Security Arguments and Tool-based Design of Block Ciphers

Security Arguments and Tool-based Design of Block Ciphers

Dissertation Thesis

Friedrich Wiemer

16th August 2019

Submitted in partial fulfillment of the requirements
for the degree of Doktor der Naturwissenschaften

to the

Faculty of Mathematics
at Ruhr-Universität Bochum

1st Reviewer	Prof. Dr. Gregor Leander
2nd Reviewer	Prof. Dr. Alexander May

IMPRINT

Security Arguments and Tool-based Design of Block Ciphers
Copyright © 2019 by Friedrich Wiemer.
All rights reserved. Printed in Germany.
Published by the Ruhr-Universität Bochum, Bochum, Germany.

COLOPHON

This thesis was typeset using LaTeX and the `memoir` documentclass. It is based on Aaron Turon's thesis *Understanding and expressing scalable concurrency*[1], itself a mixture of `classicthesis`[2] by André Miede and `tufte-latex`[3], based on Edward Tufte's *Beautiful Evidence*.

The bibliography was processed by Biblatex. All graphics and plots are made with PGF/Ti*k*Z.

The body text is set 10/14pt (long primer) on a 26pc measure. The margin text is set 8/9pt (brevier) on a 12pc measure. Matthew Carter's Charter acts as both the text and display typeface. Monospaced text uses Jim Lyles's `Bitstream Vera Mono` ("Bera Mono").

[1] `https://people.mpi-sws.org/~turon/turon-thesis.pdf`

[2] `https://bitbucket.org/amiede/classicthesis/`

[3] `https://github.com/Tufte-LaTeX/tufte-latex`

If we knew what it was we were doing,
it would not be called research, would it?
—Albert Einstein

Abstract

Block ciphers form, without doubt, the backbone of today's encrypted communication and are thus justifiably the workhorses of cryptography. While efficiency of modern designs improved ever since the development of the DES and AES, the case with the corresponding security arguments differs. The thesis at hand aims at two main points, both in the direction of improving security analysis of block ciphers.

Part I studies a new notion for the better understanding of a special type of cryptanalysis and proposes a new block cipher instance. This instance comes with a tight bound on any differential, to the best of our knowledge the first such block cipher.

Part II turns to automated methods in design and analysis of block ciphers. Our main contribution here is an algorithm to propagate subspaces through encryption rounds, together with two applications: an algorithmic security argument against a new type of cryptanalysis and an idea towards the automation of key recovery attacks.

Zusammenfassung

Blockchiffren bilden ohne Zweifel das Rückgrat unserer heutigen digitalen Kommunikation und werden somit zu Recht als Arbeitstier der Kryptographie bezeichnet. Während die Effizienz neuer Chiffren stetig steigt, gilt dies nur bedingt für deren Sicherheitsargumente. Die vorliegende Arbeit beschäftigt sich daher mit zwei Hauptthemen.

In Teil I untersuchen wir eine neue Notation einer speziellen Kryptanalyse-Technik und geben neue theoretische Einsichten zu dieser. Außerdem konstruieren wir eine Blockchiffre und zeigen scharfe Schranken für jedes Differential; nach bestem Wissen die erste solche Chiffre.

In Teil II wenden wir uns algorithmischen Methoden für Design und Analyse von Blockchiffren zu. Der Hauptbeitrag ist ein Algorithmus um Unterräume durch Chiffren-Runden zu propagieren. Abschließend diskutieren wir zwei Anwendungen: ein Sicherheitsargument für eine neue Kryptanalyse-Technik und einen Ansatz zur Automatisierung von Key Recovery-Angriffen.

Contents

List of Algorithms

List of Figures

List of Tables

Part I

PROLOGUE

1

Introduction

"Nanos gigantum humeris insidentes."
—Bernard of Chartres

"If I have seen further, it is by standing on the shoulders of giants."
—Isaac Newton

Cryptography is by now used for more than 30 centuries. While in ancient times encrypting messages seemed more like an art, our understanding of the security needs and techniques greatly evolved during the last decades. Likewise, the field of cryptology has greatly grown and research specialised in many different aspects, mainly dividable in the underlying techniques of symmetric and asymmetric cryptography. The prime difference between these two is that for symmetric ciphers both encryption and decryption needs the same secret key, while in asymmetric ciphers the so-called public key is used for encryption (and does not have to be kept secret) and only the secret key allows decryption.

Especially such fascinating recent developments of techniques like secure multiparty computation or fully homomorphic encryption have the chance to significantly impact our privacy needs of today's digital world. While these techniques mainly rely on asymmetric cryptography, symmetric cryptography is still the technique of choice when it comes to plain old encryption of messages.

► BLOCK CIPHERS are today's workhorse of symmetric encryption – and thus responsible for the majority of all encrypted data. Opposed to public key primitives, which security properties can often be reduced to cryptographic or complexity theoretic hardness assumptions and thus often come with a provable security notion, our trust in symmetric primitives is mainly built from extensive cryptanalysis. Informally, a secure block cipher is thus one that withstood cryptanalysis long enough and resists all known attacks; from a more theoretical point of view we require a secure block cipher to be interchangeably usable for a random permutation, i. e. the block cipher should not be distinguishable from a family of random permutations.

For a more precise and technical introduction to block ciphers and their analysis see the following Chapter 2, in particular Sections 2.2 and 2.3.

To ease this analysis, simplifying the concrete design of a primitive is a fruitful ansatz. A very common approach is to design the cipher as an iteration of a simple, in itself insecure, round function. The overall security of the block cipher is then built piecewise over several rounds of encryption. To further ease the analysis of the round function, it is often built independently of the encryption (round) keys. The key-material is instead introduced in between the rounds, often by XOR-ing it into the cipher's state. Such constructions are known as key-alternating ciphers, whereof the best known instance is the Advanced Encryption Standart (AES).[1] Compared to "older" block cipher designs, such as that of the Data Encryption Standart (DES) or BLOWFISH,[2] this kind of, very simple and elegant, design allows us to throughly understand its properties and build confidence in its security. After 20 years of cryptanalysis, we can justifiably say that the AES is today's best understood design and (even more important) remains a *secure* block cipher.

Albeit one might think the existence of standards like the AES renders the design of new block ciphers needless, plenty of new designs where proposed since the publication of the AES.

▸ INCENTIVES FOR NEW CIPHER DESIGNS

It is a legitimate concern to ask why we still keep designing new ciphers all the time. *Lightweight cryptography* is the latest most important trend in symmetric cryptography, at least since the publication of the block cipher PRESENT,[3] with its current peak in NIST's (the National Institute of Standards and Technology) Lightweight Crypto (LWC) competition. The main reason for lightweight cryptographic designs is the assumption that existing standards, e. g. the AES, are too costly for heavily constrained devices such as medical implants, RFID tags, or sensors for the Internet-of-Things (IoT). Another interesting direction is the design of special purpose ciphers,[4] i. e. ciphers which are built for the use in one specific application and are thus specifically tailored to the corresponding requirements.

An obvious advantage is the gain in efficiency for such specialised designs.[5] The main advantage is actually a different one. Vigorously optimising block cipher designs to be as efficient as possible leads to designs with sometimes very narrow security margins – or them even being broken. Such designs at the border of security sometimes enable weaknesses to occur, for which "normal" designs are just "too secure". A prime example is the PRINTCIPHER which was later broken by structural attacks with the discovery of invariant subspace attacks. However, these designs provide us with the valuable insight of what is important for the security of a block cipher.

Designing new block ciphers thus enables us to broaden our understanding of which structures provide what kind of security. Eventually this process of designing and analysing new block ciphers will lead to very few standardised encryption algorithms that are then recommended to use in real world applications.

1.1 OPEN PROBLEMS IN BLOCK CIPHER DESIGN

Efficiency of symmetric ciphers have been significantly improved further and further over the past years, in particular within the trend of lightweight cryptography. However, when it comes to arguing about the security of ciphers, the progress is rather limited and the arguments basically did not get easier nor stronger since the development of the AES. It might thus be worth shifting the focus to improving security arguments for new designs rather than (incrementally) their efficiency.

Maybe the top goal of cryptanalysis should be to fully understand and thoroughly exploit every cryptanalytic attack that we can mount on block ciphers.

[1] Some people call it the "king of bock ciphers".

[2] DES' Feistel structure is arguable more complex than that of comparable modern Feistel ciphers as e. g. SM4, the Chinese symmetric encryption standard. BLOWFISH uses key-dependent S-boxes, which make it considerable harder to reason about its precise resistance against known attacks.

[3] Since its publication in 2007 and at the time of writing, the most cited cryptographic paper, according to Konrad Rieck [Rie] (visited 09–06–2019).

[4] Two examples are LOWMC, see Albrecht et al., "Ciphers for MPC and FHE", and RASTA, see Dobraunig et al., "RASTA: A cipher with low ANDdepth and few ANDs per bit".

[5] However it might often be that the "AES is lightweight enough", see Aumasson et al. [Aum] (visited 09–06–2019).

► Full Understanding of Cryptanalytic Attacks

Being a noble goal, the often computational infeasible kind of cryptanalytical attacks on cryptographic designs renders it seemingly impossible to achieve.

In particular even with so well-studied attacks as differential and linear cryptanalysis, effects (differentials and linear hulls) occur which we do not fully understand, even for ciphers like the AES. Giving tight security bounds in such a case is inherently difficult. Thus, two problems, exemplary for differential and linear cryptanalysis, are

See Section 2.3.2 for a detailed explanation of differential cryptanalysis and the problems that appear when trying to bound the differential probability.
Section 2.3.3 discusses linear cryptanalysis and the respective problems with the linear hull.

Problem 1 (Differentials)**.** *Given a cipher E, compute a (tight) bound on the probability that a differential through E holds.*

and

Problem 2 (Linear Hulls)**.** *Given a cipher E, compute a (tight) bound on the correlation for a linear hull to approximate E.*

► Security Arguments for Block Ciphers

Two very distinct flavours of security arguments for cipher designs are provable security arguments (asymptotic and concrete security) on the one hand, and resistance to practical attacks on the other hand. For example, the generic (or idealised) security of key-alternating ciphers was studied in recent years and one can prove that for breaking such an r-round construction one has to make roughly $2^{\frac{r}{r+1}n}$ oracle queries.

While this bound can be proven to be tight for key-alternating ciphers, it is unsatisfactory for two reasons. First, one might hope to get better security bounds with different constructions and second one might hope to lower the requirements used in the security proof. A similar goal to the above is thus to find new generic constructions that allow very strong security proofs.

Problem 3 (Block Cipher Constructions)**.** *Find new secure constructions for block ciphers.*

We recall the most common constructions for block ciphers in Section 2.2.2 and discuss results on their security in Section 2.2.3.

For the more practical part of cryptanalysis, and apart from the first two problems, it might also well be the case that new attacks are discovered. In such a case, all previously published ciphers that remained secure have to be evaluated against this new attack. To ease this process, easily verifiable arguments for the resistance against a new technique are very helpful.

Problem 4 (Provable Resistance)**.** *Given an attack for block ciphers, find provable properties to show the resistance against this attack.*

The common heuristic argument for resistance against differential and linear cryptanalysis is discussed in Sections 2.3.2 to 2.3.4.

► Automatisation of Design and Cryptanalysis

Having easy to check theorems for the resistance of a cipher against some type of attack is already a good basis. It might still be connected with a huge amount of (tedious) work, to check the security of many block ciphers. For example, in the first round of the LWC competition, 56 candidate designs were accepted (out of 57 submissions). In case we found a new attack for block ciphers, we would have to check each of these candidates if and how well this attack is applicable.

It is thus very helpful if tedious and repetitive analysis tasks can be automated. This does not only save time we can spend on more productive research, but also reduces the probability of making errors during the analysis. So, similar to Problem 4:

Problem 5 (Algorithmic Resistance)**.** *Given an attack for block ciphers, find an algorithmic way to show the resistance against this attack.*

Attacks on symmetric ciphers are often based on two parts, see Section 2.3 for more details. First, the cryptanalyst finds some non random behaviour that can be turned into a distinguisher for some rounds of the cipher. For a lot of attacks there are methods that find such distinguishers automatically. Second, this distinguisher is extended by a key guessing part to turn the distinguishing attack into a key recovery attack, where the key guesses are verified by the distinguisher. This second part involves a lot of manual bit-fiddling and precise details of the cipher under scrutiny – and is thus prone to errors.

Problem 6 (Automate Key Recovery)**.** *Given a distinguisher D for r rounds of a cipher E, algorithmically turn this distinguisher into a key recovery attack over a maximal number of $r + s$ rounds for E.*

But not only during the analysis of cryptographic primitives, also when initially designing them, automated methods can be of great help. For example when specific parameters have to be optimised, a computer-aided process is virtually indispensable for the exploration of the design space.

Problem 7 (Optimise Building Block)**.** *Given a set of constraints for a building block and some cost metric, find an optimal instance for this building block regarding that metric.*

In the next section, we first briefly sketch the outline of the thesis and then summarise the contributions in the context of the above problems.

1.2 OUTLINE AND CONTRIBUTIONS

The remainder of the thesis is structured as follows. In the following Chapter 2 we discuss preliminary notions, constructions, and analysis techniques for block ciphers. The technical parts of the thesis are then split in two.

In SECURITY ARGUMENTS FOR BLOCK CIPHERS, we discuss an analysis tool[6] published at EUROCRYPT'19 and its connection to established notions for (vectorial) Boolean functions, see Chapter 3 and [Can+19b]. The following Chapter 4 describes our instances BISON and WISENT of the WSN construction; it is based on [Can+19a].

The next part, AUTOMATED METHODS IN DESIGN AND ANALYSIS, is split in three Chapters 5 to 7. We first discuss heuristic methods for optimising implementations of matrix multiplications in hardware, see also [Kra+17b]. Then we turn our attention to cryptanalysis and study an algorithmic way to propagate truncated differentials and subspace trails through Substitution Permutation Network (SPN) constructions. For the discussed algorithm, we propose two applications: bounding the length of any subspace trails, and

[6] The Differential-Linear Connectivity Table by Bar-On et al.

[Can+19b] Canteaut et al., *Observations on the DLCT and Absolute Indicators*

[Can+19a] Canteaut et al., "BISON – Instantiating the Whitened Swap-Or-Not Construction"

[Kra+17b] Kranz et al., "Shorter Linear Straight-Line Programs for MDS Matrices" approaching the automatisation of key recovery attacks (the first is based on [Lea+18]).

[Lea+18] Leander et al., "Searching for Subspace Trails and Truncated Differentials"

Finally, Chapter 8 concludes the thesis.

In particular, the respective contributions are the following.

1.2.1 *Security Arguments for Block Ciphers and Understanding of Attacks*

This part consists of contributions to the understanding of differential-linear cryptanalysis and our instances of the WSN construction.

► Autocorrelation Tables and Differential-Linear Cryptanalysis

Bar-On et al. [Bar+19] proposed a new analysis tool for differential-linear cryptanalysis which allows them to give a more precise estimation of several differential-linear distinguishers published so far. Following the same pattern as similar tools for differential, linear and boomerang attacks, they coin it Differential-Linear Connectivity Table (DLCT). However, the authors did not analyse the DLCT from a theoretical point in detail.

[Bar+19] Bar-On et al. "DLCT: A New Tool for Differential-Linear Cryptanalysis"

We continue the analysis at this point and provide further insights on the DLCT. In particular, we show that this table coincides with the already known Autocorrelation Table (ACT) – highlighting an interesting new connection between resistance to differential-linear attacks and the autocorrelation of vectorial Boolean functions. Additionally, we discuss further properties of the autocorrelation in the case of vectorial Boolean functions. Our analysis contributes to the understanding of resisting differential-linear attacks, see Problem 4 (Provable Resistance).

► Bison – Instantiating the WSN construction

The Whitened Swap-Or-Not (WSN) construction by Tessaro [Tes15] is an alternative construction for block ciphers. By proving the security of his construction, Tessaro provides a solution to Problem 3 (Block Cipher Constructions). In this regard, his work just comes with one drawback: there is no instance of this construction yet. But when building such an instance, one fundamentally has to instantiate the building blocks modeled as ideal functionalities. Furthermore, such a practical instance has to be scrutinised from a symmetric cryptanalysis perspective and has to provide security against known attacks. Thus, without an instance and further analysis, the underlying constructing remains of no avail.

[Tes15] Tessaro "Optimally Secure Block Ciphers from Ideal Primitives"

In [Can+19a], we close this gap and suggest an instance of the WSN construction, completing Tessaro's solution to Problem 3. For this instance, we first derive some (rather trivial but nevertheless important) design rationales. Based on these properties every secure instance has to fulfill, we conduct further cryptanalysis. As a result of this analysis, we give a tight bound for any non-trivial differential through such an instance, thus solving Problem 1 (Differentials) for this case. Note that, to the best of our knowledge, no other secure cipher is known, for which we are able to solve this problem.

In the remainder of the chapter, we then define our instance more precisely – that is, we give a concrete specification for an instance fulfilling our generic restrictions. We call this instance Bison[7], as it is based on bent Boolean functions. Afterwards, we extend the cryptanalysis of Bison to

[7] Bison is a Bent whItened Swap-Or-Not cipher.

other attacks and discuss performance figures. The main drawback from an implementers perspective of BISON (apart from being impractically slow) is that we can only instantiate odd block length versions of it. For an actual use in hard- or software, we would however strongly prefer even block lengths.

[Can+19a] Canteaut et al., "BISON – Instantiating the Whitened Swap-Or-Not Construction"

We thus extend the work from [Can+19a] and give a second instance, for even block lengths, of the WSN construction, which follows our initial analysis. Due to its close connection to BISON, we name this instance WISENT. Here, too, we extend the analysis to other common attacks and give performance figures.

1.2.2 *Automated Methods in Design and Analysis*

This part consists of contributions to the optimisation of XOR counts, algorithmic security arguments for subspace trails and an automatisation approach for key recovery strategies.

