$

The eigenvector corresponding to the largest eigenvalue is called the first principal component. The rest of the components are ordered according to the values of the eigenvalues. Since, most of the information about the observations is contained in $m < d$ components, by keeping only these directions the dimensionality of the dataset is reduced significantly. The matrix of transformed observations $\hat{X}^{n,m}$ is calculated according to Equation (2.17) where $\hat{W}$ is the projection matrix constructed as the first $m$ columns of $U$.

$$\hat{X}^{n,m} = X_{n,d} \hat{W} \quad (2.17)$$

### 2.2.2 Linear Discriminant Analysis

LDA [17] is a dimensionality reduction method used for supervised classification purposes in machine learning, pattern recognition, etc. Unlike PCA which preserves as much variance as possible, LDA seeks for projection directions that preserve as much class discriminatory information as possible. It projects the observations of random variables into a low-dimensional space where the ones corresponding the same class are very similar to each other while being significantly dissimilar to the ones in other classes. To put it in other words, LDA transformation looks for projection directions that maximise the ratio of between-class ($S_B$) and within-class ($S_W$) scatters. The maximisation problem is solved by finding the
transformation matrix $\mathbf{W}$ such that $J(\mathbf{W})$ [eq. (2.18)] is maximum.

$$J(\mathbf{W}) = \frac{\mathbf{W}^T \mathbf{S}_B \mathbf{W}}{\mathbf{W}^T \mathbf{S}_W \mathbf{W}}$$  \hspace{1cm} (2.18)

The whole transformation process is performed as follows:

- Each observation is assigned to a class $C_j \in \mathcal{C}$. The $i$-th observation in class $C_j$ is denoted as $x_{i,j}^d$ and $N_j$ is the number of observations in this class.

- The means of the classes $\mu_j$ and all of the observations $\mu$ are calculated.

$$\mu_j = \frac{1}{N_j} \sum_{i=1}^{N_j} x_{i,j}^d$$  \hspace{1cm} (2.19)

$$\mu = \frac{1}{n} \sum_{j=1}^{\mid \mathcal{C} \mid} \sum_{i=1}^{N_j} x_{i,j}^d$$  \hspace{1cm} (2.20)

- Within-class and between-class scatter matrices are calculated.

$$\mathbf{S}_B = \sum_{j=1}^{\mid \mathcal{C} \mid} N_j (\mu_j - \mu)(\mu_j - \mu)^T$$  \hspace{1cm} (2.21)

$$\mathbf{S}_W = \sum_{j=1}^{\mid \mathcal{C} \mid} \sum_{i=1}^{N_j} (x_{i,j}^d - \mu_j) (x_{i,j}^d - \mu_j)^T$$  \hspace{1cm} (2.22)

- The matrix of eigenvectors $\mathbf{U}$ and eigenvalues $\Delta$ are obtained.

$$\mathbf{S}_W^{-\frac{1}{2}} \mathbf{S}_B \mathbf{S}_W^{-\frac{1}{2}} = \mathbf{U} \Delta \mathbf{U}^{-1}$$  \hspace{1cm} (2.23)

- The projection directions are calculated.

$$\mathbf{W} = \mathbf{S}_W^{-\frac{1}{2}} \mathbf{U}$$  \hspace{1cm} (2.24)

The truncated projection matrix $\tilde{\mathbf{W}}$ can be constructed by keeping the columns of $\mathbf{W}$ corresponding to the first $m$ highest eigenvalues of $\Delta$ according to the Eckart-Young theorem [13]. Consequently, the matrix of projected observations is calculated according to Equation (2.25).

$$\hat{\mathbf{X}}_{n,m} = \mathbf{X}_{n,d} \tilde{\mathbf{W}}$$  \hspace{1cm} (2.25)

### 2.3 Side-Channel Analysis

Many computational devices use cryptographic algorithms to ensure confidentiality, integrity and authenticity of data. These algorithms use a secret cryptographic key to map a user given input to a ciphertext. Most commonly used cryptographic algorithms follow the principles set by Dutch cryptographer Auguste Kerckhoffs [28] and do not rely on *security by obscurity*. Therefore, they are designed to ensure that even a full knowledge about the implementation and the input/output pairs
2.3. Side-Channel Analysis

of the algorithm makes the extraction of the secret information practically infeasible.

Nevertheless, in the last few decades several methods have been developed to extract the secret key from the cryptographic devices. These methods, also known as attacks, can be classified according to different criteria. One way to categorise them is to differentiate between active and passive attacks. If the attack manipulates the environment or the inputs of the device to reveal the key by the observation of its abnormal behaviour, it is considered as the active attack. On the other hand, passive attacks refer to the ones where the device operates within its specifications and the key is extracted by observing its physical properties.

Another way to classify the methods is based on which interfaces are used for conducting the attack. Within this context, the attacks can be categorised as invasive or non-invasive. 

Invasive attacks are not bound by any restrictions regarding to which extent the device can be manipulated. They usually start by de-packaging the device to have a direct access to its hidden interfaces or electronic components. The secret key can be revealed either by passively observing the signals [32] or actively manipulating the cryptographic process through these interfaces [34, 35, 23, 31]. The attacks of this category which do not require a direct contact to the chip surface are sometimes referred to as semi-invasive attacks.

Non-invasive attacks are mounted only through the directly available interfaces of the device. An active non-invasive attack can be conducted on the device by changing the operational temperature, causing glitches, etc. [2, 9]

In particular, the relatively low cost of conducting passive non-invasive attacks on cryptographic devices has attracted a lot of attention. Timing attacks [22, 6], power analysis attacks [27], and electro-magnetic (EM) emanation analysis attacks [19, 29] are among the most popular SCA methods. The latter two of these techniques are based on similar theoretical primitives and we use these types of attacks for conducting our research.

The power analysis attacks introduced by Kocher et al. [21] showed that, while a cryptographic algorithm might be mathematically secure, it was still possible to obtain the secret information by conducting a statistical analysis. SCA exploits the fact that the processed data and performed operation influences the instantaneous power consumption of cryptographic device. In this work we adopt the terminology and notations of [3], and consider the schematic representation of a classic SCA represented in Figure 2.3 to give the general description of the analysis.

In this scenario, a targeted cryptographic implementation is performing an encryption $E_k(p)$ of the plaintext $p$ using a constant key $k$. During computation, the sensitive intermediate value $V_{s,p}$ that depends on a part $s$ of the key $k$, and the plaintext $p$ are handled. The physical leakage generated during the computation of $V_{s,p}$ is denoted as $Y_{k,p}$, since the leakage may potentially depend on the whole key $k$. The adversary acquires leakage traces by sampling or measuring the side-channel observables (power, electromagnetic emanation) at successive time instances. The value $Y_{k,p}$ can be captured in one sample or spread over multiple samples depending on the implementation details and the parameters of the acquisition.

To recover the key, the adversary predicts the intermediate values handled during the computation of $E_k(p)$ and calculates the values $V_{j,p}$ for every possible subkey
Chapter 2. Preliminaries

candidate $j \in S$ and the given plaintexts $p$. The adversary maps the intermediate values $V_{j,p}$ to the hypothetical leakage value $X_{j,p}$ by applying a guessed leakage model.

The last step of the key recovery is to compare the hypothetical leakage values $X_{j,p}$ to the actual measurements $Y_{k,p}$ using a distinguisher function $D$. The correct subkey candidate is obtained as the value of $j$ which maximizes the distinguisher function. To recover $k$ the same steps are repeated for all the subkeys $s$.