▸ HEURISTICS FOR XOR COUNTS

Recently, finding optimal linear layers in block ciphers, an instance of Problem 7 (Optimise Building Block), attracted a lot of interest. Here, the building block we are looking for is a matrix M, where the multiplication by M corresponds to the application of the linear layer. Typically, this matrix should exhibit some special properties, e. g. being Maximum Distance Separable (MDS), due to security reasons. The metric under which we want to find an optimal matrix is then the number of gates (hardware) or instructions (software) needed in an implementation.

As the design space usually is too large for exhaustive search, we have to apply other methods. For example, proofs for the minimal costs of multiplications by single elements were given by some authors, e. g. Beierle et al. [Bei+16a] and Kölsch [Köl19]; a typical optimisation strategy for a complete matrix would then be to choose the matrix elements from the cheapest elements possible. This can be seen as a local optimisation strategy.

[Bei+16a] Beierle et al. "Lightweight Multiplication in GF(2^n) with Applications to MDS Matrices"

[Köl19] Kölsch "XOR-Counts and Lightweight Multiplication with Fixed Elements in Binary Finite Fields"

[Kra+17b] Kranz et al., "Shorter Linear Straight-Line Programs for MDS Matrices"

In [Kra+17b], we observe that optimising the matrix implementation globally, that is reusing intermediate results from one element multiplication for another, can save a lot of resources. Further we observe that this problem of finding an optimal implementation is known as finding the shortest linear straight line program for this implementation. Moreover, a whole branch of research exists on solving this problem. As finding short linear straight line programs is NP-hard, several heuristic approaches were proposed in the literature. In the remainder of this chapter, we discuss and evaluate the effectiveness of these heuristics. Eventually we are able to improve all known current best implementations with little effort.

While this contributes towards solving Problem 7 for this case, it seems rather impossible to ultimately solve this problem, as we discuss at the end of this chapter.

▸ PROPAGATING TRUNCATED DIFFERENTIALS

While standard differential cryptanalysis is quite well understood, this is not the same case for its variants. For example when it comes to truncated differentials, no generic security argument for the resistance against this attack is known. Additionally, the new notion of subspace trails, see [Gra+16], which is related to truncated differentials, allowed to discover new properties of the AES, e. g. [Gra+17; Bar+18] (after 20 years of cryptanalysis). It is thus important to improve our understanding of these attacks.

[Gra+16] Grassi et al., "Subspace Trail Cryptanalysis and its Applications to AES"

In [Lea+18] we contribute towards this goal. We first show the exact relation between subspace trails and truncated differentials for their probability-one versions. Afterwards, we develop an algorithm to propagate a given subspace trail through an SPN round function. We further give properties and algorithmic ways to show an (SPN) cipher's resistance, thus partly solving Problems 4 and 5 for this attack. Finally we apply our algorithms to a list of SPN ciphers and give precise bounds on the longest subspace trails for each of them.

[Lea+18] Leander et al., "Searching for Subspace Trails and Truncated Differentials"

- Key Recovery

As a second application of our propagation algorithm for truncated differentials, we turn our focus to Problem 6 (Automate Key Recovery). We observe that extending a differential distinguisher is closely related to subspace trails and truncated differentials of probability one. Based on this finding, we discuss ideas how to exploit this relation in order to automate the construction of key recovery attacks.

1.3 PUBLICATIONS

During the course of my doctoral studies, I worked on several projects which are not all covered in the remainder of this thesis. In particular, these are the following.

- Conference Publications

  Anne Canteaut, Virginie Lallemand, Gregor Leander, Patrick Neumann, and Friedrich Wiemer. "Bison – Instantiating the Whitened Swap-Or-Not Construction". In: *EUROCRYPT 2019, Part III*. ed. by Yuval Ishai and Vincent Rijmen. Vol. 11478. LNCS. Springer, Heidelberg, May 2019, pp. 585–616. DOI: 10.1007/978-3-030-17659-4_20. IACR PREPRINT: 2018/1011, see Chapter 4.

- Journal Publications

  Thorsten Kranz, Gregor Leander, and Friedrich Wiemer. "Linear Cryptanalysis: Key Schedules and Tweakable Block Ciphers". In: *IACR Trans. Symm. Cryptol.* 2017.1 (2017), pp. 474–505. ISSN: 2519-173X. DOI: 10.13154/tosc.v2017.i1.474-505. IACR PREPRINT: 2017/154.

  Thorsten Kranz, Gregor Leander, Ko Stoffelen, and Friedrich Wiemer. "Shorter Linear Straight-Line Programs for MDS Matrices". In: *IACR Trans. Symm. Cryptol.* 2017.4 (2017), pp. 188–211. ISSN: 2519-173X. DOI: 10.13154/tosc.v2017.i4.188-211. IACR PREPRINT: 2017/1151, see Chapter 5.

Gregor Leander, Cihangir Tezcan, and Friedrich Wiemer. "Searching for Subspace Trails and Truncated Differentials". In: *IACR Trans. Symm. Cryptol.* 2018.1 (2018), pp. 74–100. ISSN: 2519-173X. DOI: `10.13154/tosc.v2018.i1.74-100`, see Chapter 6.

► OTHER PUBLICATIONS, PREPRINTS AND SUBMITTED WORK

Anne Canteaut, Lukas Kölsch, and Friedrich Wiemer. *Observations on the DLCT and Absolute Indicators*. 2019. IACR PREPRINT: `2019/848`, see Chapter 3.

To these five papers, all authors contributed equally. We merged the last paper with the results from [Li+19a]:

Anne Canteaut, Lukas Kölsch, Chunlei Li, Chao Li, Kangquan Li, Longjiang Qu, and Friedrich Wiemer. *On the Differential-Linear Connectivity Table of Vectorial Boolean Functions*. Submitted to IEEE Transactions on Information Theory. 2019.

I am also involved in the design of the NIST LWC submission SPOOK:

Davide Bellizia, Francesco Berti, Olivier Bronchain, Gaëtan Cassiers, Sébastien Duval, Chun Guo, Gregor Leander, Gaëtan Leurent, Itamar Levi, Charles Momin, Olivier Pereira, Thomas Peters, François-Xavier Standaert, and Friedrich Wiemer. *Spook: Sponge-Based Leakage-Resilient Authenticated Encryption with a Masked Tweakable Block Cipher*. Submitted to NIST LWC. 2019.

I contributed to the cryptanalysis of the underlying primitives CLYDE and SHADOW, in particular to the choice of round constants, the analysis of subspace trails, and division properties.

Further, I contributed to the development of SAGEMATH by authoring the following tickets. Additionally I also reviewed tickets not listed here.

Friedrich Wiemer. *Memory leak in algebraic_immunity of BooleanFunction class*. #14549. 2017

Rusydi H. Makarim and Friedrich Wiemer. *Computing nonlinear invariants in mq.SBox*. #21252. 2019

Friedrich Wiemer. *A mistake in the mq.Sbox.polynomials*. #22453. 2018

Friedrich Wiemer. *Refactor SBox code (move from crypto.mq to crypto)*. #22986. 2017

Friedrich Wiemer. *Add SBox instances*. #22988. 2017

Friedrich Wiemer. *Add SBox instances: DBlock Cipher*. #23830. 2017

Friedrich Wiemer. *BooleanFunction evaluated on Integer computes IndexError wrongly*. #24566. 2018

Friedrich Wiemer. *sbox linear approximation matrix scaling*. #24819. 2018

Rusydi H. Makarim and Friedrich Wiemer. *Implement algebraic_degree() for Boolean functions*. #24872. 2018

Friedrich Wiemer. *Huge delay introduced in SBox nonlinearity*. #25516. 2018

Friedrich Wiemer. *Speed up SBox module*. #25633. 2018

Friedrich Wiemer. *Rename SBox methods for difference_distribution_matrix and similar to _table*. #25708. 2018

Friedrich Wiemer. *Speed up SBox.boomerang_connectivity_matrix*. #25733. 2018

Friedrich Wiemer. *Crypto Linear Layer Module*. #25735. 2019

Friedrich Wiemer and Jean-Pierre Flori. *sig_check in boolean_function cython code*. #25735. 2018

Friedrich Wiemer. *construct sbox from univariate polynomial*. #25744. 2018

Friedrich Wiemer. *Add some more SBox constructions*. #25765. 2018

Friedrich Wiemer. *SBox.boomerang_uniformity*. #25766. 2018

Friedrich Wiemer. *Add derivative method for SBox*. #26009. 2019

Friedrich Wiemer. *Rename BooleanFunction.absolut_indicator to .absolute_indicator*. #28001. 2019

Before we proceed to the technical parts, we now recall basic techniques for the design and analysis of block ciphers.

2

The Art and Science of Block Cipher Design

"Nicht Kunst und Wissenschaft allein, Geduld will bei dem Werke sein."
—Johann Wolfgang von Goethe

"During human progress, every science is evolved out of its corresponding art."
—Herbert Spencer

► SHANNON'S "A Mathematical Theory of Cryptography" [Sha45] is commonly referred to as the founding stone of modern cryptography. While modern cryptography knows several kinds of encryption and authentication schemes, the trinity of symmetric cryptography consists of encryption schemes, hash functions and message authentication codes. This thesis focuses on the design and analysis of symmetric encryption schemes, in particular block ciphers. In the following sections we give an overview of constructions and techniques used for these tasks. While doing so, we discuss some issues in our actual understanding of the subject and where the work of a practising cryptographer might still be declared an art. At best, these issues can be answered by developing new tools and techniques to advance the science of cryptography.

Before coming to the technical parts, let us quickly recall some basic notations and linear algebra.

2.1 BASICS AND NOTATIONS

By $\mathbb{F}_2 = \{0, 1\}$ we denote the finite field with two elements, by $\mathbb{F}_{2^k}$ the extension field with 2^k elements, often also denoted as $\mathrm{GF}(2^k)$, and by $\mathbb{F}_2^n$ the n dimensional vector space over $\mathbb{F}_2$. We denote the addition in a finite field, an extension field and vector space by $+$; in the last case it corresponds to the component-wise addition in the underlying field. The addition is also commonly referred to as "XOR". For the (scalar) multiplication in $\mathbb{F}_2$, $\mathbb{F}_{2^n}$ and $\mathbb{F}_2^n$, as well as the canonical *inner product* in $\mathbb{F}_2^n$ we write $x \cdot y := \sum_i x_i y_i$ (for $x, y \in \mathbb{F}_2^n$), and it is clear from the context whether the multiplication or inner product is meant. The *span* of a set $U \subseteq \mathbb{F}_2^n$

$$\mathrm{Span}\{U\} := \left\{ \sum_{u \in U} \lambda_u u \;\middle|\; \lambda_i \in \mathbb{F}_2 \right\}$$

denotes the minimal subspace of $\mathbb{F}_2^n$ that contains all linear combinations of elements in U. Given a subspace $U \subseteq \mathbb{F}_2^n$, we denote a *basis* of U by $\mathrm{Basis}(U)$, that is a minimal subset of U such that it's span is equal to U:

$$U = \mathrm{Span}\{\mathrm{Basis}(U)\}\,.$$

Note that in general a basis is not unique. The canonical basis vectors are e_i, i. e. e_i has exactly one component equal to one at position i. The *dimension* of a subspace is the size of its basis, denoted $\dim U = |\mathrm{Basis}(U)|$. Given another subspace $V \subseteq \mathbb{F}_2^n$ with trivial intersection $U \cap V = \{0\}$, we denote the *direct sum* by

$$U \oplus V := \{u + v \mid u \in U \text{ and } v \in V\} \subseteq \mathbb{F}_2^n$$

and the *direct product* by

$$U \times V := \{(u, v) \mid u \in U \text{ and } v \in V\} .$$

If $U \oplus V = \mathbb{F}_2^n$, we say $V = U^c$ is a *complement* of U. If further for all $u \in U$

$$u \cdot v = 0 \quad \forall v \in V ,$$

$V = U^\perp$ is the *orthogonal complement*. It is also referred to as perpendicular complement or *dual space*.

The two following properties of orthogonal complements are well known.

Lemma 8 (Orthogonal Complement of direct products). *The orthogonal complement of a direct product is the direct product of its orthogonal complements:*

$$(V_1 \times \cdots \times V_n)^\perp = V_1^\perp \times \cdots \times V_n^\perp .$$

Proof. We give the proof for $n = 2$. It can be extended to general n by induction. Let V_1, $V_2 \subseteq \mathbb{F}_2^n$ be subspaces.

- $\subseteq$ Let $a \in (V_1 \times V_2)^\perp$. $\forall v = (v_1, v_2) \in V_1 \times V_2 : a \cdot v = a_1 \cdot v_1 + a_2 \cdot v_2 = 0$. As this holds for every v (in particular also for every $v = (0, v_2)$ and $v = (v_1, 0)$), both summands have to be constant zero. It follows that $a_1 \in V_1^\perp$, $a_2 \in V_2^\perp$ and thus $a = (a_1, a_2) \in V_1^\perp \times V_2^\perp$.
- $\supseteq$ Let $a = (a_1, a_2) \in V_1^\perp \times V_2^\perp$. $\forall (v_1, v_2) \in V_1 \times V_2 : \binom{a_1}{a_2} \cdot \binom{v_1}{v_2} = a_1 \cdot v_1 + a_2 \cdot v_2 = 0$. Thus $a = (a_1, a_2) \in (V_1 \times V_2)^\perp$.

Lemma 9 (Orthogonal Complement of subsets). *Let $U \subseteq V$ be a subspace. The orthogonal complement of V is a subset of the orthogonal complement of U. Formally:*

$$U \subseteq V \Rightarrow V^\perp \subseteq U^\perp .$$

Proof. From the definition of the orthogonal complement we have $\forall a \in V^\perp : \forall v \in V : a \cdot v = 0$. As U is a subset of V this also holds in particular for all the elements in U: $\Rightarrow \forall a \in V^\perp : \forall u \in U : a \cdot u = 0$. Which finally implies that $V^\perp \subseteq U^\perp$.

We denote the set of $\mathbb{F}_2$-linear functions $L : \mathbb{F}_2^n \to \mathbb{F}_2^m$ ($\mathbb{F}_2$-homomorphisms) as

$$\mathrm{Hom}_{\mathbb{F}_2}(\mathbb{F}_2^n, \mathbb{F}_2^m) := \left\{ L : \mathbb{F}_2^n \to \mathbb{F}_2^m \;\middle|\; L \text{ is linear} \right\} .$$

By fixing a basis there exists a natural isomorphism to $m \times n$ matrices, $\mathrm{Hom}_{\mathbb{F}_2}(\mathbb{F}_2^n, \mathbb{F}_2^m) \cong \mathrm{Mat}_{m,n}(\mathbb{F}_2)$ and we can thus identify every linear function L by a corresponding matrix T_L, called the *matrix representation* of L. The *kernel* of L (or T_L) is the set of all elements mapped to zero:

$$\mathrm{Ker}\, L := \left\{ x \in \mathbb{F}_2^n \;\middle|\; L(x) = 0 \right\} \subseteq \mathbb{F}_2^n$$

the *image* of L (or T_L) is

$$\operatorname{Im} L := \left\{ L(x) \,\middle|\, x \in \mathbb{F}_2^n \right\} \subseteq \mathbb{F}_2^m \,.$$

We denote the set of functions on $\mathbb{F}_2^n$ as Func_n, the set of permutations on $\mathbb{F}_2^n$ as Perm_n, and the set of Boolean functions, i. e. all $f : \mathbb{F}_2^n \to \mathbb{F}_2$, as BF_n. The subset of linear Boolean functions, $\mathrm{Hom}_{\mathbb{F}_2}(\mathbb{F}_2^n, \mathbb{F}_2) \subset \mathrm{BF}_n$, forms a subspace with the direct sum of functions $(f+g)(x) := f(x)+g(x)$. Further, the natural isomorphism for linear Boolean functions to $\mathbb{F}_2^n$ is

$$\begin{aligned} \Phi &: \mathbb{F}_2^n \to (\mathbb{F}_2^n \to \mathbb{F}_2) \\ \Phi(\alpha) &:= x \mapsto \alpha \cdot x \,. \end{aligned}$$

Affine linear Boolean functions are of the form $x \mapsto \alpha \cdot x + 1$ for $\alpha \in \mathbb{F}_2^n$. Any linear or affine Boolean function which is not constant is balanced, i. e. $f(x)$ equals zero and one equally often. This implies the fundamental fact

$$\sum_{x \in \mathbb{F}_2^n} (-1)^{\alpha \cdot x} = \begin{cases} 2^n & \text{if } \alpha = 0 \\ 0 & \text{else} \end{cases} \,. \tag{2.1}$$

The *adjoint* function $L^\top$ of L is the function, such that

$$\forall x, y \in \mathbb{F}_2^n : x \cdot L(y) = L^\top(x) \cdot y \tag{2.2}$$

The matrix representation of $L^\top$ corresponds to the transposed matrix representation of L. For further background on Boolean functions, we refer to Section 2.4.

A fundamental theorem is the following.

Theorem 10 (Rank-Nullity Theorem). *Given a linear function $L : \mathbb{F}_2^n \to \mathbb{F}_2^m$, where $n \geqslant m$. Then*

$$\dim \operatorname{Im} L + \dim \operatorname{Ker} L = n \,.$$

Proof. Denote $\mathrm{Basis}(\operatorname{Ker} L) = B = \{u_1, \ldots, u_m\}$. We can extend this basis to form a basis of the whole $\mathbb{F}_2^n = \mathrm{Span}\{A \cup B\}$, with $A = \{v_1, \ldots, v_{n-m}\}$. Note that we can write the image of L as

$$\begin{aligned} \operatorname{Im} L &= \mathrm{Span}\{L(A \cup B)\} \\ &= \mathrm{Span}\{L(u_1), \ldots, L(u_m), L(v_1), \ldots, L(v_{n-m})\} \\ &= \mathrm{Span}\{L(v_1), \ldots, L(v_{n-m})\} \\ &= \mathrm{Span}\{L(A)\} \,, \end{aligned}$$

where A forms a basis of $\operatorname{Im} L$, as we show next.