2.3.1 Correlation Power Analysis

Different methods such as Mutual Information Analysis, Difference of Means, CPA, etc. have been proposed in the literature to compare the hypothetical power consumption to the actual measured leakages. Authors of [12] show that when the model guessed for mapping the intermediate values to the hypothetical power consumption exactly corresponds to the actual leakage function of the cryptographic device, CPA is the most efficient distinguisher. The fundamental assumption of the CPA is the linear relationship between the hypothetical and actual power consumptions. To perform an attack using this method, adversary first maps the predicted intermediate values to the hypothetical power consumption by applying a guessed power model. Afterwards, the linear relationship between these values and the leakage traces is calculated as Pearson’s correlation coefficient. This coefficient is computed according to Equation (2.26) and is the measure of linear dependence between random variables $X$ and $Y$.

$$\rho(X, Y) = \frac{cov(X, Y)}{\sigma_X \cdot \sigma_Y} = \frac{E[XY] - E[X] \cdot E[Y]}{\sigma_X \cdot \sigma_Y} \tag{2.26}$$

In this equation, $E[\cdot]$ denotes the expectation value and $\sigma$ the standard deviation of a random variable. The correlation coefficient can have values in the interval of $-1$ to $1$. The boundaries of this interval are reached if and only if $Y$ is an affine function of $X$. On the other hand, if $X$ and $Y$ are independent from each other, then value $0$ is achieved. It should be noted that the opposite of the latter relationship does not always hold: it is possible that $X$ and $Y$ are dependent while the correlation coefficient is equal to $0$. 

![Figure 2.3: Schematic illustration of a side-channel key recovery](image-url)
2.3.2 Template Analysis

The template attacks (TA) introduced in [10] are considered to be the strongest type of side-channel attacks from information theoretic point of view. Unlike other attacks they can successfully extract secret information from limited number of side-channel traces. The attacks are carried out in two main steps: (i) Profiling step is needed to build templates corresponding to each subkey value - key class; (ii) Template matching step is executed to match the newly obtained trace to the pre-defined templates.

In order to carry the attack the adversary first identifies a part of the trace which depends only on a few bits of the secret data. Then, for each possible value of these bits multiple number of traces are measured and templates that consist of the mean signal and noise probability distribution are constructed. Within the context of the template attacks it is assumed that the noise has a multivariate Gaussian distribution.

Let \( x^d \) denote the \( i \)-th \( d \)-dimensional trace corresponding to the key class \( j \) and suppose \( N_j \) for each of the key classes \((C_1, \ldots, C_k)\) are recorded. The template \( \mathcal{N}(\cdot|\mu_j, \Sigma_j) \) for the key class \( j \) is constructed as shown in eq. (2.27).

\[
\mu_j = \frac{1}{N_j} \sum_{i=1}^{N_j} x_{j,i} \quad \Sigma_j = \frac{1}{N_j} \sum_{i=1}^{N_j} (x_{j,i} - \mu_j)(x_{j,i} - \mu_j)^T \quad (2.27)
\]

In the template matching process the adversary records the leakage trace \( x^d \) and compares it against the existing templates. Usually maximum likelihood method is used [10, 1, 37] for this purpose (See eq. (2.28)).

\[
p(x^d|\mu_j, \Sigma_j) = \frac{2}{\sqrt{(2\pi)^d|\Sigma_j|}} \exp(-\frac{1}{2}(x^d - \mu_j)\Sigma_j^{-1}(x^d - \mu_j)^T) \quad (2.28)
\]

A practical challenge for the conduction of template attacks is the large power and memory usage for the calculation of the covariance matrix \( \Sigma_j \), its determinant and inverse for large \( d \). The general idea for solving this challenge is selecting the points which are “interesting” for the side-channel analysis. Several approaches have been suggested for selecting such points of interest. Chari et al. [10] suggests to compute the pairwise difference between the mean signals and to select the points which have the largest difference. Rechberger and Oswald [30] suggest to select the points with the largest cumulative difference. Additionally, they also apply Fast Fourier Transform (FFT) to the traces to remove the noise with high frequency.

2.4 Cryptographic Algorithms

2.4.1 Data Encryption Standard

The Data Encryption Standard has been developed by IBM in 1970 and was adapted as a Federal Information Processing Standard in 1976 [16]. While it has been proven to be insecure [41, 24], its adaptation – 3DES is still considered to be secure and is widely used in practice. This algorithm encrypts/decrypts data by applying DES three times to achieve a desirable level of security.
DES is a block cipher that encrypts 64-bit long data with a 64-bit key to calculate the ciphertext of the same length. 8 bits of this key are used for error detection purposes which leave only 56 bits for encryption. The algorithm consists of two separate processes: (i) key scheduling; and (ii) round transformation.

**Key scheduling** process generates 16 *round keys* each having 48-bits. During this procedure the cipher key is initially permuted and then separated into two equal sized halves. The round keys are constructed iteratively by shifting the halves by one or two bits (depending on the round) to the left and selecting 24 bits from each of them.

**Round transformation** process is performed 16 times to obtain the ciphertext. Before the first transformation, the plaintext is permuted by pre-defined constant *Initial Permutation* function to obtain the *state*. For each round the following operations are performed:

- The state is divided into two 32-bit halves, a left and a right half.
- The right half is assigned as the left half of the output.
- The right half is expanded to 48-bit array by applying yet another constant function called *Expansion Permutation*.
- The result is exclusive-ored with the corresponding round-key and passed to the substitution function.
- The 32-bit output of the substitution function is permuted via a fixed permutation (P-box).
- The result is exclusive-ored with the left half and assigned as the right half of the output.

After 16 iterations, the left and right halves of the resulting state are swapped and then permuted by *Final Permutation* function to obtain the ciphertext.

The algorithm uses 8 pre-defined byte substitution functions i.e. S-boxes. Each of them takes 6 bits of the intermediate state and maps them to 4-bit outputs.

### 2.4.2 Advanced Encryption Standard

Rijndael has been officially announced as the Advanced Encryption Standard (AES) on October of 2000 [15] with an intention to completely replace DES. Since then, the standard has been adopted by various organisations such as banks, industries and governmental institutions. Therefore, we have chosen it as another target implementation for our research purposes.

AES is a block cipher that encrypts 128-bit long data with a key that can have either 128, 192 or 256 bits. We consider the 128-bit version of the algorithm which is usually abbreviated as AES-128. In this case, 16 bytes of data and the key are represented by $4 \times 4$ rectangular array in column-major order. The whole encryption procedure can be described as two separate processes as in the previous algorithm: (i) key scheduling; and (ii) round transformation.

**Key scheduling** process generates the *round keys* which are later used as an input by round transformation algorithm. 11 round keys, each having 128 bits are constructed iteratively from the given key which is also the first round key.
**Round transformation** process consists of four different operations - SubBytes, ShiftRows, MixColumns and AddRoundKey executed in the given order. These operations are performed on a $4 \times 4$ array which is called the *state*. The state is initialised as being equal to the data and the encryption starts by performing AddRoundKey operation on it. Then, nine round transformations are executed and finally, the tenth round which skips the MixColumns operation is performed to obtain the resulting ciphertext. Each of these operations can be described as follows:

**AddRoundKey**
- The key addition operation that exclusive-ors the round key with the state.

**SubBytes**
- The byte substitution operation that is applied to each byte of the state separately. The substitution is performed by a non-linear mapping which is also called S-box function.

**ShiftRows**
- The row shifting operation that performs a cyclic shift of the rows with different offsets. The first row of the state is not shifted while the second, third and fourth rows are shifted by 1, 2 and three bytes to the left in the corresponding order.

**MixColumns**
- The column mixing operation that treats each column of the state as a polynomial over Galois Field of order $2^8$ and multiplies it with a fixed polynomial $c(x) = 3x^3 + x^2 + x + 2$ in modulo $x^4 + 1$. Together with the ShiftRows operation, it is the main source of diffusion in round transformation.
Chapter 3

Template Attacks in Principal Subspaces

Several solutions have been proposed to address the challenges of conducting template attacks as described in Section 2.3.2. However, these approaches rely on heuristics for determining the samples in the leakage traces that can be used for modelling the templates. A more systematic technique is applying dimensionality reduction methods for selecting such POI.