By contradiction. Assume there are $\lambda_i \in \mathbb{F}_2$ not all zero, such that

$$0 = \sum_{i=1}^{m} \lambda_i L(v_i)$$

and, as L is linear

$$= L\left(\sum_{i=1}^{m} \lambda_i v_i \right) ,$$

which implies that $\sum_{i=1}^{m} \lambda_i v_i$ is actually in the kernel of L. This contradicts the assumption of $A \cup B$ being a basis of $\mathbb{F}_2^n$.

Finally

$$\dim \operatorname{Im} L + \dim \operatorname{Ker} L = |A| + |B| = n - m + m = n$$

concludes the proof.

The name of the theorem stems from the fact that the dimension of the image is called *rank* and the dimension of the kernel *nullity*.

Up to isomorphisms, every extension field with 2^k elements is equal to the univariate polynomial ring over $\mathbb{F}_2$ modulo an irreducible polynomial $q(x)$ of degree k: $\mathbb{F}_{2^k} \cong \mathbb{F}_2[x]/q(x)$. In favour of a more compact notation, we stick to the common habit and write a polynomial as its coefficient vector interpreted as a hexadecimal number, i. e. $q(x) = x^4 + x + 1$ corresponds to `0x13`.

$$\begin{tikzcd}
\mathbb{F}_{2^k} \arrow[r, "\cdot\alpha"] \arrow[d, "\Phi"'] & \mathbb{F}_{2^k} \\
\mathbb{F}_2^k \arrow[r, "T_\alpha"'] & \mathbb{F}_2^k \arrow[u, "\Phi^{-1}"']
\end{tikzcd}$$

FIGURE 2.1: Multiplication using a vector space isomorphism

It is well known that we can represent the elements in a finite field with characteristic two as vectors with coefficients in $\mathbb{F}_2$. More precisely, by fixing a basis there exists a vector space isomorphism $\Phi : \mathbb{F}_{2^k} \to \mathbb{F}_2^k$. Every multiplication by an element $\alpha \in \mathbb{F}_{2^k}$ can then be described by a left-multiplication with a matrix $T_\alpha \in \mathbb{F}_2^{k \times k}$ as shown in Figure 2.1 (and is thus a linear operation). For an element α, T_α is usually called its *multiplication matrix*. Note that T_α depends on the chosen basis.

Given an $n \times n$ matrix $M = (\alpha_{i,j})$ with $\alpha_{i,j} \in \mathbb{F}_{2^k}$ for $1 \le i, j \le n$, we define its *binary representation* as the corresponding binary $nk \times nk$ matrix

$$\operatorname{BinMatr}(M) := (T_{\alpha_{i,j}}) \subseteq \operatorname{GL}(k, \mathbb{F}_2)^{n \times n} \subseteq (\mathbb{F}_2^{k \times k})^{n \times n} \cong \mathbb{F}_2^{nk \times nk} .$$

Here, $\operatorname{GL}(k, \mathbb{F}_2)$ denotes the general linear group, that is the group of invertible matrices over $\mathbb{F}_2$ of dimension $k \times k$.

Given a matrix M and a vector u, the *Hamming weights* $\operatorname{hw}(M)$ and $\operatorname{hw}(u)$ are defined as the number of nonzero entries in M and u, respectively. In the case of a binary vector $v \in \mathbb{F}_2^{nk}$, we define the *k-bit word Hamming weight* as $\operatorname{hw}_k(v) := \operatorname{hw}(v')$, where $v' \in (\mathbb{F}_2^k)^n$ is the vector that has been constructed by partitioning v into groups of k bits.

Finally, some notational conventions: We denote uniformly random sampling from a finite set by $\in_R$, that is $x \in_R M$ denotes that x is drawn uniformly at random from the set M. Given a vector $x \in \mathbb{F}_2^n$, we normally index a single element at position i as x_i. However, to avoid ambiguity, we might also denote the i-th element as $x[i]$. If x is split into parts, we denote a "slice" of x as $x[i : j] := (x_i, x_{i+1}, \ldots, x_j)$ for $i \leqslant j$.

2.2 PSEUDORANDOM PERMUTATIONS AND BLOCK CIPHERS

[Ker83] Kerckhoffs, "La cryptographie militaire"

First and foremost, we assume Kerckhoffs' century-old principle [Ker83, p. 12, 2nd condition], i. e. that the description of any encryption scheme (and of all its parts) is publicly available.

On the way to build a secure encryption system, several questions arise. In this section we strive to answer these in a top-down approach. The first and most important question is: how do we define security? There are multiple approaches to answer this initial problem, which we briefly sketch here. For the first part of this section, we orientate ourself by the asymptotic security notion, discussed in detail e. g. by Katz and Lindell [KL08]. To relate the security properties of particular block cipher constructions at the end of this section, we recall the concrete security notion, see [Bel+97]. Such an answer basically consists of three parts: what we want to achieve (the notion of security), what an adversary can do (her capabilities), and how she wants to break our security notion (her goal). We summarise these properties in our attacker model.

[KL08] Katz and Lindell *Introduction to Modern Cryptography*

[Bel+97] Bellare et al., "A Concrete Security Treatment of Symmetric Encryption"

Apart from this fundamental security discussion, we then shortly discuss ways of turning a block cipher into a secure encryption scheme. In particular, we cover how the security functionality is abstractly modeled by a pseudorandom permutation, which is then in practice instantiated as a block cipher. Such a pseudorandom permutation or block cipher is then turned into a secure encryption scheme using a mode of operation and a padding scheme.

Afterwards we turn to block ciphers in more detail and discuss typical constructions and inner building blocks for them. Eventually we recall concrete security statements for the most common block cipher construction.

2.2.1 *Generic Security for Generic Constructions*

Our ultimate goal for a secure encryption is to hide information from an adversary, i. e. providing *confidentiality*.[1] We thus want to design a secure encryption scheme for which an adversary is not capable of deducing any information from the plaintext.

▸ ATTACKER MODEL

We assume probabilistic polynomial-time (and thus computationally bounded) adversaries. Typically three *attack scenarios* are discussed which describe the capabilities of such an attacker.

Known Plaintext Attack (KPA) The adversary learns some pairs of plain- and ciphertexts, encrypted under the same key.

Chosen Plaintext Attack (CPA) The adversary 1) gets oracle access to the encryption function, 2) outputs two challenge plaintexts m_0, m_1, and learns the encryption c' of m_b for a uniformly random chosen bit b.

Chosen Ciphertext Attack (CCA) The adversary 1) gets oracle access to the encryption and decryption function, 2) outputs two challenge plaintexts m_0, m_1, learns the encryption c' of m_b, for a uniformly random chosen bit b, 3) is not allowed to query the decryption of c'.

Clearly, KPA is the weakest and CCA the strongest scenario.

When dealing with asymptotic security, the standard goal of an attacker in the above scenarios is to distinguish between two messages, i. e. guessing the bit b correctly. However, for an practical attack on an encryption scheme, other goals might also be of interest. Standard goals of an attacker are thus

Distinguishing The adversary has to distinguish two encryptions.

Plaintext Recovery The adversary has to decrypt a fresh ciphertext.

[1] The other two most important security goals are *integrity* (the data is unmodified) and *message authentication* (verifying the identity of the sender), which can be achieved by message authentication codes. However, for the remainder of the thesis, we focus on confidentiality.

Key Recovery The adversary has to recover the secret key used by the oracle.

In general, security against a distinguishing attack is the most generic goal and thus the most difficult to achieve. This is intuitively clear, as an attacker that successfully recovers a key or a plaintext can easily be turned into an attacker that distinguishes ciphertexts (i. e. by recovering the key, decrypting the ciphertext or directly recovering the plaintext and then distinguish based on this plaintext). If not specified, we thus assume the adversary's goal to be a distinguishing attack. We later discuss the connection between different goals in the specific case of symmetric cryptanalysis, but keep the discussion general for this introduction.

In order to prove asymptotic security, the standard technique are reductionistic, game-based proofs. For these, we define a security game that an adversary has to win with a high enough probability, in order to break the security claim of the scheme. One then shows under a suitable hardness assumption that such an attacker can be transformed into an attacker on the underlying hardness assumption. In the context of symmetric cryptography, we do not base our proofs on hardness assumptions, but give *concrete* bounds on the adversary's advantage. Thus we refrain from defining all these notions in full rigor; provable security is anyway not the topic of the remaining thesis.

Let us now turn to the constructive part and discuss approaches to achieve a secure encryption system.

▸ Constructions: Pseudorandom Permutations and Block Ciphers

The first encryption scheme that can be proven secure is the *One Time Pad (OTP)*. It was developed even before World War II and used then, during the Cold War, and most probable is still in use today. The encryption is simply $\text{OTP}_k : m \mapsto k + m$, where key and message are both in $\mathbb{F}_2^n$, while the crucial fact for the OTP's security is based on the *one time* use of the key k. In particular, for every message, we need a fresh key sampled uniformly at random.

The OTP comes with one prime advantage and one prime disadvantage. Using a uniformly random key only once enables us to prove the OTP secure in a information-theoretic sense – that is an even stronger security notion that we have discussed and remains secure even if we allow computationally *un*bounded adversaries. So it provides the best security we can hope for, see also [Sha49, Section 10] for a generalised treatment. However, as the downside of this *perfect security*, we need uniformly random keys of the same length as our message which we want to encrypt. In some sense this just shifts the problem of keeping the message confidential to the problem of exchanging a key of this length in a secure manner. While it might be possible for some parties (i. e. on a state level), to solve this problem, it is not practical for almost anyone.

[Sha49] Shannon, "Communication theory of secrecy systems"

Note that encrypting more than one message with the same key, directly breaks the security of the OTP (in a KPA scenario). Nevertheless, this blinding technique is basically at the heart of every encryption.

As this first encryption scheme is not practically useful in most cases, we are interested in alternative constructions. We cut the discussion short and mimic the OTP encryption, but with the goal of achieving a secure system that can use a fixed, small key to encrypt many messages. This is achievable, as we are aiming for security against computationally *bounded* adversaries.

A candidate for such a secure (probabilistic) encryption scheme is the following:

$$E_k(x) := (r, F_k(r) + x)\,. \tag{2.3}$$

Here, r is used to randomise the encryption as otherwise we cannot achieve CPA-security. Any deterministic encryption scheme is CPA-insecure: An attacker can simply query the encryption of its challenge messages and distinguish the challenge ciphertext based on these two encryptions.

The remaining, yet unspecified, part is F_k, and we want to take a more detailed look at what properties it has to exhibit, so that our scheme is secure.

The intuition we follow here is that the output of F_k, for any fixed key k, should look random and is (against our computationally bounded adversary) indistinguishable from a true uniformly random output. This is captured in the notion of a pseudorandom permutation, that is: an efficient, (length-preserving), keyed permutation which is computationally indistinguishable from a random permutation. Here, being efficient implies a polynomial-time algorithm for evaluating the permutation. Length-preserving is assumed for simplicity and denotes a permutation of $\mathbb{F}_2^n$, where the key is also an element of $\mathbb{F}_2^n$.[2] The notion of computationally indistinguishability captures the intuition that any probabilistic polynomial-time attacker's success probability is negligible, i. e. smaller than $1/\text{poly}(n)$. More formally we end up with the following definition.

Definition 11 (Pseudorandom Permutation)**.** Given a family of functions F on $\mathbb{F}_2^n$. We call F_k a *Pseudorandom Permutation (PRP)* if

- F_k is a permutation of $\mathbb{F}_2^n$,
- there exists polynomial-time algorithms evaluating F_k and F_k^{-1}, and finally
- for all probabilistic polynomial-time distinguishers D the following probability bound holds:

  $$\left|\Pr\left[D^{F_k}(1^n) = 1\right] - \Pr\left[D^{f}(1^n) = 1\right]\right| \leqslant \text{negl}(n)\,, \tag{2.4}$$

  where $\text{negl}(n) < \frac{1}{\text{poly}(n)}$ for all polynomials in n. The probability space is over the uniform choice of the key k, the permutation $f \in_R \text{Perm}_n$ and the randomness of D.

We call n the *security parameter*, which is, for length-preserving functions, equal to the *block size* and *key size*.

The above Eq. (2.4) has to be interpreted in the following way. The distinguisher D has oracle access to either F_k or to a random permutation f – here, oracle access implies that she can query the oracle for inputs to the corresponding functions and learns as a response the output of the

[2] We later discuss an instance of a pseudorandom permutation with a different key length than n – which in practice often occurs, but for the moment we stay with the simplified notation (as the generalisation is trivial anyway).

function. D has then to distinguish which oracle (the pseudorandom or the true random permutation) she has access to and answers this by outputting an answer bit b. This bit encodes the pseudorandom case as $b = 1$ and the true random case as $b = 0$. The distinguishers *advantage* is then the statistical distance between these two cases, i. e.

$$\mathrm{Adv}_D^{\mathrm{PRP}}(F_k, f) := \left|\Pr\left[D^{F_k}(1^n) = 1\right] - \Pr\left[D^{f}(1^n) = 1\right]\right| .$$

If D is not able to distinguish the pseudorandom permutation from a true random one, this distance is negligible and thus *computational indistinguishable*. The 1^n input specifies the all-one-vector $(1, \dots, 1) \in \mathbb{F}_2^n$, a technical detail to link the security parameter with the requested "polynomial in the input length"-runtime of D.

An unbounded adversary breaks computational indistinguishability

The following distinguisher D has a non negligible advantage.

- Get access to an oracle $\mathcal{O}$
- Query $\mathcal{O}$ for every possible input, i. e. compute the vector $v = (\mathcal{O}(i))_{i \in \mathbb{F}_2^n}$.
- For every possible $k \in \mathbb{F}_2^n$, compare $v \overset{?}{=} (F_k(i))_{i \in \mathbb{F}_2^n}$.
- If they are equal, output $b = 1$.
- If none of the vectors are equal, output $b = 0$.

Clearly, the probability to output the correct bit in the case of $\mathcal{O} = F_k$ is one. Moreover, D outputs the wrong bit in the case of $\mathcal{O} = f$, if the uniformly chosen random permutation $f = F_k$. This occurs with probability less than $1/(2^n - 1)$ and hence is negligible. D's overall advantage is thus close to one.

Note that a computationally unbounded adversary can always violate the computational indistinguishable notion, see the margin note. This problem does not occur when analysing the concrete security of a cipher, as the adversary's advantage is then bounded by her resources.

The above security definition can be extended, by allowing the distinguisher to access the inverse permutation, too. This leads to the notion of a strong pseudorandom permutation:

Definition 12 (Strong Pseudorandom Permutation)**.** A function-family F on $\mathbb{F}_2^n$ is a *strong Pseudorandom Permutation* if F is a PRP and for any $k \in \mathbb{F}_2^n$ and for all probabilistic polynomial-time distinguishers D the following probability bound holds:

$$\left|\Pr\left[D^{F_k, F_k^{-1}}(1^n) = 1\right] - \Pr\left[D^{f, f^{-1}}(1^n) = 1\right]\right| \leqslant \mathrm{negl}(n) .$$

The probability space is over the uniform choice of the key k, the permutation $f \in_R \mathrm{Perm}_n$ and the randomness of D.

However, returning to our construction, we do not need a strong PRP to show the following. Under the assumption that F is a PRP, the construction in Eq. (2.3) is CPA-secure. As we have not formally defined CPA-security, we leave out the proof. The interested reader can find the rigorous definitions, constructions and security proofs in [KL08].

[KL08] Katz and Lindell, *Introduction to Modern Cryptography*

On our way to a secure encryption scheme, we are now left with building a PRP. Block ciphers are designed to yield exactly this, a secure instance of a (strong) PRP (where we now allow the key to have a different length).

Definition 13 (Block Cipher)**.** A *Block Cipher* is a family of permutations E of the $\mathbb{F}_2^n$. Any instance of E is indexed by the *master key* $k \in \mathbb{F}_2^m$. We call n the *block size* and m the *key size* of E. Furthermore we require that (for every k) E_k (and its inverse $D_k = E_k^{-1}$) is efficiently computable, i. e. there exists a probabilistic polynomial-time algorithm evaluating E_k.

While standard block cipher designs fix one block size n, a technical oddity is that among others polynomial time looses its meaning. For the efficient computable aspect, we simply ignore this and just require "an efficient implementation". However, for the asymptotic security argument, this fixed n is problematic, too. Namely in order that a block cipher is secure, we would typically request it to be a (strong) PRP. Above, we have already mentioned the problems with a standard rigorous security proof: it would require proving the PRP property under a (weak) complexity theoretic hardness assumption (as it is done for most asymmetric designs). Now, fixing any block size n directly renders the asymptotic argument of bounding the advantage by some negligible term useless. In contrast to encryption schemes or similar, for block ciphers it is often not possible to give a generic construction for any block size n. This problem is typically circumvented by bounding the adversary's advantage with the concrete resources she spends, see also the discussion in Section 2.2.3, in particular Definition 18.

Nevertheless, proofs in this concrete setting still retain the problem that they base on idealised building blocks. Instead of doing such a proof, we typically "just assume" that any *practical* block cipher instance is secure, i. e. a (strong) PRP. The main reason for this approach is owed to efficiency requirements and the fact that designs asymptotically proven to be secure are often several magnitudes slower than modern block ciphers. Certainly, this missing last brick in the security proof should not be interpreted wrongly. Years of public scrutiny support our assumption that a specific block cipher actually yields a PRP. Comparing such a thoroughly understood design with asymmetric schemes that rely on e. g. the complex Gap-Diffie-Hellman assumption,[3] we should be much more confident in the security of the block cipher. Additionally to the used hardness assumption, a block cipher also typically corresponds to one fixed instance for any parameter set, while an asymmetric scheme might change the parameters for every key pair. This is the case e. g. in RSA, where every key pair has to use a different modulus (otherwise the encryption is directly broken), and is in some sense even worse for elliptic curve-based systems. Choosing efficient and secure elliptic curves is a highly non-trivial task – and examples as the `Dual_EC_DRBG` incident[4] confirm that it is possible to easily hide backdoors in such parameters, whereas no such way to hide backdoors for symmetric schemes are publicly known.