In this chapter we analyse the performances of the template attacks based on two mostly used dimensionality reduction techniques, namely, PCA and LDA. We compare the results to the case where the POI are selected as the ones which have the largest correlation with the hypothetical power consumption. We report the performance of the attacks by looking at the Global Success Rates (GSR), i.e. the ratio of the correctly guessed subkeys to the total number of subkeys.

3.1 Experimental Setup

For this research, we consider software implementations of AES128 [15] and DES [16] running on an ARM Cortex-M4F core based board operating at a 168 MHz clock frequency. The board has been physically modified and programmed in order to be a target for SCA and it accurately models current 32-bit embedded devices. We acquire EM and power measurements for each of the implementations where the former have lower SNR due to the higher levels of measurement noise. To do so, we build a standard setup (as described e.g. in [25]). We utilize a PicoScope 3207B [38] digital oscilloscope with a 500 MHz sampling rate. We carry out four measurement campaigns (i.e., two for each cryptographic algorithm implementation), with the following parameters (See Figure 3.1):

\textit{TraceSet}_1: 50,000 power traces were obtained for the implementation of the AES128 algorithm. The key was fixed and the traces were obtained for random plaintext inputs. The SNR value is 16.8 dB.

\textit{TraceSet}_2: 50,000 EM traces were obtained for the implementation of the AES128 algorithm. The key was fixed and the traces were obtained for random plaintext inputs. The SNR value is 1.01 dB.

\textit{TraceSet}_3: 50,000 power traces were obtained for the implementation of the DES algorithm. The key was fixed and the traces were obtained for random plaintext inputs. The SNR value is 18.2 dB.
Chapter 3. Template Attacks in Principal Subspaces

FIGURE 3.1: The experimental setup for dual-channel trace acquisition.

TraceSet: 50,000 EM traces were obtained for the implementation of the DES algorithm. The key was fixed and the traces were obtained for random plaintext inputs. The SNR value is 2.78 dB.

The noticeable difference in the SNR values of the similar (EM or power) measurements originates from the architectural designs of the implementations. The parallel S-box lookups during the AES rounds generates more algorithmic noise which leads to a lower SNR value.

3.2 Template Building

Instead of building templates for the parts of the secret key, we build them for hypothetical power consumption of intermediate values that are generated at some point during the encryption. The power model for both implementations are chosen as the Hamming weight (HW) of the intermediate values. In the case of AES128 encryption, the chosen intermediate values are the inputs to the S-box during the first round. These inputs correspond to the bitwise XOR of the first byte of the plaintext with the first byte of the secret key. As the result we obtain 9 different values which will be called key classes throughout this chapter.

The intermediate values targeted during the DES encryption are the 6-bit inputs to the first S-box during the first encryption round. In this case 7 (i.e. HW from 0 to 6) different key classes are obtained.
3.2. Template Building

Dimensionality reduction methods introduced in the Section 2.2 are applied for finding the transformation matrix that preserves the maximum amount of information about the traces in a sub-space with dimensions less than the number of key classes. The traces are transformed into the new sub-space and templates for each key class are built as a pair of their mean and covariance matrices. 85% of the traces are used for building the templates for each trace-set while the remaining traces are used for testing purposes.

3.2.1 PCA-Based POI Selection

The mean traces each associated with a different key class are plotted in Figure 3.2. The application of PCA to these traces will find the linear transformation that maximises the variance between these empirical mean traces. In this case, instead of the leakages, the covariance matrix of their means is eigendecomposed.

\[
\bar{\mu} = \frac{1}{|C|} \sum_{j=1}^{\left| C \right|} \mu_j
\]

\[
\bar{\Sigma} = \frac{1}{|C|} \sum_{j=1}^{\left| C \right|} (\mu_j - \bar{\mu})(\mu_j - \bar{\mu})^T
\]

After constructing the transformation matrix \( \bar{W} \), the principal subspace templates \((\nu_j, \Lambda_j)\) are computed by transforming the template pairs \((\mu_j, \Sigma_j)\) according to...
Chapter 3. Template Attacks in Principal Subspaces

Equation (3.3).

\[ \nu_j = \mu_j \tilde{W} \quad \Lambda_j = \tilde{W}^T \Sigma_j \tilde{W} \quad (3.3) \]

The main discriminatory information about the templates are accumulated in the first and the second components after the transformation. Therefore, the visual inspection of these components can indicate the quality of the templates. The more separable the traces corresponding to different key classes are, the better performance during the template matching step should be expected.

The analysis of the first 2 principal components in Figure 3.3 reveals that the power traces measured for both implementations lead to a better separation among different classes. It is possible to visually identify each of the clusters. On the other hand, no clear distinction can be made among the classes for the transformed EM traces. Such distribution of the principal components makes the probability of assigning the newly measured trace to pre-built templates very low.

3.2.2 LDA-Based POI Selection

The objective of the LDA transformation is to find the projection directions that apart from capturing the maximum variance between the classes, also preserves minimum scatter within each of them.
3.3 Template Matching

In the template matching stage the newly acquired test traces are transformed according to the projection directions obtained during the template building step. Then, for every $m$-dimensional transformed leakage trace $\tilde{x}^m_l$ the likelihood of it being equal to the each of the templates $(\nu_j, \Lambda_j)$ is calculated according to the Equation (3.4) where $J \sim B(n, \frac{1}{2})$ can be approximated by a binomial distribution with $n = 9$ for AES-128 and $n = 7$ for DES encryption (See Figure 3.5). The template with the maximum likelihood score is chosen as the predicted key class and the success rate is calculated as the ratio of the correct guesses to the total number

The steps for achieving this goal are the same as described in Section 2.2.2. After assigning the leakage traces to different groups based on the value of the corresponding key classes, the mean traces of each group and the overall mean of the whole trace set are determined. Then, the between-class and within-class scatter matrices are calculated as shown in Equations (2.21) and (2.22). After the construction of the projection directions and the transformation of the traces to the new sub-space (See Equations (2.23) to (2.25)), the discriminatory information is captured in the first two directions similar to the previous case.

The visual comparison of the LDA transformation shown in Figure 3.4 to that of PCA displays the significant difference between two methods. Unlike the previous case, this transformation achieves clear separation even among the noisy EM traces.

Figure 3.4: Visualisation of the leakage traces after their projection into the sub-space determined by LDA.
Chapter 3. Template Attacks in Principal Subspaces

Figure 3.5: Empirical and binomial probability distributions of the key classes for AES and DES implementations.

\[
p(\nu_j, \Lambda_j | \hat{x}_m) = \frac{2}{\sqrt{(2\pi)^m|\Lambda_j|}} \exp \left( -\frac{1}{2} (\hat{x}_m - \nu_j) \Lambda_j^{-1} (\hat{x}_m - \nu_j)^T \right) \Pr[J = j]
\]

(3.4)

3.4 Results

For comparing the GSR of the template attacks run by using PCA and LDA as POI selection methods, experiments are conducted for all of the trace sets. Figures 3.6 and 3.7 show that as expected after the visual inspection of the transformed traces, the projection directions extracted via LDA transformation lead to the better performance of the template attacks when compared to PCA transformation.

For both of the methods it can be further observed that, the GSR for the EM traces are significantly lower than the power traces. Since the SNR level is directly related to the success rates [14], the higher amount of noise in the EM traces leads to the poor performance of the template attacks. While in the case of the PCA transformation the attacks run on EM traces fail completely, near-optimal SNR level achieved after the LDA transformation [8] results in higher success rates even in the presence of large noise in the traces.

The experiments are also further extended to determine the performance of the template attacks where the POI are selected as the samples with the highest correlation coefficient. To do so, the hypothetical power consumptions are predicted as the HW of the S-box outputs for both of the implementations. The correlation between these consumptions and the leakage traces are calculated and the transformed traces are constructed from samples with the highest coefficients. As can be observed from Figure 3.8, the success rates in this case are also significantly lower than the LDA based transformation.

From the visualisation of the transformations and the analysis of the GSR it can be concluded that when applied as the feature extraction technique LDA finds the projection directions that achieves better separation among the key classes. Such separation in its turn also leads to better matching rate of the newly acquired traces to the pre-defined templates. Even in the case of low SNR this method outperforms
the other techniques, as it builds the projection directions based on the discriminatory information rather than the variance or correlation.
Chapter 3. Template Attacks in Principal Subspaces

**Figure 3.7:** Success rates for the POI selected with LDA transformation

**Figure 3.8:** Success rates for the POI selected as the samples with the highest correlation coefficient
Chapter 4

(De-)classification of Keys

Apart from its application in the domain of the template analysis PCA has also been studied for both data preprocessing and as a method for key recovery. Batina et al. [4] propose to utilize it as a preprocessing technique before conducting the DPA attack. The observed benefits of PCA in such scenarios is the noise reduction when the location of the leakage is known in the traces. Moreover, better performance of the DPA after the transformation of the traces into a lower dimension subspace spanned by eigenvectors is also observed.

In contrast to being used as a pre-processing tool, Souissi et al. [36] have investigated the applicability of the PCA as another distinguisher by merely using the first principal component. The method which is called the First Principal Component Analysis (FPCA) distinguishes key candidates based on the dispersion of the references. To conduct an attack, the trace set is partitioned based on a some criteria (e.g. the HW of the intermediate values) for each key hypothesis and the references for each partition are constructed based on some chosen statistics. Then, these traces are centred and their covariance matrix is eigendecomposed as shown in Section 2.2.1 to obtain the matrices of eigenvectors and eigenvalues. The largest eigenvalue for each key hypothesis is considered its distinguishing score and the candidate with the highest score is selected as the key.

The ANOVA (ANalysis Of VAriance) F-test is using a distance measure between the classes, which has similarities to the LDA transformation. The metric called Normalized Inter-Class Variance (NICV) is used for leakage detection in SCA. While efficient in determining the time where the sensitive information is computed, comparing different leakage models or speeding up attacks on asymmetric cryptography, this method cannot be used as a distinguisher for recovering the secret information.

Differential Cluster Analysis (DCA) introduced by Batina et al. [5] is also approaching the key recovery task as a classification problem. The authors use metrics like sum-of-squared-error and sum-of-squares to obtain the cluster statistics. While the method does not depend on a good leakage prediction it can benefit from it to achieve a better performance.

After the analysis of two most popular linear transformation and dimensionality reduction methods in Chapter 3 we observe the superior performance of LDA in extracting the features from the leakage traces. The better separation of the classes in the transformed subspace and the near-optimal noise reduction after the projection are the main reasons behind the success of the technique.
In this chapter we propose to exploit such properties of LDA and to apply the method for recovering the key from the observed leakage traces. We show how to classify the observations without the knowledge of the key and to transform them into a low-dimensional subspace. Afterwards, we introduce a distinguisher which makes use of the characteristics of the transformation for extracting the key.

The rest of the chapter is organized as follows. In section we describe the operation required at different stages of the attack method. In section we address the caveats that arise due to the intrinsic characteristics of LDA transformation. The results of the experiments are reported in section. Furthermore, we compare the GSR of the attacks performed using the proposed method to those of a standard CPA attack.

4.1 Attack description

The key recovery attack proposed in this chapter relies on the central assumption that all leakages corresponding to the processing of some fixed key-dependent intermediate value are similar. In other words, when a set of physical leakages $Y_{k,p}$ is classified according to the values of $X_{s,p}$ as defined in Section 2.3, the between-class to within-class scatter matrices ratio is large.

Note that the above requirement is indeed met in the context of side-channel attacks, as the instantaneous power consumption of a cryptographic implementation is generally expected to be data dependent. However, in practice side-channel measurements often include noise, which leads to a weaker separation amongst classes and in consequence decreases the success rate of key recovery attacks.

The approach proposed in this chapter targets such challenging scenarios where the SNR is low, and achieves a better key extraction with a smaller amount of traces. It consists of two steps: (i) the leakage transformation step; and (ii) the distinguishing step.

In the following we describe in more detail the working principles of our attack. In Section 4.1.1 we describe how parts of the plaintext can be used for classification purposes and how measured leakages can be projected into a subspace where they are maximally separated and the SNR level is higher. Then in Section Section 4.1.2 we propose a function that enumerates subkeys based on the separation of the model based classes.

4.1.1 The Leakage Transformation Step

The objective of the leakage transformation step is to identify and select time samples where the difference between mean traces corresponding to distinct classes of intermediates is maximized. Furthermore, this step increases the SNR which, unlike other noise reduction methods such as band-pass filtering, aligning, Single Spectrum Analysis (SSA) [18], etc. takes into account the target part of data corresponding to the traces.

In order to apply a LDA transformation in this step, information that allows for the separation of traces into classes must be available, e.g. one must know the plaintexts or ciphertexts. The sensitive key dependent intermediate variables are predicted as $V_{j,p}$, as represented in Figure 4.1. Although the correct intermediate
values $V_{s,p}$ depend on the unknown subkey $s \in S$, they may still be classified based only on the value of the plaintext due to the fact that for any $j \in S$ and $(p_1, p_2) \in P$, if $p_1 = p_2$ then $V_{j,p_1} = V_{j,p_2}$. After separating the physical leakages into groups based on the plaintext or ciphertext values, the projection directions are calculated and the leakages are projected onto the new subspace. The transformed leakages are subsequently used for key recovery, as represented in Figure 4.1.

### 4.1.2 The Distinguishing Step

The objective of this step is to distinguish between the key candidates. Since in the previous step the traces have been linearly transformed to maximize the separation
between classes, it cannot be guaranteed that correlation between the traces and the hypothetical power consumption is preserved. By definition, the transformation is equal to calculating the inner product of each leakage with the projection directions where each direction is a column of the transformation matrix $\tilde{W}$. It follows that the magnitudes of the coefficients in each direction resemble the contribution of the corresponding samples to the transformation.

Figure 4.2 shows the Pearson correlation coefficients for each sample of $\text{TraceSet}_1$. It can be observed that the maximum correlation is achieved for the samples in the interval of 159 to 164. Since CPA takes the maximum correlation coefficient as the distinguishing score for each candidate, the contribution of the samples in this interval is of utmost significance to the correct extraction of the key.

In Figure 4.3 we can observe the projection direction that determines the first component of the traces after LDA transformation. As mentioned before, the magnitude of the coefficients in this direction is an indication of how much the corresponding sample is preserved in the component. Unlike the previous case the samples in the interval of 80 to 105, 249 and 326 can be seen to have the highest contribution to the first component. Since the presence of the samples with the highest correlation coefficients in this dimension is small, application of the CPA after the transformation does not guarantee an effective distinguishing among the key candidates. Therefore, the need for a new distinguisher that better matches the properties of the transformed traces arises. We propose to use the ratio of the between-class and the within-class scatter for this purpose.

The features extracted through the LDA transformation correspond to the linear combination of the leakage samples that maximally separate classes. At the same time, for a given leakage model, traces corresponding to the same values of $X_{s,p}$
4.1. Attack description

are expected to have similar features. Since for a given projection direction the contribution of each sample of the side-channel leakages towards this direction is of the same ratio, when the projected leakages are labelled according to the model obtained from the correct key, the separation of the clusters should be maximal. Whereas, if the model obtained from the wrong key is used for labelling, the lack of similar features within classes should lead to a weaker separation. Such behaviour can be visually observed in Figure 4.4. In Figure 4.4a the transformed traces are classified according to the HW of the S-box outputs where the inputs are the plaintext and the known key. When the classification of the same set is performed similarly but with a wrong key as an input the resulting plot becomes as Figure 4.4b.

Such characteristics of the classification can be exploited for distinguishing purposes. We propose to classify the transformed traces according to the values of the hypothetical power consumption for each key candidate. Later, the best candidate is selected based on the ratio of the between-class and within-class scatters. Since the objective of the distinguisher is to retrieve an ordinal information about the variance of the ratio matrix, its largest eigenvalue can be used as a numerical measure for separation [39].