As our encryption scheme in Eq. (2.3), and analogously any block cipher, only encrypts single n-bit blocks of data, we typically need a *mode of operation* for E, such that we can encrypt data of arbitrary length. In the next section, we discuss these modes together with padding schemes, completing our treatment of generic constructions for symmetric encryption schemes.

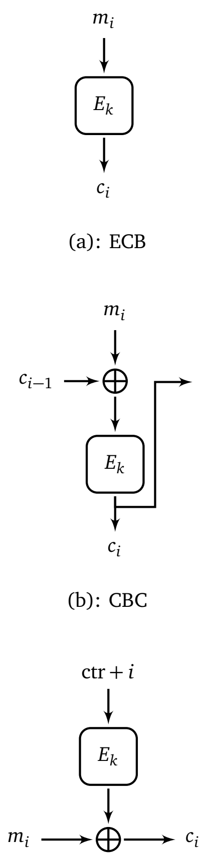
▸ Modes of Operation

Actually, the scheme from Eq. (2.3) can already be seen as a mode of operation, which turns a PRP into a fixed-length CPA-secure encryption function. However, as just mentioned, in practice we need to encrypt multiple blocks of data. A mode of operation then serves as a sort of protocol, describing how these multiple blocks can be encrypted. It is then important to show the security guarantees of such a mode, typically under the assumption that the used block cipher is a PRP.

Given several blocks of a plaintext message $m_1, m_2, \ldots \in \mathbb{F}_2^n$ and a block cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^n$. We recall three standard modes of operation, which encrypt the message as follows, see Figure 2.2 for a graphical representation.

[3] For Gap-Diffie-Hellman we assume the Decisional-Diffie-Hellman problem (given g, g^a, g^b, and g^c, decide if $c = ab$ for a group generator g and integers a, b and c) to be easy and the Computational-Diffie-Hellman problem (given g, g^a, g^b, compute g^{ab} for a group generator g and integers a and b) to be hard.

[4] See the Snowden leaks, in particular NSA's "Bullrun" program.



(a): ECB

(b): CBC

(c): CTR

FIGURE 2.2: Modes of Operation

Electronic Codebook (ECB) Deterministically encrypts every block independently:

$$c_i = E_k(m_i) .$$

Recall that any deterministic encryption does not provide CPA-security. This mode has thus a more educational purpose on "how to *not* encrypt several blocks".

Cipher Block Chaining (CBC) Chooses a random initial value IV $= c_0 \in_R \mathbb{F}_2^n$. The encryption adds the previous ciphertext to the next plaintext before encrypting it:

$$c_i = E_k(m_i + c_{i-1}) ,$$

and the IV $= c_0$ is sent as first ciphertext block.

Counter (CTR) Chooses a random counter ctr $\in_R \mathbb{F}_2^n$. The block cipher is then used in a streaming mode, where the counter is encrypted under the key and the produced output is added as a kind of key-stream to the plaintext. For the next block, the counter is increased, thus ctr $+ i$ (where the addition is done in $\mathbb{Z}/2^n\mathbb{Z}$) is used to encrypt the i-th block:

$$c_i = m_i + E_k(\text{ctr} + i) .$$

A nice property of the CTR mode is that it is embarrassingly parallel and computing errors in the encryption function only affect the corresponding message block (in contrast to the CBC mode, where errors propagate to all consecutive blocks).

As just mentioned, Eq. (2.3) can be seen as a fixed-length mode of operation; it is actually almost equal to the CTR mode. Besides the above mentioned modes, there are also other modes of operation which be used to build other symmetric primitives. In particular, to turn a block cipher into a hash function the Davies-Meyer mode (proven secure in [Bla+02]) can be used. For a message authentication code Bellare et al. proved the CBC-MAC to be secure, see [Bel+94]. More advanced modes are those that provide not only confidentiality, but also authentication. These are known as authenticated encryption (AE) or authenticated encryption with associated data (AEAD). As we normally want to provide a CCA-secure encryption scheme, such authentication is needed. Hence the NIST LWC competition's objective is to standardise such an AEAD scheme.

[Bla+02] Black et al., "Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV"

[Bel+94] Bellare et al., "The Security of Cipher Block Chaining"

These manifold applications have lead to the fact that block ciphers are today's best understood primitives in the area of symmetric cryptography. In particular the AES [AES] has been scrutinised by cryptanalysts ever since its development in 1998 [DR98] without any significant security threat discovered for the full cipher (see e. g. [GM00; Fer+01; Bir+09; BK09; Dun+10; Der+13; Gra+16; Gra+17; Røn+17]).

[DR98] Daemen and Rijmen, "The Block Cipher Rijndael"

Before turning our attention to actually designing block ciphers, we want to cover one last part which is needed for the design of an encryption scheme: padding. As the mode of operation allows us to encrypt messages of arbitrary length, it might well be the case that the message to be encrypted has not a length that is a multiple of the block cipher's block length. Padding is then used to expand the last message block to the block length. A very simple variant is the following bit-padding. Given a message with ℓ blocks, where the last block m_ℓ is of length $r \leqslant n$. We then extend m_ℓ by the string 10^k, where $k = n - r - 1$, if $r < n$, or $k = n - 1$ in the case $r = n$. Note that the last case actually result in a new message block $m_{\ell+1} = 10\ldots0$ and thus in a worst-case ciphertext expansion of n bits.

An alternative to a padding scheme which does not expand the ciphertext is the so called ciphertext stealing technique as described e. g. by Meyer and Matyas [MM82, Section 2.4].

[MM82] Meyer and Matyas *Cryptography: A New Dimension In Computer Data Security*

With the above parts, we have a thorough overview of the requirements and applications of block ciphers. We now turn to a detailed discussion on the inner constructions for block ciphers.

2.2.2 *Block Cipher Constructions*

Shannon's seminal work [Sha45; Sha49] already contains several ideas, which are nowadays standard design approaches for block ciphers.

[Sha45] Shannon, *A Mathematical Theory of Cryptography*

[Sha49] Shannon, "Communication theory of secrecy systems"

- Product Cipher
  A first idea is to build a secure encryption function by iterating a (weaker) function over several rounds, see [Sha45, Sections 37 and 38]. This is nowadays typically expressed as a product cipher, see also Figure 2.3.

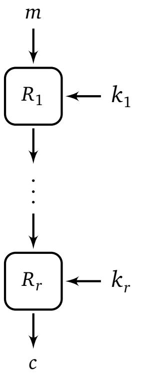


Figure 2.3: Product cipher

Definition 14 (Product cipher). Given a block cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^{rm} \to \mathbb{F}_2^n$, with keys $k = (k_1, \ldots, k_r) \in \mathbb{F}_2^{rm}$. We call E an (r-round) *product cipher*, if it can be written as the (r-times) application of (key-dependent) round functions R_{i,k_i}:

$$E_k^r = R_{r,k_r} \circ \cdots \circ R_{1,k_1}$$

where R_{i,k_i} permutes $\mathbb{F}_2^n$. Further, k is the *master key* and the k_i's are *round keys*.

- Key-Alternating Cipher and Substitution-Permutation-Network
  A second idea by Shannon are the notions of confusion and diffusion, see [Sha45, Section 35]. Roughly, the idea is that statistical properties are spread locally due to the confusion property of non-linear elements and globally due to the diffusion property of the linear elements in the round functions. Interestingly, this suggested splitting, see also [Sha49, pp. 712f], is the de facto standard in today's block ciphers. This can be seen as a first description of the so called SPN – the by far most common construction for modern block ciphers. Here, we directly introduce it with the idea of *key-alternating* ciphers. These introduce the key material in a very specific way: instead of consisting of merely key dependent round functions, the round keys are added to the state in between the round functions. Denoting this key addition by $\oplus_k : x \mapsto x + k$, we get the following definition.

Definition 15 (Key-alternating Substitution-Permutation-Network). Given a block cipher $E^r : \mathbb{F}_2^n \to \mathbb{F}_2^n$, with keys $k = (k_1, \ldots, k_r) \in \mathbb{F}_2^{rm}$. We call E^r an (r-round) *Substitution Permutation Network (SPN)*, if it can be written as the (r-times) application of round functions $R_{i,k_i} = L_{i,k_i} \circ S_{i,k_i}$, where the

S_{i,k_i} (substitution layer) are non-linear and can be split into the parallel application of (small) *S-boxes*, and the L_{i,k_i} (permutation or linear layer) are linear. We further call E *key-alternating*, if $R_{i,k_i} = \oplus_{k_i} \circ L_i \circ S_i$.

The first cipher following this approach was, to the best of our knowledge, the cipher MMB [Dae+93], while the name key-alternating cipher first appears in [DR01] and in the book describing the design of the AES [DR02]. Figure 2.4 depicts the most common key-alternating SPN instance:

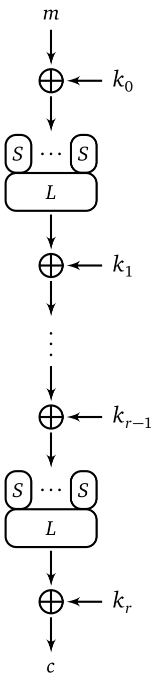


FIGURE 2.4: Key-Alternating Substitution Permutation Network

[Dae+93] Daemen et al., “Block ciphers based on Modular Arithmetic”

[DR01] Daemen and Rijmen, “The Wide Trail Design Strategy”

[DR02] Daemen and Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*

$$E^r_{k_0,\dots,k_r} = \oplus_{k_r} \circ L \circ S \circ \oplus_{k_{r-1}} \circ L \circ \dots \circ S \circ \oplus_{k_1} \circ L \circ S \circ \oplus_{k_0} ,$$

where the first round consists of a single key addition and all consecutive rounds are identical up to the used round key. This initial key addition is called (pre-) key-whitening and ensures that an adversary cannot peel of the first round without any guesses on the round key. Similarly, if we write the round functions with the key addition as first component, the last key addition is called post-key-whitening and serves the same purpose.

The prime advantage of introducing this much structure to the block cipher design is the simplification that accompanies such a structure. In particular, under some standard assumptions we can choose these parts, the substitution and the linear layer, independently of each other while standard security arguments still apply. We discuss this in more detail below.

Turning to the single layers of an SPN construction, the most common design of a substitution layer is to use small Substitution-boxes (S-boxes) in parallel for the whole state.

Definition 16 (S-box layer)**.** Let $S_i : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be S-boxes (permutations) for all $1 \leqslant i \leqslant k$. Then the *S-box layer* $S_1 || \cdots || S_k$ is the parallel application of all S_i:

$$\begin{aligned} S_1 || \cdots || S_k &: \left(\mathbb{F}_2^n\right)^k \to \left(\mathbb{F}_2^n\right)^k \\ S_1 || \cdots || S_k(x_1, \dots, x_k) &:= (S_1(x_1), \dots, S_k(x_k)) \,. \end{aligned}$$

If $S = S_1 = \cdots = S_k$, we may simply write $S^{||k} = S_1 || \cdots || S_k$.

This further splitting into S-boxes greatly simplifies the design exploration for substitution layers: typical S-boxes operate on a single byte (8 bit) or even only a byte nibble (4 bit). For such small input sizes it can actually be possible to find optimal instances for this building block, when no generic construction with equally good properties is known. As we restrict ourself to block ciphers over $\mathbb{F}_2^n$, the analysis of S-boxes is basically the study of (vectorial) Boolean functions. The corresponding background is thus covered in Section 2.4 on Boolean functions.

▶ FEISTEL NETWORK

Besides SPNs, the second popular construction for block ciphers are Feistel networks, named after their inventor Horst Feistel [Fei71].

[Fei71] Feistel, *Block Cipher Cryptographic System*

Definition 17 (Feistel network)**.** Given a product cipher $E^r : \mathbb{F}_2^n \to \mathbb{F}_2^n$, where $n = 2m$ is even, with keys $k = (k_1, \dots, k_r) \in \mathbb{F}_2^{r\ell}$. E^r is a *Feistel network*, if the round functions R_{i,k_i} can be written as

$$R_{i,k_i} : \mathbb{F}_2^m \times \mathbb{F}_2^m \to \mathbb{F}_2^m \times \mathbb{F}_2^m$$
$$x, y \mapsto \begin{cases} y, x + F_{i,k_i}(y) & \text{if } i < r \\ x + F_{i,k_i}(y), y & \text{else} \end{cases} .$$

Here, the F_{i,k_i} are called the *F-functions* of the Feistel network and often they come in a "key-alternating" fashion, i. e. $F_{i,k_i} = G + k_i$ for a $G : \mathbb{F}_2^m \to \mathbb{F}_2^m$. Leaving out the last round swapping has no influence on the security property, but allows to use the same implementation for decryption, as only the order of the round keys has to be reversed (if only the same G is used in the F-functions).

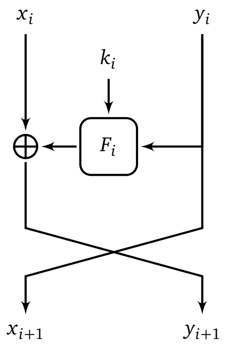


FIGURE 2.5: One round of a Feistel network

▸ KEY SCHEDULE

Up to now we have always assumed that the master key already consist of all the round keys $k = (k_1, \dots, k_r)$. For practical ciphers, this would imply a huge key size. Instead, the block cipher uses a *key schedule* $\mathrm{KS} : \mathbb{F}_2^m \to \left(F_2^m\right)^r$ that computes the r round keys given the initial master key.

However, as we typically assume independent round keys during our analysis, see Hypothesis 25, we ignore the key schedule from here on and assume the master key to contain all round keys.

2.2.3 *On the concrete and practical security of block cipher constructions*

Similar to the above discussed asymptotic security proofs, we can see the SPN and Feistel constructions as a way to turn an even simpler and smaller building block into a PRP, resp. block cipher. A legitimate question is thus how the security of SPNs or Feistel networks can be judged. While we could just use another asymptotic reduction, this notion also has drawbacks. Typically, asymptotic reductions come with a security degradation, caused by non-tight reductions or similar. Additionally, while this proof technique is obviously *asymptotic*, block ciphers are typically instantiated with fixed block lengths like 64- or 128-bit and thus work on finite domains. We thus cannot scale this security factor asymptotically and loose the main point of our argumentation.

A different approach is the notion of concrete security, developed by Bellare et al. [Bel+94] and applied to assess the security of typical modes of operations [Bel+97]. Here, one looks for (tight) reductions which link the adversary's advantage with the resources she spent, i. e. the running time and number of queries to the given oracle, in order to get a bound on the advantage. Additionally, by finding an attacker strategy matching this, it is turned into a (tight) "concrete" bound. A "concrete" pseudorandom permutation is then defined as follows, see [Bel+94, Definition 2.2].

[Bel+94] Bellare et al. "The Security of Cipher Block Chaining"

Definition 18 (Concrete Pseudorandom Permutation). Given a family of permutations F on $\mathbb{F}_2^n$, the *(concrete) PRP-insecurity* is measured as

$$\mathrm{Adv}_F^{\mathrm{PRP}}(q,t) = \max_D \left|\Pr\left[D^{F_k} = 1\right] - \Pr\left[D^f = 1\right]\right| ,$$

where the probability is over the uniform choice of the key k, the permutation $f \in_R \mathrm{Perm}_n$ and the randomness of the distinguishers D. Further the distinguishers D make at most q queries to the oracle and run in time at most t.

Note that this notion does not define a "secure" PRP, instead the PRP-insecurity provides a measure of the "pseudorandomness" of the studied permutation and we call a family of permutations a PRP, if $\mathrm{Adv}_F^{\mathrm{PRP}}$ is "low enough".

Interestingly, besides its overwhelming use in practice and the intense cryptanalytic efforts spent to understand its practical security, that is the resistance to known attacks, the concrete security of key-alternating ciphers has not been investigated until 2012. Here, following the above definition, an attacker is given access to the key-alternating round functions via oracle queries and additional oracle access to the block cipher or a random permutation. As any attack in this setting is obviously independent of any particular structure of the round function, those attacks are generic for all key-alternating ciphers. In this setting, the construction behind key-alternating ciphers is referred to as the iterated Even-Mansour construction. Indeed, the Even-Mansour cipher [EM97],

[EM97] Even and Mansour, "A Construction of a Cipher from a Single Pseudorandom Permutation"

$$E_{k_1,k_2} = \oplus_{k_2} \circ P \circ \oplus_{k_1} ,$$

can be seen as a one-round version of the key-alternating cipher where the round function P is a random permutation. It was further shown that the advantage of any adversary is bounded by $2pq \cdot 2^{-n}$, where p is the number of queries to the underlying permutation P and q is the number of queries to the cipher E. A polite note by Daemen [Dae93] showed a CPA attack reaching this bound, proving that it is tight.

[Dae93] Daemen "Limitations of the Even-Mansour Construction (Rump Session)"

The first result on the iterated Even-Mansour construction (basically focusing on the two-round version) was given in [Bog+12]. Since then, quite a lot of follow-up papers, see e. g. [And+13; GL15; LS15; HT16], managed to improve and generalise this initial result significantly. In particular, [HT16] managed to give a tight security bound for any number of rounds, i. e. they bounded the advantage of any adversary asking at most p queries to the permutations and q queries to the block cipher against an r-round Even-Mansour by

[Bog+12] Bogdanov et al., "Key-Alternating Ciphers in a Provable Setting: Encryption Using a Small Number of Public Permutations - (Extended Abstract)"

[HT16] Hoang and Tessaro, "Key-Alternating Ciphers and Key-Length Extension: Exact Bounds and Multi-user Security"

$$\mathrm{Adv}_{E^r}^{\mathrm{PRP}}(p,q) \leqslant 4^r p^r q \cdot 2^{-rn} .$$

Thus for breaking the r-round Even-Mansour construction, any attacker needs to make roughly $2^{\frac{r}{r+1}n}$ oracle queries.

While the attack by Daemen [Dae93] can be generalised to the iterated Even-Mansour construction, proving the bound tight, it is unsatisfactory for two reasons. First, one might hope to get better security bounds with different constructions and second one might hope to lower the requirement of relying on r random permutations.