Summarizing, in the second stage $|S|$ models (each corresponding to a different $j \in S$) are computed and the transformed physical leakages are classified accordingly. Subsequently, the ratio of the between-class and within-class scatter matrices is eigendecomposed. The distinguisher function $D$ is defined as

$$D\left(\hat{Y}_{k,P}, X_{j,P}\right) = \max(\hat{\Delta})$$

where $\hat{\Delta}$ is the diagonal matrix of eigenvalues of the ratio of between-class and within-class scatters of $\hat{Y}_{k,P}$, and $X_{j,P}$ is the vector of class labels generated according to the classification of the hypothetical power consumption. Finally, the candidate leading to the largest eigenvalue is selected as the correct key.
4.2 Caveats

Due to intrinsic characteristics of the LDA transformation, the proposed method has two caveats.

First, the number of side-channel traces must be larger than the number of analysed samples. To overcome the need for a very large trace set, it is possible to analyse only a selected block of samples at a time. In this case for each key candidate the number of discriminant scores will be the same as the number of blocks. If a selected block does not include samples related to the calculation of the predicted intermediate values, classification of the leakages according to possible values of the subkey candidate will not be significantly different from each other. Whereas, in the block where leakage occurs, the correct key candidate should lead to a significantly better separation among the classes. In order to find the block where the leakage occurs, the scores for each block have to be normalised and the one with the highest ratio of the scores for the first and second candidates is chosen as the leaking block. The first candidate of the leaking block is subsequently chosen as the correct key.

Second, the size of the plaintext space $P$ must be reasonably small. In order to estimate the between-class scatter, more than one trace should belong to each class. Since in the classification and transformation stage the number of classes is equal to $|P|$, the number of leakage traces needed for finding the projection directions would be significantly high. This restriction can be avoided by obtaining the measurements for chosen plaintexts such that the size of the text space is small.

4.3 Experimental Validation

We now validate our attack methodology using the EM trace sets (i.e. $TraceSet_2$ and $TraceSet_4$) described in Section 3.1 under different leakage assumptions. These sets are selected because they are representatives of the challenging scenarios where the SNR is low. We apply LDA transformation after classifying the traces according to the values of the target byte of plaintext to obtain the projected low-dimensional trace sets.

In Section 4.3.1 we describe the attacks where the hypothetical power consumption is linked to the HW of intermediate values, and in Section 4.3.2 we describe how the (partial) identity leakage model can be exploited. We report the performance of the attacks by looking at the GSR of the key extraction.

4.3.1 HW Leakage Model

As shown in Figure 4.1, the intermediate values for both of the implementations are predicted as $V_{j,p} = Sbox(j \oplus p)$ and the leakages are modelled as the HW of the intermediate values. The subkeys of the first round key are targeted at every implementation with the goal of recovering the full round key. As studied by Doget et al. [12], when the chosen power model exactly corresponds to the actual leakage function of the implementation, CPA has one of the best performances for key extraction. Therefore, we have used this method as a reference for comparing the performance of the proposed attack. It should be noted that while the CPA attack
4.3. Experimental Validation

is based upon an assumption of linear dependence between the HW of the interme-
diate values and the actual power consumption, our attack does not require such a
strict relation. We only assume that the power consumption corresponding to the
processing of intermediate values that have the same HW is consistent and it differs
from that corresponding to other HW values.

For CPA attacks, the hypothetical power consumption models for each possible
value of the subkey were built and the Pearson correlation coefficients were cal-
culated for each sample of the trace sets. The key candidate which maximizes the
absolute value of the correlation coefficient was chosen as the correct key. Both
the proposed attack and CPA were run on randomly selected subsets of the trace
sets multiple times and the average results were compared. Figure 4.5 reports the
GSR for both implementations. This figure clearly shows that the proposed attack
is outperforming CPA for both implementations.

The analysis of the leakage traces after the LDA transformation shows that de-
pending on the number of retained components, the SNR level can be significantly
higher compared to the original traces. The graph in Figure 4.6 shows the SNR lev-
els as the function of the number of projection directions retained after the trans-
formation. Since the increase in SNR together with the supervised classification
are the reasons for the better performance of the proposed attack method, it is
important to select a significantly large number of components. We have adapted
the heuristics of keeping the directions corresponding to the 95th percentile of the
eigenvalues after the eigendecomposition of $S_W^{-\frac{1}{2}}S_BS_W^{-\frac{1}{2}}$ [40].

4.3.2 Identity Leakage Model

To further extend our experiments, we have also investigated key extraction when
no assumptions about the leakage model are made. To this end, instead of classi-
fying leakage traces according to the HW of intermediate values, we separate them
according to some selected bits of the intermediate values. Due to intrinsic proper-
ties of the AES and DES encryption algorithms (in particular: the bijectivity of the
S-box), we will analyse them separately.
AES Encryption.

The intermediate values in this case were also predicted as $V_{j,P} = \text{Sbox}(j \oplus P)$. The classification of the leakage traces does not depend on the value of key candidate $j$ due to the bijectivity of the S-box function. In other words, let us define an event $T$ such that $T = 1$ if $V_{j,P_1} = V_{j,P_2}$ and $T = 0$ if $V_{j,P_1} \neq V_{j,P_2}$. Then, the MI between the subkey candidate $J$ and $P_1, P_2, T$ can be calculated according to the Equation (4.2) where $H$ is the Shannon entropy function.

$$I(J; P_1, P_2, T) = H[J] - H[J|P_1, P_2, T]$$

$$H[J] = \sum_{j=0}^{255} \text{Pr}[J = j] \log \frac{1}{\text{Pr}[J = j]} = \sum_{0}^{255} \frac{1}{256} \log 256 = 8$$

$$H[J|P_1, P_2, T] = H[J|P_1, P_2, T = 0] + H[J|P_1, P_2, T = 1]$$

$$= \frac{255}{256} \log 256 + \frac{1}{256} \log 256 = 8$$

$$I(J; P_1, P_2, T) = 8 - 8 = 0$$

The absence of MI leads to the conclusion that the classification based on the hypothetical intermediate values will be the same for each key candidate. Therefore, instead of assigning identical intermediate values, we assign similar intermediate values to the same class. In this context, we define similar intermediate values as the ones whose preselected $l$ bits are equal, where $l \in \{1 \ldots 7\}$.

DES Encryption.

The intermediate values were again chosen as $V_{j,P} = \text{Sbox}(j \oplus P)$. Similar calculations to Equation (4.2) show that the MI between the classification based on the intermediate values and the key candidate is larger than 0. Therefore, it is possible to select $l$ in the interval of $\{1 \ldots 4\}$.

As can be seen from the results plotted in Figures 4.7 and 4.8, the GSR for the Identity Model is lower than that of the HW model when the implementations are attacked with the proposed method. When compared to the results of CPA, it can

![Figure 4.6: The SNR before and after the LDA transformation](image-url)
be observed that depending on the number of selected bits and traces the new attack can be more successful in extracting the subkeys. The empirical study of the S-box functions of the encryption algorithms reveals that the MI between the key candidate and the classification increases with decreasing \( l \) (see Table 4.1), while the GSR does not follow the same pattern. When \( l \) gets smaller, the number of distinct intermediate values that are assigned to the same class increases, which leads to weaker separation among classes. Therefore, a compromise between getting maximum possible MI and keeping the classes well separable has to be made. Given that for fairly large amount of traces the performance of the attack is better than CPA even without making any assumptions about the leakage model, we can argue that the proposed attack is preferable.

![GSR for AES implementation when the hypothetical power consumption is predicted according to the Identity Model](image)

**Figure 4.7:** GSR for AES implementation when the hypothetical power consumption is predicted according to the Identity Model

<table>
<thead>
<tr>
<th>Mutual Information</th>
<th>AES</th>
<th>DES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S, S_1, S_2</td>
<td>S_3, S_4, S_5</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0.116</td>
</tr>
<tr>
<td>7</td>
<td>0.060</td>
<td>0.204</td>
</tr>
<tr>
<td>6</td>
<td>0.116</td>
<td>0.204</td>
</tr>
<tr>
<td>5</td>
<td>0.204</td>
<td>0.545</td>
</tr>
<tr>
<td>4</td>
<td>0.306</td>
<td>0.545</td>
</tr>
<tr>
<td>3</td>
<td>0.302</td>
<td>0.545</td>
</tr>
<tr>
<td>2</td>
<td>0.294</td>
<td>0.537</td>
</tr>
<tr>
<td>1</td>
<td>0.306</td>
<td>0.537</td>
</tr>
</tbody>
</table>

**Table 4.1:** The analysis of the MI between the key and the classification for AES and DES S-box outputs
4.3.3 Computational Complexity

While the success rates of different key extraction attacks may be high, their adaptation in real world scenarios is also bounded by the computational complexity. Since the side-channel security evaluations of cryptographic devices can involve millions of traces, it is desirable to be able to perform the analysis within the bounds of target time interval. The infeasibility of running the analysis using the proposed method on large number of traces will therefore render it practically worthless.

The analysis of the attack algorithm described in Algorithm 1 shows that the costly part is the transformation of the original leakage traces to the new subspace spanned by the eigenvectors of the ratio of scatter matrices. In particular, the calculation of the between-class and within-class scatter matrices have the complexity of $O(md^2)$ where $m$ is the number of leakage traces and $d$ is the number of samples. Similarly, the complexities of the operations in lines 2-4 are equal to $O(d^3)$. Since the number of traces is larger than the number of samples as described in Section 4.2, the complexity of the attack is $O(md^2)$. The linear relation between the computational complexity and the number of traces implies that the attack can indeed be carried out using large number of leakage traces if the number of samples per trace is kept small.
Algorithm 1: Pseudo-code of the attack

Input: Matrix of leakage traces: $Y$ \((m \times d)\)
Input: Vector of plaintexts: $P$ \((m \times 1)\)
Output: Vector of key candidate scores: $k$ \((|\mathcal{S}| \times 1)\)

1. $[S_W, S_B] = \text{scatter}(Y, P)$;
2. $T = S_W^{-\frac{1}{2}}$;
3. $M = TS_B T$;
4. $[U, \Delta] = \text{eig}(M)$;
5. $I = \text{sort}(\Delta)$;
6. $\tilde{U} = U(I)$;
7. $\tilde{W} = T \tilde{U}$;
8. $\hat{Y} = Y \tilde{W}$;
9. for $j \in \mathcal{S}$ do
10. $X_P = \text{model}(P, j)$;
11. $[\hat{S}_W, \hat{S}_B] = \text{scatter}(\hat{Y}, X_P)$;
12. $\hat{T} = \hat{S}_W^{-\frac{1}{2}}$;
13. $\hat{M} = \hat{T} \hat{S}_W \hat{T}$;
14. $[\hat{U}, \hat{\Delta}] = \text{eig}(\hat{M})$;
15. $k(j) = \max(\hat{\Delta})$;
16. end
Chapter 5

Final Words

Linear transformation methods like PCA and LDA have been applied in the domain of SCA, especially for conducting template analysis. PCA has also been used for key extraction purposes both as a pre-processor and a distinguisher. In this thesis we have conducted several experiments to compare and analyse the performance of the template attacks where the POI are chosen based on projection directions constructed by both of the techniques. We have also proposed a method for utilizing LDA to extract the key from the acquired side-channel leakages.

5.1 Conclusion

The comparison of the feature extraction methods has shown that the success rate of assigning a single trace to the pre-computed templates is higher for LDA. The difference between the performances gets significantly larger when the SNR of the traces is very low (i.e. below 3 dB).

The further look into the projections has revealed that due to its intrinsic properties PCA transformation fails to retain discriminatory information about the traces in the presence of high noise levels. As the objective of PCA in this context is to find a linear transformation that preserves the most of variance in the leading principal components based on the average traces corresponding to different groups, it does not take into account the distribution of the traces within each of them.

On the contrary, LDA finds the projection directions such that after transforming the traces into a low-dimensional space the leakages that belong to the same group are similar to each other while being significantly different from other groups. Such clear separation of the traces subsequently leads to better matching of new observations to the templates via maximum likelihood scores.

In addition to reducing the dimensionality and extracting discriminatory information, LDA transformation also achieves a near-optimal noise reduction. By making use of these properties, we have proposed to project the traces to a sub-space according to the directions determined without the knowledge of the encryption key. As already confirmed by the results, in this sub-space the maximum separation among the traces belonging to different groups is achieved.

Our last contribution in this work is the introduction of a distinguisher that ranks the key candidates based on the maximum eigenvalues obtained after the eigen-decomposition of between-class and within-class scatter ratios. We have shown that when the transformed traces are classified based on the hypothetical power
consumptions, the correct key candidate leads to the better separation among the classes, which subsequently leads to higher eigenvalue.

The method has been tested against noisy trace sets with and without making assumptions about the leakage model of the implementations. We have also discussed the theoretical restrictions arising from the application of the LDA transformation and proposed a method for achieving a higher GSR with lower number of traces. The experiments conducted on the software implementations of the AES and DES encryption have confirmed the efficiency of the proposed method. We have compared the new method to the CPA and have observed that significantly less number of traces were needed to achieve the same GSR.

5.2 Future Research

The advances in the side-channel analysis force the designers of cryptographic devices to implement countermeasures against known attacks. Random delays, hiding and masking are among the most used countermeasures in practice. We have developed and experimentally validated our method based on the unprotected implementations of cryptographic algorithms and have not systematically investigated the scenarios with protection. While we have succeeded to extract keys from some implementations with random time delays, no effort has been made to analyse the characteristics and performance of such attacks. Further research into the possible application of the method against the devices with countermeasures could reveal some interesting results.

It has been mentioned in Section 4.2 that in order to overcome the need for large amount of traces, they can be analysed a block at a time. Moreover, such approach also decreases the memory and power requirements since the computational complexity of the method is quadratic with respect to the number of samples. Further research could find out the relation between the number of traces, number of samples and the SNR for conducting a successful attack. The results of such research might set the theoretical limits of the proposed method.
Bibliography


[40] Li-Jen Weng and Chung-Ping Cheng. “Parallel Analysis with Unidimensional Binary Data”. In: *Educational and Psychological Measurement* 65.5 (2005), pp. 697–716.