This theoretical defect and the importance of encrypting small domains with full security (see e. g. [MY17]) motivated researchers to investigate alternative ways to construct block ciphers with the highest possible security level under minimal assumptions in ideal models, see also Problem 3 (Block Cipher Constructions). The most interesting result along those lines is the construction by Tessaro [Tes15]. His construction is based on the Swap-or-Not construction, which was designed for the setting where the component

[MY17] Miracle and Yilek, "Cycle Slicer: An Algorithm for Building Permutations on Special Domains"

[Tes15] Tessaro "Optimally Secure Block Ciphers from Ideal Primitives" functions are secret, see [Hoa+12]. Instead of being based on random permutations, this construction requires only a set of random (Boolean) functions. Tessaro's construction, coined Whitened Swap-Or-Not (WSN), requires only two public random (Boolean) functions f_i with n-bit input, and can be proven to achieve full security. We take a detailed look at this construction in Chapter 4. Up to the time of writing, there are no practical instances of this construction, we thus pursuit exactly such a first instance of the WSN, completing this solution to Problem 3.

[Hoa+12] Hoang et al., "An Enciphering Scheme Based on a Card Shuffle"

We now turn our attention to *practical security*, that is the resistance to actual cryptanalytic attacks.

2.3 PRACTICAL SECURITY – SYMMETRIC CRYPTANALYSIS

"*Secure, or not secure, that is the question*"
—freely adapted from Hamlet

Instantiating any ideal primitive, such as a PRP by building an actual block cipher, turns our system from a black box model into one which inner workings we can exploit in our analysis (due to Kerckhoffs' principle). Thus any proposed design has to withstand currently known attacks, and those which are only developed in the future, see Problem 4 (Provable Resistance) and Problem 5 (Algorithmic Resistance), to be still considered secure. During the design of a new cipher, the designers typically justify their design choices and give first cryptanalytic results on their new proposal. This initial cryptanalysis is also used to finalise parameter choices, e. g. the number of rounds used for the cipher. One surely can argue that this process is not optimal: future improvements to the cryptanalysis can hardly be taken into account and specifying an efficient cipher with a not too big security margin (as this would result in a performance loss) can only be done with the right intuition at hand. This process might thus be the most artistic part of the design process, however below we discuss heuristic arguments, which do work out well in practice, to give this process some guidance.

While today several distinct attacks on block ciphers exist, normally categorised as statistical, algebraic or structural, we concentrate on statistical attacks.

In particular we start with discussing the general idea of statistical attacks, that is finding some non-random characteristic of the (round-reduced) cipher under scrutiny and then turning this distinguisher into a key recovery attack. When covering the individual attacks, we recall standard assumptions which are taken to facilitate cryptanalysis (or make it possible in the first place). We do not exhaustively discuss statistical cryptanalysis techniques, but restrict ourselves to differential, linear, differential-linear, truncated differential and subspace trail cryptanalysis. The description of the standard security argument against differential and linear cryptanalysis, based on the so called wide trail strategy, is split into three parts: one in each of the corresponding parts on the attack and a third part concluding with the remaining part on the wide trail strategy, where we discuss two approaches to compute a lower bound on the number of active S-boxes. The margin note gives an overview of this structure.

Roadmap to Practical Security

1. Distinguisher vs. Key Recovery
2. Differential Cryptanalysis
3. Linear Cryptanalysis
4. The Wide Trail Strategy
5. Differential-Linear, Truncated Differentials and Subspace Trails

2.3.1 *From Distinguishing to Key Recovery*

A typical attack procedure works in a two step manner. We first analyse the cipher to find a distinguishing property over a reduced round version of the cipher, e. g. r rounds. This property can be used to distinguish an r-round version of the cipher from a random permutation.

In the second step, we extend this distinguisher to a key recovery attack over $r+s$ rounds. Hereto we pre- and append some rounds before and after the distinguisher (overall $s = s_1 + s_2$ more rounds). To still be able to evaluate the distinguisher, we analyse which key bits are necessary to evaluate if an input is valid for the distinguisher after s_1 rounds and which key bits influence the output of the distinguisher after s_2 rounds. We thus split $E^{s_1+r+s_2} = E^{s_2} \circ E^r \circ E^{s_1}$ into three parts, where we guess key bits in E^{s_1} and E^{s_2} and use a distinguisher over E^r, see Figure 2.6.

After this analysis, we end up with a distinguisher (which holds with probability p), and a key recovery strategy. This consists of conditions for the initial s_1 rounds which a valid input to the distinguisher has to fulfill and conditions for the last s_2 rounds which we need to invert to check the distinguishers output. Both sets of conditions come with a set of key bits that have an influence on them, hence these are the key bits we have to guess to check the conditions. A good example of this process is the paper by Lallemand and Rasoolzadeh, in particular see [LR17a, Section 6.3, pp. 14f] for the final strategy.

[LR17a] Lallemand and Rasoolzadeh, "Differential Cryptanalysis of 18-Round PRIDE"

The intuition why this key guessing works is captured in the hypothesis of wrong key randomisation, first stated in [Har+95, Section 4].

[Har+95] Harpes et al., "A Generalization of Linear Cryptanalysis and the Applicability of Matsui's Piling-Up Lemma"

Hypothesis 19 (Wrong Key Randomisation). *Given a fixed-key, r-round product cipher E_k^r with round function R_{k_i} and a distinguishing property D over $(r-1)$ rounds that holds with some probability p, i. e.* $\mathrm{Adv}_D^{\mathrm{PRP}}(E_k^{r-1}, f) = p$. *We assume that*

$$\mathrm{Adv}_D^{\mathrm{PRP}}(R^{-1}_{k_{wrong}} \circ E_k^r, f) \ll \mathrm{Adv}_D^{\mathrm{PRP}}(R^{-1}_{k_{right}} \circ E_k^r, f) = p$$

where the key recovery adversary's right last round key guess is k_{right} and the wrong last round key guesses are k_{wrong}.

In other words, if the key recovery adversary guesses the last round key correctly, she has the same advantage as the distinguisher on $r-1$ rounds. But if she instead guesses the last round key wrongly, her advantage drops significantly.

We justify this hypothesis based on the following observation. Write the r-round product cipher $E^r_{k_1,\ldots,k_r} = R_{k_r} \circ \cdots \circ R_{k_1}$. For the right last round key guess, the adversary basically peels of the last round,

$$R^{-1}_{k_r} \circ E^r_{k_1,\ldots,k_r} = R^{-1}_{k_r} \circ R_{k_r} \circ \cdots \circ R_{k_1} = R_{k_{r-1}} \circ \cdots \circ R_{k_1} = E^{r-1}_{k_1,\ldots,k_{r-1}} ,$$

while for wrong last round key guesses k'_r, she effectively adds another encryption round:

$$R^{-1}_{k'_r} \circ E^r_{k_1,\ldots,k_r} = R^{-1}_{k'_r} \circ R_{k_r} \circ \cdots \circ R_{k_1} = E^{r+1}_{k_1,\ldots,k_r,k'_r} .$$

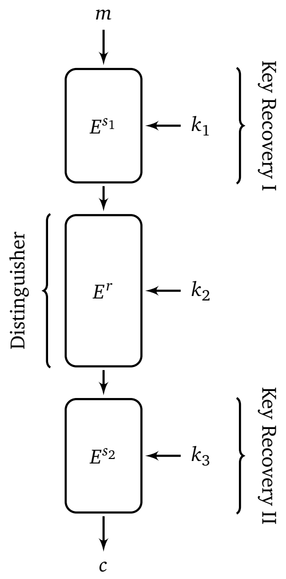


FIGURE 2.6: Generic key recovery strategy for an $r+s$ round cipher that exploits an r round distinguisher.

Regarding the key recovery strategies, note that first, typically the conditions and key bits to check are given in a quite "coarse" way, and second this process is up to now highly manual, requiring lots of "bit fiddling". In Chapter 7 we discuss steps towards automating this approach, giving first ideas for a solution to Problem 6 (Automate Key Recovery). There, we also discuss the exact key recovery in more detail, but for now this high level overview shall be enough for this introduction.

Let us now consider specific types of distinguishers. We start with the two most common types of cryptanalysis, differential and linear, and then look at three more advanced types of cryptanalysis, i. e. differential-linear, truncated differentials and subspace trails.

2.3.2 *Differential Cryptanalysis*

Biham and Shamir [BS91] developed differential cryptanalysis while studying the DES. The underlying idea is to trace the propagation of *differences* of plaintext pairs through the encryption rounds. In particular, we are interested in the probability

$$\Pr_x\left[E_k^r(x) + E_k^r(x+\alpha) = \beta\right] = p\,,$$

for some chosen α and $\beta \in \mathbb{F}_2^n$. We call α the input difference and β the output difference. If we found such a pair of differences that holds with a high enough probability, we can use it as a distinguisher for the cipher. Here, high enough means a probability $p > 2^{-n}$, which is related to the complexity of the resulting distinguisher. To observe the occurrence of the above event, i. e. the input difference α leading to an output difference β, we have to test $\mathcal{O}(p^{-1})$ plaintext pairs, after which we expect that the differential is fulfilled at least once.

[BS91] Biham and Shamir "Differential Cryptanalysis of DES-like Cryptosystems"

In particular, the probability of the above event to happen at least once after $1/p$ trials is the probability of at least one success for $1/p$ Bernoulli trials with probability p, if we assume the encryptions to be independent. We can compute this probability as (for n trials)

$$1 - \Pr_X\left[X = 0\right] = 1 - \left[\binom{n}{k} p^k (1-p)^{n-k}\right]_{k=0} = 1-(1-p)^n$$

where X follows a binomial distribution and for $n = 1/p$ trials we have

$$1-(1-p)^{1/p} \geqslant 1 - \frac{1}{e} > 0.63\,.$$

Thus c/p plaintext pairs should suffice, for a small constant, e. g. $c = 2$. As these pairs have to be chosen such that they follow the right input difference, differential cryptanalysis is a Chosen Plaintext Attack.

The margin notes summarises our top-down approach by giving an overview of the results discussed in this section.

Roadmap to Differential Cryptanalysis

$$\Pr_x\left[E_k^r(x) + E_k^r(x+\alpha) = \beta\right]$$

1. Differentials model this probability

$$= \Pr_x\left[\alpha \overset{E_k^r}{\underset{x}{\rightrightarrows}} \beta\right] = \mathrm{DP}_{E_k}(\alpha, \beta)$$

2. Differentials consist of many trails

$$= \sum_{\substack{\delta_i \in \mathbb{F}_2^n \\ \delta_0 = \alpha, \delta_r = \beta}} \Pr_x\left[\delta_0 \overset{F}{\underset{x}{\rightrightarrows}} \cdots \overset{F}{\underset{E_k^{r-1}(x)}{\rightrightarrows}} \delta_r\right]$$

3. Key influence is averaged to EDP

$$\mathrm{EDP}_E(\alpha, \beta)$$
$$= \frac{1}{2^{rm}} \sum_{k \in \left(\mathbb{F}_2^m\right)^r} \mathrm{DP}_{E_k}(\alpha, \beta)$$

4. Markov ciphers and indp. keys imply

$$= \frac{1}{2^{rm}} \sum_{\substack{k \in \mathbb{F}_2^{r \times m} \\ \delta_i \in \mathbb{F}_2^n \\ \delta_0 = \alpha, \delta_r = \beta}} \prod_{i=1}^{r} \Pr_x\left[\delta_{i-1} \overset{F}{\underset{x}{\rightrightarrows}} \delta_i\right]$$

5. Hypothesis of Stochastic Equivalence

$$\approx \mathrm{DP}_{E_k}(\alpha, \beta)$$

6. Differential trail probability for key-alternating SPN is bounded by

$$\prod_{i=1}^{r} \Pr_x\left[\delta_{i-1} \overset{L \circ S^{||k}}{\underset{x}{\rightrightarrows}} \delta_i\right] \leqslant p_S^{\#\text{act. S-boxes}}$$

▸ Differentials and Differential Trails

It is inherently hard to compute the exact probability p for a block cipher, as typical ciphers have block lengths of 64- or 128-bit or even larger and key lengths of 80- or 128-bit or again even larger. We thus have to take some assumptions to ease the cryptanalysis and are then able to exploit the special structure of product ciphers and SPNs in particular. Let us first look at the simplifications, we can use when taking the iterative structure into account.

The overall probability we are looking for is that of the so called differential to hold.

Definition 20 (Differential)**.** Given a cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^n$, an *input difference* $\alpha \in \mathbb{F}_2^n$, and an *output difference* $\beta \in \mathbb{F}_2^n$. The probability of the *differential* $\alpha \rightrightarrows^{E} \beta$ to hold is

$$\Pr_x\left[\alpha \underset{x}{\overset{E_k}{\rightrightarrows}} \beta\right] = \Pr_x\left[E_k(x) + E_k(x+\alpha) = \beta\right] = \mathrm{DP}_{E_k}(\alpha, \beta)\,,$$

and is also called the *differential probability*. For notational simplicity, we may leave out the limits, if clear from the context, and simply write $\alpha \rightrightarrows \beta$.

Due to block and key sizes, we cannot compute this probability for any practical secure block cipher. However, instead of only fixing the input and output differences, we can specify each intermediate difference in the encryption function. This leads to the definition of a differential trail or characteristic.

Definition 21 (Differential Trail)**.** Given an r-round cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^{rm} \to \mathbb{F}_2^n$ with round functions $R_i : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^n$, and $r+1$ differences $\delta_i \in \mathbb{F}_2^n$. The probability of the *differential trail* $\delta_0 \rightrightarrows^{R_{1,k_1}} \cdots \rightrightarrows^{R_{r,k_r}} \delta_r$ to hold is

$$\begin{aligned}&\Pr_x\left[\delta_0 \underset{x}{\overset{R_{1,k_1}}{\rightrightarrows}} \cdots \underset{E_k^{r-1}(x)}{\overset{R_{r,k_r}}{\rightrightarrows}} \delta_r\right] \\ &= \frac{\left|\left\{x \in \mathbb{F}_2^n \;\middle|\; \forall 1 \leqslant i \leqslant r : E_k^i(x) + E_k^i(x+\delta_0) = \delta_i\right\}\right|}{2^n}\,,\end{aligned}$$

that is the probability that each intermediate difference is fulfilled.

Using this differential trail notation, we can further specify the differential which consists of all trails starting with the input difference α and ending in the output difference β:

$$\Pr_x\left[\alpha \underset{x}{\overset{E_k^r}{\rightrightarrows}} \beta\right] = \sum_{\substack{\delta_i \in \mathbb{F}_2^n \\ \delta_0 = \alpha, \delta_r = \beta}} \Pr_x\left[\delta_0 \underset{x}{\overset{R_{1,k_1}}{\rightrightarrows}} \cdots \underset{E_k^{r-1}(x)}{\overset{R_{r,k_r}}{\rightrightarrows}} \delta_r\right].$$

While the differential can be expressed by all the differential trails it contains, it is not directly obvious why this helps in computing the probabilities. Actually these probabilities are also key dependent. This is captured in the notion of Expected Differential Probability that equals the differential probability averaged over all keys.

Definition 22 (Expected Differential Probability)**.** Given a cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^n$ and input/output differences $\alpha, \beta \in \mathbb{F}_2^n$. Then the *Expected Differential Probability (EDP)* over the master keys $k \in \mathbb{F}_2^m$ is

$$\mathrm{EDP}_E(\alpha, \beta) = \frac{1}{2^m} \sum_{k \in \mathbb{F}_2^m} \mathrm{DP}_{E_k}(\alpha, \beta) = \frac{1}{2^m} \sum_{k \in \mathbb{F}_2^m} \Pr_x\left[\alpha \underset{x}{\overset{E_k}{\rightrightarrows}} \beta\right], \tag{2.5}$$

The *Maximum Expected Differential Probability (MEDP)* bounds the EDP:

$$\mathrm{MEDP}(E) = \max_{\alpha, \beta \in F_2^n \setminus \{0\}} \mathrm{EDP}_E(\alpha, \beta)\,.$$

Lai et al. [Lai+91, Section 3] showed that for a so called *Markov* cipher and independently chosen round keys, the differential trails form a Markov chain – and thus the i-th differential probability $\Pr\left[\delta_{i-1} \rightrightarrows^{F} \delta_i\right]$ only depends on δ_{i-1}. This implies that the EDP of a single trail equals the product of the EDP's over the single rounds and we get for the differential's EDP:

[Lai+91] Lai et al. "Markov Ciphers and Differential Cryptanalysis"

$$\mathrm{EDP}_E(\alpha,\beta) = \frac{1}{2^{rm}} \sum_{k\in\left(\mathbb{F}_2^m\right)^r} \sum_{\substack{\delta_i\in\mathbb{F}_2^n\\ \delta_0=\alpha,\delta_r=\beta}} \prod_{i=0}^{r-1} \Pr_x\left[\delta_i \overset{R_{i,k_i}}{\underset{x}{\rightrightarrows}} \delta_{i+1}\right]. \tag{2.6}$$

A Markov cipher has to fulfill the following property.

Definition 23 (Markov cipher)**.** Given a product cipher $E^r : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^n$ with round functions R_i. Let further for all $\alpha,\beta\in\mathbb{F}_2^n$

$$\forall x\in\mathbb{F}_2^n : \sum_{k\in\mathbb{F}_2^m} \Pr\left[\alpha \overset{R_{i,k}}{\rightrightarrows} \beta\right] = \left|\left\{k\in\mathbb{F}_2^m \;\middle|\; R_{i,k}(x)+R_{i,k}(x+\alpha)=\beta\right\}\right|,$$

that is every differential probability over the round functions is independent of the chosen input value x. Then E^r is a *Markov cipher*.

In particular, key-alternating ciphers can be shown to be Markov ciphers.