Appendix A

Supplementary Material for Cryptographic Algorithms

A.1 Supplementary Material for AES

<table>
<thead>
<tr>
<th>$x_0$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$x_5$</th>
<th>$x_6$</th>
<th>$x_7$</th>
<th>$x_8$</th>
<th>$x_9$</th>
<th>$x_{a}$</th>
<th>$x_{b}$</th>
<th>$x_{c}$</th>
<th>$x_{d}$</th>
<th>$x_{e}$</th>
<th>$x_{f}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0y</td>
<td>63</td>
<td>7c</td>
<td>77</td>
<td>7b</td>
<td>f2</td>
<td>6b</td>
<td>6f</td>
<td>c5</td>
<td>30</td>
<td>01</td>
<td>67</td>
<td>2b</td>
<td>1e</td>
<td>d7</td>
<td>ab</td>
</tr>
<tr>
<td>1y</td>
<td>ca</td>
<td>82</td>
<td>c9</td>
<td>7d</td>
<td>fa</td>
<td>59</td>
<td>47</td>
<td>f0</td>
<td>ad</td>
<td>d4</td>
<td>a2</td>
<td>af</td>
<td>9c</td>
<td>a4</td>
<td>72</td>
</tr>
<tr>
<td>2y</td>
<td>b7</td>
<td>fd</td>
<td>93</td>
<td>26</td>
<td>36</td>
<td>3f</td>
<td>f7</td>
<td>cc</td>
<td>34</td>
<td>a5</td>
<td>e5</td>
<td>f1</td>
<td>7d</td>
<td>h8</td>
<td>31</td>
</tr>
<tr>
<td>3y</td>
<td>04</td>
<td>c7</td>
<td>23</td>
<td>c5</td>
<td>18</td>
<td>96</td>
<td>05</td>
<td>9a</td>
<td>07</td>
<td>12</td>
<td>80</td>
<td>e2</td>
<td>eb</td>
<td>27</td>
<td>b2</td>
</tr>
<tr>
<td>4y</td>
<td>09</td>
<td>83</td>
<td>2c</td>
<td>1a</td>
<td>1b</td>
<td>6e</td>
<td>5a</td>
<td>a0</td>
<td>52</td>
<td>3b</td>
<td>d6</td>
<td>b3</td>
<td>29</td>
<td>e3</td>
<td>2f</td>
</tr>
<tr>
<td>5y</td>
<td>53</td>
<td>d1</td>
<td>00</td>
<td>ed</td>
<td>20</td>
<td>fc</td>
<td>b1</td>
<td>5b</td>
<td>6a</td>
<td>cb</td>
<td>be</td>
<td>39</td>
<td>4a</td>
<td>4c</td>
<td>58</td>
</tr>
<tr>
<td>6y</td>
<td>d0</td>
<td>ef</td>
<td>aa</td>
<td>fb</td>
<td>43</td>
<td>4d</td>
<td>33</td>
<td>85</td>
<td>45</td>
<td>19</td>
<td>02</td>
<td>7f</td>
<td>50</td>
<td>3c</td>
<td>9f</td>
</tr>
<tr>
<td>7y</td>
<td>51</td>
<td>a3</td>
<td>40</td>
<td>8f</td>
<td>92</td>
<td>9d</td>
<td>38</td>
<td>f5</td>
<td>bc</td>
<td>b6</td>
<td>da</td>
<td>21</td>
<td>10</td>
<td>ff</td>
<td>f3</td>
</tr>
<tr>
<td>8y</td>
<td>cd</td>
<td>0c</td>
<td>13</td>
<td>ec</td>
<td>5f</td>
<td>97</td>
<td>44</td>
<td>17</td>
<td>c4</td>
<td>a7</td>
<td>7e</td>
<td>3d</td>
<td>64</td>
<td>5d</td>
<td>19</td>
</tr>
<tr>
<td>9y</td>
<td>60</td>
<td>81</td>
<td>4f</td>
<td>dc</td>
<td>22</td>
<td>2a</td>
<td>90</td>
<td>88</td>
<td>46</td>
<td>ee</td>
<td>b8</td>
<td>14</td>
<td>de</td>
<td>5e</td>
<td>0b</td>
</tr>
<tr>
<td>ay</td>
<td>e0</td>
<td>32</td>
<td>3a</td>
<td>0a</td>
<td>49</td>
<td>06</td>
<td>24</td>
<td>5c</td>
<td>c2</td>
<td>d3</td>
<td>ac</td>
<td>62</td>
<td>91</td>
<td>95</td>
<td>e4</td>
</tr>
<tr>
<td>by</td>
<td>e7</td>
<td>c8</td>
<td>37</td>
<td>6d</td>
<td>8d</td>
<td>d5</td>
<td>4e</td>
<td>a9</td>
<td>6c</td>
<td>56</td>
<td>f4</td>
<td>ea</td>
<td>65</td>
<td>7a</td>
<td>ae</td>
</tr>
<tr>
<td>cy</td>
<td>ba</td>
<td>78</td>
<td>25</td>
<td>2e</td>
<td>1c</td>
<td>a6</td>
<td>b4</td>
<td>c6</td>
<td>e8</td>
<td>dd</td>
<td>74</td>
<td>1f</td>
<td>4b</td>
<td>bd</td>
<td>8b</td>
</tr>
<tr>
<td>dy</td>
<td>70</td>
<td>3e</td>
<td>b5</td>
<td>66</td>
<td>48</td>
<td>03</td>
<td>f6</td>
<td>ce</td>
<td>61</td>
<td>35</td>
<td>57</td>
<td>b9</td>
<td>86</td>
<td>c1</td>
<td>1d</td>
</tr>
<tr>
<td>ey</td>
<td>e1</td>
<td>f8</td>
<td>98</td>
<td>11</td>
<td>69</td>
<td>d9</td>
<td>8e</td>
<td>94</td>
<td>9b</td>
<td>94</td>
<td>1e</td>
<td>87</td>
<td>e9</td>
<td>cc</td>
<td>55</td>
</tr>
<tr>
<td>fy</td>
<td>8c</td>
<td>a1</td>
<td>89</td>
<td>0d</td>
<td>bf</td>
<td>e6</td>
<td>42</td>
<td>68</td>
<td>41</td>
<td>99</td>
<td>2d</td>
<td>0f</td>
<td>b0</td>
<td>54</td>
<td>bb</td>
</tr>
</tbody>
</table>

**Table A.1:** Hexadecimal notation of the AES S-box function
A.2 Supplementary Material for DES

A.2.1 Permutation Tables

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>58</td>
<td>50</td>
<td>42</td>
<td>34</td>
<td>26</td>
<td>18</td>
<td>10</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>52</td>
<td>44</td>
<td>36</td>
<td>28</td>
<td>20</td>
<td>12</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>54</td>
<td>46</td>
<td>38</td>
<td>30</td>
<td>22</td>
<td>14</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>56</td>
<td>48</td>
<td>40</td>
<td>32</td>
<td>24</td>
<td>16</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>49</td>
<td>41</td>
<td>33</td>
<td>25</td>
<td>17</td>
<td>9</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>51</td>
<td>43</td>
<td>35</td>
<td>27</td>
<td>19</td>
<td>11</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>53</td>
<td>45</td>
<td>37</td>
<td>29</td>
<td>21</td>
<td>13</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>55</td>
<td>47</td>
<td>39</td>
<td>31</td>
<td>23</td>
<td>15</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE A.2: Initial Permutation**

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>20</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>29</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
<td>32</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE A.3: Expansion Permutation**

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>7</td>
<td>20</td>
<td>21</td>
<td>29</td>
<td>12</td>
<td>28</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>15</td>
<td>23</td>
<td>26</td>
<td>5</td>
<td>18</td>
<td>31</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>24</td>
<td>14</td>
<td>32</td>
<td>27</td>
<td>3</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>13</td>
<td>30</td>
<td>6</td>
<td>22</td>
<td>11</td>
<td>4</td>
<td>25</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE A.4: P-box**

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>8</td>
<td>48</td>
<td>16</td>
<td>56</td>
<td>24</td>
<td>64</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>7</td>
<td>47</td>
<td>15</td>
<td>55</td>
<td>23</td>
<td>63</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>6</td>
<td>46</td>
<td>14</td>
<td>54</td>
<td>22</td>
<td>62</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>5</td>
<td>45</td>
<td>13</td>
<td>53</td>
<td>21</td>
<td>61</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>4</td>
<td>44</td>
<td>12</td>
<td>52</td>
<td>20</td>
<td>60</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>3</td>
<td>43</td>
<td>11</td>
<td>51</td>
<td>19</td>
<td>59</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>2</td>
<td>42</td>
<td>10</td>
<td>50</td>
<td>18</td>
<td>58</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>1</td>
<td>41</td>
<td>9</td>
<td>49</td>
<td>17</td>
<td>57</td>
<td>25</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE A.5: Final Permutation**

---

1All the permutation tables entries indicate the position of the input bit that is mapped to the corresponding output bit. For example, the 1st bit of the output corresponds to the 58th bit of the input, the 2nd bit corresponds to the 50th bit, 9th bit to the 60th bit, etc.
A.2.2  S-Box Functions

<table>
<thead>
<tr>
<th>S1</th>
<th>0000</th>
<th>0001</th>
<th>0100</th>
<th>0101</th>
<th>1000</th>
<th>1001</th>
<th>1010</th>
<th>1011</th>
<th>1100</th>
<th>1101</th>
<th>1110</th>
<th>1111</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>14</td>
<td>4</td>
<td>13</td>
<td>1</td>
<td>2</td>
<td>13</td>
<td>11</td>
<td>8</td>
<td>5</td>
<td>10</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>01</td>
<td>4</td>
<td>1</td>
<td>14</td>
<td>8</td>
<td>13</td>
<td>6</td>
<td>2</td>
<td>11</td>
<td>15</td>
<td>12</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>12</td>
<td>8</td>
<td>2</td>
<td>4</td>
<td>9</td>
<td>1</td>
<td>7</td>
<td>5</td>
<td>11</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>10</td>
<td>13</td>
<td>1</td>
<td>3</td>
<td>15</td>
<td>4</td>
<td>2</td>
<td>11</td>
<td>6</td>
<td>7</td>
<td>12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S2</th>
<th>0000</th>
<th>0001</th>
<th>0100</th>
<th>0101</th>
<th>1000</th>
<th>1001</th>
<th>1010</th>
<th>1011</th>
<th>1100</th>
<th>1101</th>
<th>1110</th>
<th>1111</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>15</td>
<td>1</td>
<td>8</td>
<td>14</td>
<td>6</td>
<td>11</td>
<td>3</td>
<td>4</td>
<td>9</td>
<td>7</td>
<td>2</td>
<td>13</td>
</tr>
<tr>
<td>01</td>
<td>3</td>
<td>13</td>
<td>4</td>
<td>7</td>
<td>15</td>
<td>2</td>
<td>8</td>
<td>14</td>
<td>12</td>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>14</td>
<td>7</td>
<td>17</td>
<td>10</td>
<td>4</td>
<td>13</td>
<td>1</td>
<td>5</td>
<td>8</td>
<td>12</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>13</td>
<td>8</td>
<td>10</td>
<td>13</td>
<td>3</td>
<td>15</td>
<td>4</td>
<td>2</td>
<td>11</td>
<td>6</td>
<td>7</td>
<td>12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S3</th>
<th>0000</th>
<th>0001</th>
<th>0100</th>
<th>0101</th>
<th>1000</th>
<th>1001</th>
<th>1010</th>
<th>1011</th>
<th>1100</th>
<th>1101</th>
<th>1110</th>
<th>1111</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>10</td>
<td>0</td>
<td>9</td>
<td>14</td>
<td>6</td>
<td>3</td>
<td>15</td>
<td>5</td>
<td>1</td>
<td>13</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>01</td>
<td>13</td>
<td>7</td>
<td>0</td>
<td>9</td>
<td>3</td>
<td>4</td>
<td>6</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>5</td>
<td>14</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>13</td>
<td>6</td>
<td>4</td>
<td>9</td>
<td>8</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>11</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>10</td>
<td>13</td>
<td>0</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>4</td>
<td>15</td>
<td>14</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S4</th>
<th>0000</th>
<th>0001</th>
<th>0100</th>
<th>0101</th>
<th>1000</th>
<th>1001</th>
<th>1010</th>
<th>1011</th>
<th>1100</th>
<th>1101</th>
<th>1110</th>
<th>1111</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>7</td>
<td>13</td>
<td>14</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td>9</td>
<td>10</td>
<td>1</td>
<td>2</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>01</td>
<td>13</td>
<td>8</td>
<td>11</td>
<td>5</td>
<td>6</td>
<td>15</td>
<td>0</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>10</td>
<td>6</td>
<td>9</td>
<td>0</td>
<td>12</td>
<td>11</td>
<td>7</td>
<td>13</td>
<td>15</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>15</td>
<td>0</td>
<td>6</td>
<td>10</td>
<td>1</td>
<td>13</td>
<td>8</td>
<td>9</td>
<td>4</td>
<td>5</td>
<td>11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S5</th>
<th>0000</th>
<th>0001</th>
<th>0100</th>
<th>0101</th>
<th>1000</th>
<th>1001</th>
<th>1010</th>
<th>1011</th>
<th>1100</th>
<th>1101</th>
<th>1110</th>
<th>1111</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>2</td>
<td>12</td>
<td>4</td>
<td>1</td>
<td>7</td>
<td>10</td>
<td>11</td>
<td>6</td>
<td>8</td>
<td>5</td>
<td>3</td>
<td>15</td>
</tr>
<tr>
<td>01</td>
<td>14</td>
<td>11</td>
<td>2</td>
<td>12</td>
<td>4</td>
<td>7</td>
<td>13</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>15</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>15</td>
<td>1</td>
<td>11</td>
<td>10</td>
<td>2</td>
<td>13</td>
<td>7</td>
<td>8</td>
<td>15</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>12</td>
<td>7</td>
<td>1</td>
<td>14</td>
<td>2</td>
<td>13</td>
<td>6</td>
<td>15</td>
<td>9</td>
<td>10</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S6</th>
<th>0000</th>
<th>0001</th>
<th>0100</th>
<th>0101</th>
<th>1000</th>
<th>1001</th>
<th>1010</th>
<th>1011</th>
<th>1100</th>
<th>1101</th>
<th>1110</th>
<th>1111</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>12</td>
<td>1</td>
<td>10</td>
<td>15</td>
<td>9</td>
<td>2</td>
<td>6</td>
<td>8</td>
<td>0</td>
<td>13</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>01</td>
<td>10</td>
<td>15</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>12</td>
<td>9</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>13</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>14</td>
<td>15</td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>12</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>12</td>
<td>9</td>
<td>5</td>
<td>15</td>
<td>10</td>
<td>11</td>
<td>14</td>
<td>1</td>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S7</th>
<th>0000</th>
<th>0001</th>
<th>0100</th>
<th>0101</th>
<th>1000</th>
<th>1001</th>
<th>1010</th>
<th>1011</th>
<th>1100</th>
<th>1101</th>
<th>1110</th>
<th>1111</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>4</td>
<td>11</td>
<td>2</td>
<td>14</td>
<td>15</td>
<td>0</td>
<td>8</td>
<td>13</td>
<td>3</td>
<td>12</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>01</td>
<td>13</td>
<td>0</td>
<td>11</td>
<td>7</td>
<td>4</td>
<td>9</td>
<td>1</td>
<td>10</td>
<td>14</td>
<td>3</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>4</td>
<td>11</td>
<td>13</td>
<td>12</td>
<td>3</td>
<td>7</td>
<td>14</td>
<td>10</td>
<td>15</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>6</td>
<td>11</td>
<td>13</td>
<td>12</td>
<td>3</td>
<td>7</td>
<td>14</td>
<td>10</td>
<td>15</td>
<td>6</td>
<td>8</td>
<td>12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S8</th>
<th>0000</th>
<th>0001</th>
<th>0100</th>
<th>0101</th>
<th>1000</th>
<th>1001</th>
<th>1010</th>
<th>1011</th>
<th>1100</th>
<th>1101</th>
<th>1110</th>
<th>1111</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>13</td>
<td>2</td>
<td>8</td>
<td>4</td>
<td>6</td>
<td>15</td>
<td>11</td>
<td>1</td>
<td>10</td>
<td>9</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
<td>15</td>
<td>13</td>
<td>8</td>
<td>10</td>
<td>3</td>
<td>7</td>
<td>4</td>
<td>12</td>
<td>5</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>10</td>
<td>7</td>
<td>11</td>
<td>4</td>
<td>1</td>
<td>9</td>
<td>12</td>
<td>14</td>
<td>2</td>
<td>0</td>
<td>6</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>1</td>
<td>14</td>
<td>7</td>
<td>4</td>
<td>10</td>
<td>8</td>
<td>13</td>
<td>15</td>
<td>12</td>
<td>9</td>
<td>0</td>
</tr>
</tbody>
</table>

Table A.6: DES S-Box functions\(^2\)

\(^2\)The output of the S-box function is determined by table entries. The first and the last bits of the input are concatenated to get the row, the remaining bits are used to get the column of the output value. For example, the input 101010 to the first S-box outputs the value in the row 10 and the column 1010, which is 6.