For studying the differential behaviour of a cipher, we have to somehow approximate Eq. (2.5) resp. Eq. (2.6). This is typically done by assuming the hypothesis of stochastic equivalence. Lai et al. [Lai+91] introduced this hypothesis which states that the actual probability for any fixed round key equals the average, see also [DR02, p. 121].

[DR02] Daemen and Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*

Hypothesis 24 (Stochastic Equivalence)**.** *Given a block cipher E and input/output differences $\alpha,\beta\in\mathbb{F}_2^n$. We assume that the EDP of E averaged over all keys k, can be approximated by the differential probability of a fixed key instance E_k, i. e.*

$$\mathrm{EDP}_E(\alpha,\beta) \approx \mathrm{DP}_{E_k}(\alpha,\beta)\,.$$

for most of the keys.

While the hypothesis of stochastic equivalence seems to hold for many ciphers in practice, there are examples which contradict it. A prominent example is the block cipher IDEA that has a quite big class of weak keys (2^{32}) for which almost sure differential probabilities $\alpha \rightrightarrows \beta$ exist and thus $DP_{\mathrm{IDEA}_k}(\alpha,\beta) \ggg \mathrm{EDP}_{\mathrm{IDEA}}$, see also [DR02, p. 121].

Additionally, in order to simplify the analysis, we ignore influences of the key schedule and assume the cipher to use independent round keys.

Hypothesis 25 (Independent Round Keys)**.** *Given a block cipher E. We assume that the EDP for a fixed key instance E_k, where the round keys are derived by a key schedule from the master key k, can be approximated by the EDP averaged over all independently chosen round keys.*

While the influence of key schedules is a crucially understudied topic and for specific instances strange effects can occur, see [Abd+12; Kra+17a], Hypothesis 25 is seen as valid for most block ciphers.

[Abd+12] Abdelraheem et al., "On the Distribution of Linear Biases: Three Instructive Examples"

[Kra+17a] Kranz et al., "Linear Cryptanalysis: Key Schedules and Tweakable Block Ciphers"

Eventually, using the above assumptions and Eqs. (2.5) and (2.6), we approximate the differential probability for a key-alternating cipher as

$$\mathrm{EDP}_E(\alpha,\beta) \approx \sum_{\substack{\delta_i\in\mathbb{F}_2^n\\ \delta_0=\alpha,\delta_r=\beta}} \prod_{i=0}^{r-1} \Pr_x\left[\delta_i \overset{F}{\underset{x}{\rightrightarrows}} \delta_{i+1}\right],$$

where F is the (key-independent) round function. Looking at this equation it is immediately clear, why in general obtaining a (tight) bound of the

MEDP over a block cipher is a hard problem (recall Problem 1): the sum over all δ_i is huge, even for very few rounds. The standard way is thus to compute a reasonable bound for the probability of any *single differential trail* $\delta_0 \rightrightarrows \cdots \rightrightarrows \delta_r$ and then hope for the best: that the clustering effect for any differential is not too big and the EDP is actually well enough estimated by this differential trail bound, i. e.

$$\mathrm{EDP}_E(\alpha,\beta) \approx \prod_{i=0}^{r-1} \Pr_x\left[\delta_i \underset{x}{\overset{F}{\rightrightarrows}} \delta_{i+1}\right]. \tag{2.7}$$

Notable exceptions are [Hon+01; DR02; Par+03; CR15], that spend effort to prove bounds on the MEDP for two rounds of some key-alternating ciphers.

For our block cipher instance Bison, we can show, for the first time, that any differential basically collapses to a single differential trail, see Chapter 4 for further details.

Bounding the Differential Trail Probability

With the last Eq. (2.7) we have broken the problem of estimating the MEDP down to bounding the maximal differential probability *of any single trail*. As we see next, this can normally be done efficiently, when exploiting the wide trail strategy, developed by Daemen [Dae95]. This approach relates the probability of any differential trail to the number of S-boxes it involves and the best differential over one such S-box. Further details follow.

[Dae95] Daemen "Cipher and Hash Function Design Strategies based on linear and differential cryptanalysis"

To bound the differential trail, we have to take a look into the round function F of the (product) cipher under scrutiny. In order to simplify this further, let us assume that the cipher is actually a (key-alternating) SPN with block length $k \cdot n$. Thus $F = L \circ S^{||k}$, where $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is the S-box and $L : \mathbb{F}_2^{kn} \to \mathbb{F}_2^{kn}$ is the linear layer.

Following this round structure, we need to compute the probabilities for $\alpha \overset{S^{||k}}{\rightrightarrows} \beta$ and $\beta \overset{L}{\rightrightarrows} \gamma$, which is easy for both cases.

Lemma 26 (Differential Propagation over S-box Layer). *Given an S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and an input/output difference $\alpha, \beta \in \mathbb{F}_2^{kn}$. Write $\alpha = (\alpha_1, \ldots, \alpha_k)$ and β analogously, for $\alpha_i, \beta_i \in \mathbb{F}_2^n$. The differential probability over $S^{||k}$ is*

$$\mathrm{DP}_{S^{||k}}(\alpha,\beta) = \Pr_x\left[\alpha \underset{x}{\overset{S^{||k}}{\rightrightarrows}} \beta\right] = \prod_{i=1}^{k} \Pr_x\left[\alpha_i \underset{x}{\overset{S}{\rightrightarrows}} \beta_i\right].$$

Proof. This observation follows from the fact that the specific transitions over the single S-boxes are statistically independent.

Note that typical S-box sizes are up to 8-bit, and thus $\Pr\left[\alpha_i \overset{S}{\rightrightarrows} \beta_i\right]$ can easily be deduced from its absolute frequency.

Lemma 27 (Differential Propagation over Linear Layer). *Given a linear function $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and an input/output difference $\alpha, \beta \in \mathbb{F}_2^n$. The differential probability is*

$$\mathrm{DP}_L(\alpha,\beta) = \Pr_x\left[\alpha \underset{x}{\overset{L}{\rightrightarrows}} \beta\right] = \begin{cases} 1 & \text{if } \beta = L(\alpha) \\ 0 & \text{else} \end{cases}.$$

Proof. We can straightforwardly compute

$$\Pr_x\left[\alpha \overset{L}{\underset{x}{\rightrightarrows}} \beta\right] = \Pr_x\left[L(x) + L(x+\alpha) = \beta\right] = \Pr_x\left[L(\alpha) = \beta\right],$$

which is independent of x and thus fulfilled with probability one, if $\beta = L(\alpha)$.

To reach our final bound, we need one further notation: the activity of an S-box.

Definition 28 (Active and passive S-boxes)**.** Given an S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and an input difference $\alpha \in \mathbb{F}_2^{kn}$ for the S-box layer $S^{||k}$. Write $\alpha = (\alpha_1, \ldots, \alpha_k)$, for $\alpha_i \in \mathbb{F}_2^n$. The *activity pattern* of α is

$$\operatorname{activity}(\alpha) = (a_1, \ldots, a_k) \quad \text{where } \forall 1 \leqslant i \leqslant k : a_i = \begin{cases} 0 & \text{if } \alpha_i = 0 \\ 1 & \text{else} \end{cases} .$$

We say the i-th S-box is *active*, if $a_i = 1$, and *passive* else.

Now, denote by $p_S = \max_{\alpha,\beta \in \mathbb{F}_2^n \setminus \{0\}} \Pr\left[\alpha \overset{S}{\rightrightarrows} \beta\right]$ the maximal differential probability over the S-box S. In the next Section 2.4 we discuss how p_S relates to properties of (vectorial) Boolean functions. For now, we use p_S to bound the maximum differential trail property as follows.

Lemma 29 (Bound on the differential trail probability)**.** *Given an r-round key-alternating SPN cipher $E : \mathbb{F}_2^{kn} \times \mathbb{F}_2^{(r+1)kn} \to \mathbb{F}_2^{kn}$ with S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and linear layer $L : \mathbb{F}_2^{kn} \to \mathbb{F}_2^{kn}$. The differential trail property for any differential trail $\delta_0 \rightrightarrows_x^{L \circ S^{||k}} \cdots \rightrightarrows_x^{L \circ S^{||k}} \delta_r$ is bounded by*

$$\prod_{i=1}^{r} \Pr_x\left[\delta_{i-1} \overset{L \circ S^{||k}}{\underset{x}{\rightrightarrows}} \delta_i\right] \leqslant p_S^{\#\text{active S-boxes}},$$

where p_S is the maximal differential property over the S-box and

$$\#\textit{active S-boxes} = \sum_{i=0}^{r-1} \operatorname{hw}(\operatorname{activity}(\delta_i)).$$

Proof. Note that for an SPN any differential trail (with maximal probability) has to follow Lemmata 26 and 27. The differential probability for an S-box layer (with input difference $\alpha = (\alpha_1, \ldots, \alpha_k)$ and corresponding output difference β) is thus bounded by

$$\prod_{i=1}^{k} \Pr_x\left[\alpha_i \overset{S}{\underset{x}{\rightrightarrows}} \beta_i\right] = p_S^{\#\text{active S-boxes}},$$

where $\#\text{active S-boxes} = \operatorname{hw}(\operatorname{activity}(\alpha)) + \operatorname{hw}(\operatorname{activity}(\beta))$.

The statement then follows by induction over the number of rounds.

The only part left for bounding the differential trail probability is now a method to compute the maximal number of active S-boxes over all possible differential trails. As a similar argument based on the wide trail strategy is used for linear cryptanalysis, we defer this discussion to the end of this section.

We conclude this part with the standard heuristic for resistance against differential cryptanalysis that is used for Problem 4 (Provable Resistance). The required bound on p_S is derived from the data complexity of the attack: if we exceed this data complexity, an attack would require more data than the full codebook.

Heuristic 30 (Heuristic Security against Differential Cryptanalysis)**.** *Given an r-round SPN cipher $E : \mathbb{F}_2^{kn} \times \mathbb{F}_2^{(r+1)kn} \to \mathbb{F}_2^{kn}$ with S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and linear layer $L : \mathbb{F}_2^{kn} \to \mathbb{F}_2^{kn}$. We assume the MEDP($E$) is bounded by*

$$MEDP(E) \leqslant p_S^{\#\text{active S-boxes}} ,$$

where p_S is the maximal differential probability over S and #active S-boxes is a lower bound for the number of active S-boxes through r rounds of E.

We claim E to be secure against differential cryptanalysis, if

$$p_S^{\#\text{active S-boxes}} < 2^{-kn} .$$

2.3.3 *Linear Cryptanalysis*

Matsui [Mat94] developed linear cryptanalysis and was able to give the first better-than-brute-force attack on DES. In this comprehension, we follow the systematisation of Kranz et al. [Kra+17a, Section 2]. However, as we assume Independent Round Keys, we ignore the influence of the key scheduling and refer the interested reader to [Kra+17a] for the key schedule details. The margin note sketches the structure of this section.

[Mat94] Matsui "Linear Cryptanalysis Method for DES Cipher"

[Kra+17a] Kranz et al. "Linear Cryptanalysis: Key Schedules and Tweakable Block Ciphers"

Matsui's main idea is to approximate the cipher by linear functions. In particular, we look for two $\alpha, \beta \in \mathbb{F}_2^n$, such that

$$\Pr_x\left[\alpha \cdot x = \beta \cdot E_k(x)\right] \tag{2.8}$$

is far from uniformly distributed, i. e. far from 0.5.

Roadmap to Linear Cryptanalysis

$$\Pr_x\left[\alpha \cdot x = \beta \cdot E_k^r(x)\right] = \frac{1}{2} + \varepsilon^L_{E_k^r}(\alpha,\beta)$$

1. Bias, Correlation and Walsh coefficient equal up to scaling

$$2^{n+1}\,\varepsilon^L_{E_k}(\alpha,\beta) = 2^n\,\mathbf{cor}_{E_k}(\alpha,\beta) = \widehat{E_k}(\alpha,\beta)$$

2. Linear hull over composed $F = H \circ G$

$$\widehat{F}(\alpha,\beta) = 2^{-n} \sum_{\theta\in\mathbb{F}_2^n} \widehat{G}(\alpha,\theta)\widehat{H}(\theta,\beta)$$

3. Key influence $E_k(x) = F(x,k)$:

$$\widehat{E_k}(\alpha,\beta) = 2^{-m} \sum_{\theta\in\mathbb{F}_2^m} (-1)^{\theta\cdot k}\widehat{F}((\alpha,\theta),\beta)$$

4. Linear hull of a key-alternating SPN

5. Maximum Expected Linear Potential

$$\mathrm{MELP}(E) = \max_{\alpha,\beta\in\mathbb{F}_2^n\setminus\{0\}} \frac{1}{2^m} \sum_{k\in\mathbb{F}_2^m} \mathbf{cor}_{E_k}(\alpha,\beta)^2$$

6. Heuristic security argument

$$\mathrm{MELP}(E) \leqslant |c_S|^{\#\text{active S-boxes}}$$

Definition 31 (Linear Hull, Input/Output Masks)**.** Given a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, an *input mask* $\alpha \in \mathbb{F}_2^n$, and an *output mask* $\beta \in \mathbb{F}_2^m$. The *linear hull* or *linear approximation* of F is

$$\alpha \cdot x = \beta \cdot F(x) .$$

We denote it by $\alpha \to_x^F \beta$.

Overall, three different metrics for linear approximations are used.

Definition 32 (Bias, Correlation, and Walsh transform)**.** Given a function F and a linear approximation $\alpha \to^F \beta$ of it. The *linear bias* of the linear approximation is $\varepsilon_F^L(\alpha,\beta)$ with

$$\Pr_x\left[\alpha \cdot x = \beta \cdot E_k(x)\right] = \Pr_x\left[\alpha \xrightarrow[x]{F} \beta\right] = \frac{1}{2} + \varepsilon_F^L(\alpha,\beta) ,$$

the *correlation* $\mathbf{cor}_F(\alpha,\beta)$ is

$$\mathbf{cor}_F(\alpha,\beta) := 2 \cdot \Pr_x\left[\alpha \xrightarrow[x]{F} \beta\right] - 1 ,$$

and the *Walsh transform* of F at point (α,β) is

$$\widehat{F}(\alpha,\beta) := \sum_{x\in\mathbb{F}_2^n} (-1)^{\alpha\cdot x + \beta\cdot F(x)} .$$

$\widehat{F}(\alpha, \beta)$ is also called the Fourier transform, Walsh-Hadamard transform, or the corresponding coefficient. All three metrics are equal up to scaling.

Lemma 33. *Given a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. For any $\alpha, \beta \in \mathbb{F}_2^n$ it holds that*

$$\widehat{F}(\alpha, \beta) = 2^n \, \mathbf{cor}_F(\alpha, \beta) = 2^{n+1} \, \varepsilon_F^L(\alpha, \beta) \, .$$

Proof. The correlation is defined as

$$\begin{aligned} \mathbf{cor}_F(\alpha, \beta) &= 2 \cdot \Pr_x \left[\alpha \cdot x = \beta \cdot F(x) \right] - 1 \\ &= 2 \left(\Pr_x \left[\alpha \cdot x = \beta \cdot F(x) \right] - \frac{1}{2} \right) = 2 \cdot \varepsilon_F^L(\alpha, \beta) \, . \end{aligned}$$

It is thus left to show that $\widehat{F}(\alpha, \beta) = 2^n \, \mathbf{cor}_F(\alpha, \beta)$. By definition

$$\widehat{F}(\alpha, \beta) = \sum_{x \in \mathbb{F}_2^n} (-1)^{\alpha \cdot x + \beta \cdot F(x)} \, ,$$

defining the set $A = \left\{ x \in \mathbb{F}_2^n \;\middle|\; \alpha \cdot x = \beta \cdot F(x) \right\}$, this is

$$= |A| - \left| \mathbb{F}_2^n \setminus A \right| \, ,$$

now note that $|A| = 2^n \Pr_x \left[\alpha \cdot x = \beta \cdot F(x) \right]$ and thus

$$\begin{aligned} &= 2^n \Pr_x \left[\alpha \cdot x = \beta \cdot F(x) \right] - \left(2^n - 2^n \Pr_x \left[\alpha \cdot x = \beta \cdot F(x) \right] \right) \\ &= 2^n \left(2 \cdot \Pr_x \left[\alpha \cdot x = \beta \cdot F(x) \right] - 1 \right) \\ &= 2^n \, \mathbf{cor}_F(\alpha, \beta) \, , \end{aligned}$$

concluding the proof.

The data complexity of a linear distinguisher is given by Matsui as $\mathcal{O}(\mathbf{cor}_{E_k}(\alpha, \beta)^{-2})$, see [Mat94, Lemma 2 and Table 2], or equivalently $\mathcal{O}(\mathrm{LP}_{E_k}(\alpha, \beta)^{-1})$, where LP is the linear potential, see the end of this part, i. e. Definition 40. Note that, in contrast to differential cryptanalysis, linear cryptanalysis is a Known Plaintext Attack.

► Linear Hulls and Linear Trail

Similar to the differential case, computing the probability of a linear approximation is computational infeasible for any reasonable block cipher. We thus have to look into the cipher's structure, and again similar to the differential case, we can exploit it's inner workings.

A common approach is to define a linear trail as a linear approximation over each round, specifying all the intermediate masks.

Definition 34 (Linear Trail)**.** Given an r-round cipher $E^r : \mathbb{F}_2^n \times \mathbb{F}_2^{rm} \to \mathbb{F}_2^n$ with round functions $R_i : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^n$, and $r+1$ masks $\delta_i \in \mathbb{F}_2^n$. Then

$$\delta_0 \overset{R_{1,k_1}}{\to} \cdots \overset{R_{r,k_r}}{\to} \delta_r$$

is a *linear trail* through E^r.

To compute the corresponding bias, we then need the Piling-Up lemma, see [Mat94, Lemma 3]

Lemma 35 (Piling-Up)**.** *Given n independent random binary variables x_i, where* $\Pr_{x_i}[x_i = 0] = p_i$*. Then*

$$\Pr_{x_i}[x_1 + \cdots + x_n = 0] = \frac{1}{2} + 2^{n-1}\prod_{i=1}^{n}\left(p_i - \frac{1}{2}\right).$$

The main drawback of defining trails this way, is the assumption of independent masks for the piling-up lemma. Indeed, Nyberg [Nyb94] showed a way to avoid this assumption, by computing the linear hull $\alpha \to \beta$ via Walsh coefficients. The same result follows with a different approach, see [Dae95; DR02; Dae+95]; however we stick to Nyberg's approach, as the resulting proofs are in our opinion simpler.

[Nyb94] Nyberg "Differentially Uniform Mappings for Cryptography"

[Dae95] Daemen, "Cipher and Hash Function Design Strategies based on linear and differential cryptanalysis"

[DR02] Daemen and Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*

[Dae+95] Daemen et al., "Correlation Matrices"

We first look at the Walsh coefficients of composite functions. Afterwards, we apply this result to (key-alternating) SPNs, as these are composed of the corresponding round functions. The following lemmata provide the basis for our statements. The first one was also discussed by Daemen et al. [Dae+95, Eq. (15)].

Lemma 36 (Linear Trail Composition)**.** *Given $F, G, H : \mathbb{F}_2^n \to \mathbb{F}_2^n$, such that $F = H \circ G$. Then for all $\alpha, \beta \in \mathbb{F}_2^n$:*

$$2^n \widehat{F}(\alpha, \beta) = \sum_{\theta \in \mathbb{F}_2^n} \widehat{G}(\alpha, \theta) \cdot \widehat{H}(\theta, \beta).$$

Proof. By the definition of the Walsh transform, we have

$$\sum_{\theta \in \mathbb{F}_2^n} \widehat{G}(\alpha, \theta) \cdot \widehat{H}(\theta, \beta) = \sum_{\theta, x, y \in \mathbb{F}_2^n} (-1)^{\alpha \cdot x + \theta \cdot G(x) + \theta \cdot y + \beta \cdot H(y)}$$

now reorder the sums

$$= \sum_{x,y} (-1)^{\alpha \cdot x + \beta \cdot H(y)} \sum_{\theta} (-1)^{\theta \cdot (y + G(x))}$$

from Eq. (2.1) it follows that if and only if $y = G(x)$ the last sum collapses to a factor of 2^n, otherwise to 0

$$= 2^n \sum_{x} (-1)^{\alpha \cdot x + \beta \cdot H(G(x))}$$

$$= 2^n \widehat{F}(\alpha, \beta)$$

as claimed. ∎

Matsui studied the spreading of masks through the components of Feistel networks, see [Mat95, Section 3 and Fig. 3]. For key-alternating ciphers, the particular case of the mask propagation through the key-XOR is interesting.

[Mat95] Matsui, "On Correlation Between the Order of S-boxes and the Strength of DES"

Lemma 37 (Composition with XOR)**.** *Given $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, and $G_I, G_O : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$, such that $G_I(x, y) = F(x + y)$ and $G_O(x, y) = F(x) + y$. Then for all $\alpha, \beta, \gamma \in \mathbb{F}_2^n$:*

$$\widehat{G_I}((\alpha, \beta), \gamma) = \begin{cases} 2^n \widehat{F}(\alpha, \gamma) & \text{if } \alpha = \beta \\ 0 & \text{else} \end{cases},$$

and

$$\widehat{G_O}((\alpha, \beta), \gamma) = \begin{cases} 2^n \widehat{F}(\alpha, \gamma) & \text{if } \beta = \gamma \\ 0 & \text{else} \end{cases}.$$

Proof. We start with the first claim. By definition

$$\widehat{G_I}((\alpha,\beta),\gamma) = \sum_{x,y\in\mathbb{F}_2^n} (-1)^{\alpha\cdot x+\beta\cdot y+\gamma\cdot F(x+y)}$$

now split the sum and for every fixed y we substitute $x' = x + y$

$$= \sum_{y\in\mathbb{F}_2^n} (-1)^{\beta\cdot y} \sum_{x'\in\mathbb{F}_2^n} (-1)^{\alpha\cdot(x'+y)+\gamma\cdot F(x')}$$
$$= \sum_{y} (-1)^{(\alpha+\beta)\cdot y} \sum_{x'} (-1)^{\alpha\cdot x'+\gamma\cdot F(x')}$$

and again from Eq. (2.1)

$$= \sum_{y} (-1)^{(\alpha+\beta)\cdot y} \widehat{F}(\alpha,\gamma) = \begin{cases} 2^n \widehat{F}(\alpha,\gamma) & \text{if } \alpha=\beta \\ 0 & \text{else} \end{cases} .$$

For the second claim, we have

$$\widehat{G_O}((\alpha,\beta),\gamma) = \sum_{x,y\in\mathbb{F}_2^n} (-1)^{\alpha\cdot x+\beta\cdot y+\gamma\cdot(F(x)+y)}$$
$$= \sum_{x,y} (-1)^{\alpha\cdot x+(\gamma+\beta)\cdot y+\gamma\cdot(F(x))}$$
$$= \sum_{y} (-1)^{(\gamma+\beta)\cdot y} \sum_{x} (-1)^{\alpha\cdot x+\gamma\cdot(F(x))} .$$

Finally, Eq. (2.1) concludes the proof. ∎

Using the Walsh transform, the influence of the key can be addressed quite simply.

Lemma 38 (Walsh transform of keyed function)**.** *Given a fixed-key function $E_k : \mathbb{F}_2^n \to \mathbb{F}_2^n$ for a key $k \in \mathbb{F}_2^m$. Write $E_k(x) = F(x,k)$ for an $F : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^n$. Then the Walsh transform of E_k is*

$$2^m \widehat{E_k}(\alpha,\beta) = \sum_{\theta\in\mathbb{F}_2^m} (-1)^{\theta\cdot k} \widehat{F}((\alpha,\theta),\beta) .$$

Proof.

$$\sum_{\theta\in\mathbb{F}_2^m} (-1)^{\theta\cdot k} \widehat{F}((\alpha,\theta),\beta) = \sum_{\theta} (-1)^{\theta\cdot k} \sum_{\substack{x\in\mathbb{F}_2^n \\ y\in\mathbb{F}_2^m}} (-1)^{\alpha\cdot x+\theta\cdot y+\beta\cdot F(x,y)}$$
$$= \sum_{x,y} (-1)^{\alpha\cdot x+\beta\cdot E_y(x)} \sum_{\theta} (-1)^{\theta\cdot(k+y)}$$
$$= 2^m \sum_{x} (-1)^{\alpha\cdot x+\beta\cdot E_k(x)} = 2^m \widehat{E_k}(\alpha,\beta)$$

∎

We can now compute the linear hull of a key-alternating SPN.

Proposition 39 (Linear Hull for key-alternating SPN)**.** *Given a key-alternating SPN $E^r_{k_0,\dots,k_r} : \mathbb{F}_2^n \to \mathbb{F}_2^n$ with round functions $R_i : \mathbb{F}_2^n \to \mathbb{F}_2^n$. In particular*

$$E^r_{k_0,\dots,k_r}(x) := R_r(\cdots(R_1(x+k_0)+\cdots)+k_r .$$

Then

$$2^{(r-1)n} \widehat{E^r_{k_0,\dots,k_r}}(\alpha,\beta) = \sum_{\substack{\theta\in\mathbb{F}_2^{(r+1)\times n} \\ \theta_0=\alpha,\theta_r=\beta}} (-1)^{\theta\cdot k} \prod_{i=1}^{r} \widehat{R_i}(\theta_{i-1},\theta_i) .$$

Proof. By induction over the number of rounds r. For the base case with $r = 1$ we have $E_{k_0,k_1}(x) = R_1(x+k_0)+k_1 = F(x,k_0,k_1)$. Applying Lemma 38 implies

$$\widehat{E_{k_0,k_1}}(\alpha,\beta) = 2^{-2n} \sum_{\theta_0,\theta_1\in\mathbb{F}_2^n} (-1)^{\theta_0\cdot k_0+\theta_1\cdot k_1}\widehat{F}((\alpha,\theta_0,\theta_1),\beta)$$

and applying both cases of Lemma 37 restricts $\theta_0 = \alpha$ and $\theta_1 = \beta$, so that

$$= 2^{-2n}\cdot(-1)^{\alpha\cdot k_0+\beta\cdot k_1}\cdot 2^{2n}\cdot\widehat{R_1}(\alpha,\beta)$$

as claimed.

For the induction step with $r+1$ rounds, let

$$E^{r+1}_{k_0,\dots,k_{r+1}}(x) = R_{r+1}(E^r_{k_0,\dots,k_r}(x)) + k_{r+1} = F_{k_0,\dots,k_r}(x,k_{r+1})\,.$$

Then, applying Lemma 38 again gives

$$\widehat{E^{r+1}_{k_0,k_{r+1}}}(\alpha,\beta) = 2^{-n}\sum_{\theta_{r+1}\in\mathbb{F}_2^n}(-1)^{\theta_{r+1}\cdot k_{r+1}}\widehat{F_{k_0,\dots,k_r}}((\alpha,\theta_{r+1}),\beta)$$

the second case of Lemma 37 implies $\theta_{r+1} = \beta$

$$= (-1)^{\beta\cdot k_{r+1}}\widehat{R_{r+1}\circ E^r_{k_0,\dots,k_r}}(\alpha,\beta)$$

which we can decompose due to Lemma 36 into

$$= 2^{-n}(-1)^{\beta\cdot k_{r+1}}\sum_{\theta_r\in\mathbb{F}_2^n}\widehat{E^r_{k_0,\dots,k_r}}(\alpha,\theta_r)\cdot\widehat{R_{r+1}}(\theta_r,\beta)$$

and the induction hypothesis

$$= 2^{-rn}\sum_{\substack{\theta\in\mathbb{F}_2^{(r+2)n}\\ \theta_0=\alpha,\theta_{r+1}=\beta}}(-1)^{\theta\cdot k}\prod_{i=1}^{r+1}\widehat{R_i}(\theta_{i-1},\theta_i)$$

completes the proof.

In order to relate the resistance of a cipher against linear cryptanalysis, we need to bound the probability of any linear approximation. This results in the so-called *Linear Potential* and its maximum averaged over all keys, the *Maximum Expected Linear Potential*.

Definition 40 (Linear Potential, Expected Linear Potential and Maximum Expected Linear Potential)**.** Given a block cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^n$ with keys $k \in \mathbb{F}_2^m$. For any linear approximation $\alpha \xrightarrow{E_k} \beta$, we denote by *Linear Potential (LP)*

$$\mathrm{LP}_{E_k}(\alpha,\beta) := \mathbf{cor}_{E_k}(\alpha,\beta)^2$$

the *key dependent* squared correlation, by *Expected Linear Potential (ELP)*

$$\mathrm{ELP}_E(\alpha,\beta) := \frac{1}{2^m}\sum_{k\in\mathbb{F}_2^m}\mathrm{LP}_{E_k}(\alpha,\beta)$$

the expected squared correlation over the keys, and by *Maximum Expected Linear Potential (MELP)*

$$\mathrm{MELP}(E) := \max_{\alpha,\beta\in\mathbb{F}_2^n\setminus\{0\}}\mathrm{ELP}_E(\alpha,\beta)\,.$$

The reason why we look at the squared correlation is that we want to eliminate the sign of the correlation – for an effective linear cryptanalysis attack the *absolute* correlation should be high.

As in the case of differential cryptanalysis, it is in general a hard problem to bound the MELP of a complete linear hull, see Problem 2. While there are some notable exceptions again that give upper bounds on the MELP for two rounds of the AES, see [Hon+01; Par+03; CR15], the standard way is to give a bound on the correlation of any *single linear trail* and hope that the clustering effect in the linear hull is not too big. We elaborate next how this bound is typically computed.

Security Argument for Linear Cryptanalysis

We can exploit the SPN's special round structure to refine the linear hull a last time. Using the wide trail strategy again, this relates the linear hull potential to the number of S-boxes it involves and the best linear hull over one such S-box.

Lemma 41 (Linear Approximation of S-box Layer). *Given an S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and an input/output mask $\alpha, \beta \in \mathbb{F}_2^{kn}$. Write $\alpha = (\alpha_1, \dots, \alpha_k)$ and β analogously, for $\alpha_i, \beta_i \in \mathbb{F}_2^n$. The linear approximation $\alpha \to^{S^{\|k}} \beta$ has the Walsh coefficient*

$$\widehat{S^{\|k}}(\alpha, \beta) = \prod_{i=1}^{k} \widehat{S}(\alpha_i, \beta_i)\,.$$

Proof. The masks split into the parallel application of S and the corresponding $\alpha_i \to^{S} \beta_i$ which are independent of each other. ∎

Lemma 42 (Linear Approximation of Linear Layer). *Given a linear function $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and an input/output mask $\alpha, \beta \in \mathbb{F}_2^n$. The linear approximation $\alpha \to^{L} \beta$ has the Walsh coefficient*

$$\widehat{L}(\alpha, \beta) = \begin{cases} 2^n & \text{if } \alpha = L^\top(\beta) \\ 0 & \text{else} \end{cases}\,.$$

Proof. This observation follows from Eq. (2.2). ∎

Analogously to the differential case, we define the activity of an S-box as in Definition 28, but with masks instead of differences. Eventually, we compute the bound as follows – again similar as for the differential case.

Lemma 43 (Bound on the linear trail correlation). *Given an r-round SPN cipher $E : \mathbb{F}_2^{kn} \to \mathbb{F}_2^{kn}$ with S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and linear layer $L : \mathbb{F}_2^{kn} \to \mathbb{F}_2^{kn}$. The correlation of any linear trail $\theta_0 \to^{L \circ S^{\|k}} \cdots \to^{L \circ S^{\|k}} \theta_r$ is bounded by*

$$|\mathbf{cor}_E(\theta_0, \dots, \theta_r)| \leqslant |c_S|^{\#\text{active S-boxes}}\,,$$

where $|c_S|$ is the maximum absolute correlation over the S-box and

$$\#\textit{active S-boxes} = \sum_{i=0}^{r-1} \mathrm{hw}(\mathrm{activity}(\theta_i))\,.$$

Proof. See the proof of Lemma 29. ∎

In conclusion, we get the heuristic for resistance against linear cryptanalysis which is used as the standard approach to Problem 4 (Provable Resistance). The required bound on $|c_S|$ is again derived from the data complexity of the attack, as the resulting distinguisher would need more than the full codebook to be evaluated.

Heuristic 44 (Heuristic Security against Linear Cryptanalysis)**.** *Given an r-round SPN cipher $E : \mathbb{F}_2^{kn} \to \mathbb{F}_2^{kn}$ with S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and linear layer $L : \mathbb{F}_2^{kn} \to \mathbb{F}_2^{kn}$. We assume the MELP$(E)$ is bounded by*

$$MELP(E) \leqslant |c_S|^{2\cdot\#\text{active S-boxes}} ,$$

where $|c_S|$ is the maximum absolute correlation over S and #active S-boxes is a lower bound for the number of active S-boxes through r rounds of E.

We claim E to be secure against linear cryptanalysis, if

$$|c_S|^{\#\text{active S-boxes}} < 2^{-kn/2} .$$

2.3.4 The Wide Trail Strategy for Differential and Linear Cryptanalysis

With the proposal of the wide trail strategy by Daemen [Dae95] the foundation for today's standard argument for the resistance against differential and linear cryptanalysis was built. It crucially relies on the assumption that we can bound the differential trail probability (or linear trail potential) by the best probability over the S-box and the number of active S-boxes. The major advantage is that this strategy allows to choose the S-box and the linear layer basically independent of each other and thus allows to optimise both building blocks on their own.

[Dae95] Daemen "Cipher and Hash Function Design Strategies based on linear and differential cryptanalysis"

We discuss the corresponding properties of S-boxes in the next Section 2.4 on Boolean functions and concentrate here on the linear layer. In the wide trail strategy, the task of the linear layer is to activate many S-boxes, or in other words to provide diffusion. To actually compute such a lower bound on the number of active S-boxes, different approaches exist today. We briefly discuss two of these: the initial approach by Daemen [Dae95] and an approach using *Mixed Integer Linear Programs (MILPs)*, first used by Mouha et al. [Mou+11]. Other notable ways are 1. using the so-called long trail strategy by Dinu et al. [Din+16] which is an analogue for ARX-based ciphers, see also [Per17, Chapter 6], or 2. an algorithm given by Matsui [Mat95, Section 4], which can be more efficient than MILPs but is not as versatile regarding different types of cryptanalysis, e. g. related-key attacks.

[Mou+11] Mouha et al. "Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming"

[Din+16] Dinu et al. "Design Strategies for ARX with Provable Bounds: Sparx and LAX"

[Per17] Perrin, "Cryptanalysis, Reverse-Engineering and Design of Symmetric Cryptographic Algorithms"

[Mat95] Matsui "On Correlation Between the Order of S-boxes and the Strength of DES"

▶ Wide Trail Strategy

Daemen's main idea to achieve a high number of active S-boxes is to align the linear layer to the S-box size, i. e. to work on $\left(\mathbb{F}_2^n\right)^k$ where the S-box permutes $\mathbb{F}_2^n$. He then defines the branch number of a linear layer, to model the S-boxes' activity propagation through the linear layer.

Definition 45 (Differential and Linear Branch Number)**.** Given a binary matrix $B \in \mathbb{F}_2^{nk\times nk}$, the *branch number for k-bit words* is defined using the k-bit word Hamming weight:

$$\mathrm{bn}_n(B) := \min_{u\in\mathbb{F}_2^{nk}\setminus\{0\}} \{\mathrm{hw}_n(u) + \mathrm{hw}_n(Mu)\} .$$

Further given a linear $L : \mathbb{F}_2^{nk} \to \mathbb{F}_2^{nk}$ and M_L, its corresponding matrix representation. The *differential branch number* of L is

$$\mathrm{bn}^D(L) := \min_{x \in \mathbb{F}_2^{nk} \setminus \{0\}} \{\mathrm{hw}_n(x) + \mathrm{hw}_n(L(x))\} = \mathrm{bn}_n(M_L)\,.$$

The *linear branch number* of L is

$$\mathrm{bn}^L(L) := \min_{x \in \mathbb{F}_2^{nk} \setminus \{0\}} \left\{\mathrm{hw}_n(x) + \mathrm{hw}_n(L^\top(x))\right\} = \mathrm{bn}_n(M_L^\top)\,.$$

This notion of branch number is linked to the minimum distance of a linear code, see [Rij+96]. This implies that optimal linear layers with respect to the branch number are MDS, achieving a branch number of $k+1$. Finding efficiently implementable MDS matrices is thus an active area of research, see also Problem 7 and Chapter 5.

[Rij+96] Rijmen et al., “The Cipher SHARK”

The branch number then gives a first lower bound on the number of active S-boxes, see [DR02, Theorem 9.3.1].

Proposition 46 (Two-Round Propagation Theorem)**.** *Given a key-alternating SPN $E : \mathbb{F}_2^{kn} \to \mathbb{F}_2^{kn}$ with S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and linear layer $L : \mathbb{F}_2^{kn} \to \mathbb{F}_2^{kn}$. Then the number of active S-boxes for any differential (linear) trail over two rounds is lower bounded by the differential (linear) branch number of L, i. e.*

$$\#\textit{differential active S-boxes} \geqslant \mathrm{bn}^D(L)$$

respective

$$\#\textit{linear active S-boxes} \geqslant \mathrm{bn}^L(L)\,.$$

Proof. The statement follows from the definition of the differential (linear) branch number.

A first bound on the number of active S-boxes is thus $r/2 \cdot \mathrm{bn}^D(L)$.

Another idea of Daemen is to split the linear layer in two parts, a mixing and a transposition step. Using the interaction of the mixing and transposition step and the Four-Round Propagation Theorem, [DR02, Theorem 9.4.1], the lower bound for the active S-boxes can then be further refined (to $r/4 \cdot \mathrm{bn}^D(L)^2$). However, we discuss an alternative approach next.

[DR02] Daemen and Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*

▸ Counting active S-boxes with MILPs

Since the proposal by Mouha et al. [Mou+11], MILPs received considerable attention and are used for the analysis of several ciphers, to name only a few see [Bil+13; Bei+16b; Köl+16; Ava17; Alf+18; Bei+19; LW19]. An advantage of this technique is that it can also be applied in the related-key setting or used for other types of cryptanalysis.

A Mixed Integer Linear Program (MILP) models the optimisation problem

$$\min_x c^\top x$$

for some coefficient vector c, where a valid solution for this objective function has to fulfill linear constraints of the form

$$A \cdot x \leqslant b\,.$$

Here, A is in general a real valued matrix and b a corresponding real valued vector. Further, the variables x and optional additional intermediate variables might be real valued, too, or restricted to integer or binary valued (hence the name *mixed integer* linear program).

During the analysis the cryptographer now models the activity propagation of S-boxes through the single rounds of the encryption. The most basic model for this uses the branch number of the linear layer, more advanced models can describe the exact linear layer operations and thus give tighter bounds when non-MDS linear layers are used.

For a detailed description, we refer the reader to [Mou+11] which covers as an educational example the MILP for active S-boxes in the AES.

[Mou+11] Mouha et al., “Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming”

Returning to cryptanalysis techniques, the above two attacks are the most basic and common approaches used to study the security of any modern block cipher. Several advanced methods based on differential and linear cryptanalysis where developed. In the remainder of this section, we cover a combined attack, differential-linear cryptanalysis, and a generalisation of differential cryptanalysis: truncated differentials and subspace trails.

2.3.5 *Differential-Linear Cryptanalysis*

For standard differential or linear cryptanalysis the attacker searches for a distinguisher targeting as many rounds as possible of the block cipher. In 1994, Langford and Hellman [LH94] published the idea to split the encryption function in two parts instead: $E = E^{\top} \circ E^{\perp}$, with r rounds in the first and s rounds in the second part. For these shorter parts we then exploit first a differential distinguisher ($\Delta \rightrightarrows^{E^{\perp}} \nabla$) and a linear distinguisher ($\lambda \rightarrow^{E^{\top}} \gamma$) on the second part. The crucial observation is how these two parts can be merged together.

[LH94] Langford and Hellman “Differential-Linear Cryptanalysis”

► Differential-Linear Distinguisher

In the case Langford and Hellman described, the differential is deterministic and will thus result in a specific difference ∇ after the r rounds of $E^{\perp}$ with certainty. If the linear mask λ fits to ∇ in a way that λ is only active in positions where ∇ is passive, we can deduce the following. Both input pairs x and $x' = x + \Delta$ will be equal in the parts masked by λ after r rounds with probability one (denote these intermediate values $y = E^{\perp}(x)$ and $y' = E^{\perp}(x')$). If both follow the linear characteristic for the s rounds of $E^{\top}$, they will behave the same in the parts masked by γ.

The natural generalisation of this approach is to allow for probabilistic differentials, as these are anyway the standard in differential cryptanalysis. This generalisation was later done by Langford [Lan95] in her PhD thesis and by Biham et al. [Bih+02], i. e. to allow for differentials with lower probability. In the following, we directly give the general approach.

[Lan95] Langford “Differential-Linear Cryptanalysis and Threshold Signatures”

[Bih+02] Biham et al. “Enhancing Differential-Linear Cryptanalysis”

Formally, with

$$\varepsilon^{L}_{E^{\top}}(\lambda,\gamma) = \mathrm{Pr}_x\left[\lambda \xrightarrow[x]{E^{\top}} \gamma\right] - \frac{1}{2} = \mathrm{Pr}_x\left[\lambda \cdot x = \gamma \cdot E^{\top}(x)\right] - \frac{1}{2} = q$$

and the input/output pairs (x, y, z), (x', y', z') over r and s rounds, we have

$$\mathrm{Pr}[\lambda \cdot y = \gamma \cdot z] = \frac{1}{2} + q \quad \text{and} \quad \mathrm{Pr}[\lambda \cdot y' = \gamma \cdot z'] = \frac{1}{2} + q$$

and thus with the Piling-Up lemma

$$\Pr\left[\lambda \cdot (y + y') = \gamma \cdot (z + z')\right] = \frac{1}{2} + 2q^2.$$

While we would not know the intermediate values y and y' for an actual cipher, we can deduce the bias of the differential-linear propagation by exploiting the differential $\Delta \rightrightarrows^{E^\perp} \nabla$ as

$$\varepsilon^{DL}_{E^\top \circ E^\perp}(\Delta, \gamma) = \Pr\left[\lambda \cdot \nabla = \gamma \cdot (z + z')\right] - \frac{1}{2} = 4pq^2 \tag{2.9}$$

where the bias of the differential is

$$\varepsilon^{D}_{E^\perp}(\Delta, \nabla) = \Pr_x\left[\Delta \overset{E^\perp}{\underset{x}{\rightrightarrows}} \nabla\right] - \frac{1}{2} = \Pr_x\left[E^\perp(x) + E^\perp(x + \Delta) = \nabla\right] - \frac{1}{2} = p.$$

Note that these biases are only estimations, as effects like the linear hull occur. The data complexity in this case is assumed to be $\mathcal{O}(p^{-2}q^{-4})$.

The theory behind this analysis was later formalised by Blondeau et al. [Blo+15; Blo+17]. They also introduced the differential-linear hull in order to compute the bias $\varepsilon^{DL}_{E}(\Delta, \gamma)$ which basically equals the sum over all biases where we iterate over the possible connections (∇, λ), see [Blo+15, Sections 3 and 4] for more details. For this, Blondeau et al. assumed the two parts $E^\perp$, and $E^\top$ to be independent.

[Blo+15] Blondeau et al. "Differential-Linear Cryptanalysis Revisited"

[Blo+17] Blondeau et al. "Differential-Linear Cryptanalysis Revisited"

▸ Differential-Linear Connectivity Table

Bar-On et al. [Bar+19] showed that the above independence assumption might fail in practice, in particular for the differential-linear attack on Ascon, which was published by Dobraunig et al. [Dob+15] and the initial analysis by Dobraunig et al. showed a discrepancy of a factor of 2^{-18} between the theoretical and the experimental differential-linear bias. Bar-On et al. state the differential-linear bias more precisely by introducing a middle layer $E^{\top\!\!\!\!\perp}$.[5] We thus have $E = E^\top \circ E^{\top\!\!\!\!\perp} \circ E^\perp$ and the differential-linear approximation splits into a differential over $E^\perp$, a linear hull over $E^\top$, and a differential-linear hull over $E^{\top\!\!\!\!\perp}$ which is evaluated using the DLCT. Using this, we get for the differential-linear bias of E

$$\varepsilon^{DL}_{E}(\Delta, \gamma) = \sum_{\nabla, \lambda} 8 \cdot \varepsilon^{D}_{E^\perp}(\Delta, \nabla) \cdot \varepsilon^{DL}_{E^{\top\!\!\!\!\perp}}(\nabla, \lambda) \cdot \varepsilon^{L}_{E^\top}(\lambda, \gamma)^2,$$

where $\varepsilon^{DL}_{E^{\top\!\!\!\!\perp}}(\nabla, \lambda)$ is computed with the help of the DLCT, see [Bar+19, Eq. (5)].

[Bar+19] Bar-On et al. "DLCT: A New Tool for Differential-Linear Cryptanalysis"

[Dob+15] Dobraunig et al. "Cryptanalysis of Ascon"

Finally, as far as we can tell, and we consider it worth to highlight it again, Blondeau et al. [Blo+15] were also the first who explicitly defined the bias of a differential-linear approximation as $\varepsilon^{DL}_{E}(\Delta, \gamma)$. Note that this is exactly what Bar-On et al. defined to be the $\overline{\mathrm{DLCT}}_E(\Delta, \gamma)$. Thus the DLCT captures exactly the property we want to exploit in a differential-linear attack. Only as it is computationally infeasible to directly compute the DLCT of the whole cipher, we fall back to compute it for smaller parts and try to approximate the entire one with it. The work by Bar-On et al. can thus be seen as a contribution towards a solution for Problem 4 for differential-linear attacks. However, this new notion still lacks a thorough theoretical analysis. We thus revisit it in the following Chapter 3, hoping to further contribute to a better understanding of the differential-linear cryptanalysis.

[5] This approach is similar to what was done for boomerang attacks.

2.3.6 *Truncated Differentials and Subspace Trails*

The last cryptanalytic techniques we want to cover here, are truncated differentials and subspace trails. We discuss the relationship between both in more detail in Chapter 6, along with approaches to show the resistance against their probability-one cases.

▸ Truncated Differentials

[Knu95] Knudsen “Truncated and Higher Order Differentials”

Knudsen [Knu95] proposed truncated differentials together with higher order differentials as generalisations of differential cryptanalysis. Subsequently, many ciphers where analysed using this technique, see e. g. [KB96; Bor+97; Knu+99; LN15; Bir+15].

The main idea for truncated differentials is not to use only one differential $\alpha \rightrightarrows \beta$, but instead several input and output differences. Formally we generalise the differences to affine subspaces of $\mathbb{F}_2^n$, following the notation of Blondeau et al. [Blo+15].

[Blo+15] Blondeau et al. “Differential-Linear Cryptanalysis Revisited”

Definition 47 (Truncated Differential)**.** Given a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and two affine subspaces $U+s, V+t$ of $\mathbb{F}_2^n$. Then

$$U+s \overset{F}{\rightrightarrows} V+t$$

is a *truncated differential*. The truncated differential holds if any of the contained differentials hold:

$$\Pr_x\left[U+s \underset{x}{\overset{F}{\rightrightarrows}} V+t\right] = \Pr_x\left[\bigvee_{u\in U, v\in V} u+s \underset{x}{\overset{F}{\rightrightarrows}} v+t\right].$$

Actually, this notion could be further generalised to subsets instead of affine subspaces.[6] However, omitting the structure (affine) subspaces provide makes it much harder to handle differentials algorithmically and we thus stick to the notation using affine subspaces.

Along the line of trails in the differential and linear cryptanalysis cases, we can define a truncated differential trail.

Definition 48 (Truncated Differential Trail)**.** Given a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $r+1$ affine subspaces U_i+s_i of $\mathbb{F}_2^n$. The *truncated differential trail* over r rounds consists of r truncated differentials over each round:

$$U_0+s_0 \overset{F}{\rightrightarrows} \cdots \overset{F}{\rightrightarrows} U_r+s_r = \bigwedge_{i=i}^{r} U_{i-1}+s_{i-1} \overset{F}{\rightrightarrows} U_i+s_i\ .$$

▸ Subspace Trails

[Gra+16] Grassi et al. “Subspace Trail Cryptanalysis and its Applications to AES”

Grassi et al. [Gra+16] introduced subspace trail cryptanalysis as a generalisation of invariant subspaces and truncated differentials.

Given a round function F over some space, a subspace trail is basically a tuple of subspaces whose cosets are mapped through this round function. More formally:

Definition 49 (Subspace Trail)**.** Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Linear subspaces $U, V \subseteq \mathbb{F}_2^n$ are called a *(one round) subspace trail*, if

$$\forall a : \exists b : F(U+a) \subseteq V+b$$

[6] For example, then any impossible differential of the form “an input difference α cannot lead to an output difference β” can be written as a probability-one truncated differential, namely $\alpha \rightrightarrows \mathbb{F}_2^n \setminus \{\beta\}$.

We denote this by $U \overset{F}{\rightrightarrows} V$. We write $U \overset{F}{\not\rightrightarrows} V$, if they do not form a subspace trail, i. e.

$$\exists a : \forall b : F(U + a) \not\subseteq V + b$$

An $r+1$-tuple of subspaces $(U_1, \dots, U_{r+1})$ is called a *subspace trail* (over r rounds), if

$$U_i \overset{F}{\rightrightarrows} U_{i+1} \quad \forall i \in \{1, \dots, r+1\}.$$

The $\rightrightarrows$ notion is overloaded on purpose. Normal differentials, truncated differentials or subspace trails can be distinguished by single elements, cosets or linear subspaces, thus the exact meaning is clear from the context. We defer all further discussion on truncated differentials and subspace trails to Chapter 6.

2.4 BOOLEAN FUNCTIONS

Boolean functions $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$ and vectorial Boolean functions $F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$ are of prime interest in the design and analysis of block ciphers, as they help to understand security properties of building blocks such as S-boxes. They are therefor well studied, see e. g. Carlet's survey of known results [Car10a; Car10b].

[Car10a] Carlet, "Boolean Functions for Cryptography and Error-Correcting Codes"

[Car10b] Carlet, "Vectorial Boolean Functions for Cryptography"

Boolean functions can be represented in several ways, the two most important for us are listed below. Given a Boolean function f in n variables, f can be represented by

Algebraic Normal Form (ANF) a multivariate polynomial

$$p \in \mathbb{F}_2[x_1, \dots, x_n]/(x_i^2 + x_i)_{i=1}^n ,$$

such that $f(x) = p(x)$: for $u \in \mathbb{F}_2^n$ we denote

$$x^u = \prod_{i=1}^{n} x_i^{u_i} ,$$

for suitable choices of $\lambda_u \in \mathbb{F}_2$ the ANF of f can then be expressed as

$$f(x) = \sum_{u \in \mathbb{F}_2^n} \lambda_u x^u ;$$

Truth Table or value vector $v \in \mathbb{F}_2^{2^n}$, such that $f(x)$ equals v_x (where x is interpreted as an integer),

For notational brevity, we may give the value vector in hexadecimal notation.

Example 50. The Boolean function $f : \mathbb{F}_2^3 \rightarrow \mathbb{F}_2$ is specified by

x	0x0	0x1	0x0	0x0	0x0	0x0	0x0	0x0
$f(x)$	0	1	0	0	1	0	0	1

or equivalently 92. Its ANF is $f(x_1, x_2, x_3) = x_1x_2x_3 + x_1x_2 + x_2x_3 + x_1 + x_3$.

A vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ consists of m coordinates or 2^m components, i. e. Boolean functions in n variables, denoted by $F_\beta(x) = \beta \cdot F(x)$, that is

$$F(x_1, \dots, x_n) = \begin{pmatrix} F_{e_1}(x_1, \dots, x_n) \\ \vdots \\ F_{e_n}(x_1, \dots, x_n) \end{pmatrix} .$$

We denote the support of f by $\operatorname{Supp}(f) := \left\{ x \in \mathbb{F}_2^n \;\middle|\; f(x) \neq 0 \right\}$. The degree of f, denoted by $\deg(f)$, is defined as the maximal weight of a monomial present in the ANF of f, i. e.

$$\deg(f) = \max \left\{ \operatorname{wt}(u) \;\middle|\; u \in \mathbb{F}_2^n \text{ such that } \lambda_u \neq 0 \right\} .$$

For the constant zero function, the degree is $\deg(0) = 0$.

In the remainder of this section, we cover 1. derivatives and the uniformity of (vectorial) Boolean functions which play an important role in the analysis of differential properties, 2. linear structures, special types of derivatives that are essential for our study of subspace trails, and 3. the Walsh transform and the linearity, essential for linear cryptanalysis. We conclude the discussion of Boolean functions with some observations on bent functions.

2.4.1 *Derivatives and Linear Structures*

Differential cryptanalysis studies the probability

$$\Pr_x \left[\alpha \overset{F}{\underset{x}{\rightrightarrows}} \beta \right] = \Pr_x \left[F(x) + F(x + \alpha) = \beta \right] . \tag{2.10}$$

This leads to the definition of derivatives for (vectorial) Boolean functions.

Definition 51 (Derivative). Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ The *derivative of f in direction* α is defined as

$$\begin{aligned} \Delta : \mathbb{F}_2^n \times \left(\mathbb{F}_2^n \to \mathbb{F}_2 \right) &\to \left(\mathbb{F}_2^n \to \mathbb{F}_2 \right) \\ \Delta_\alpha(f)(x) &:= f(x) + f(x + \alpha). \end{aligned}$$

If we generalise to the derivative of a vectorial Boolean function, we can additionally specify an output difference β. The DDT captures the distribution of all possible derivatives.

Definition 52 (Difference Distribution Table). Given a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$. The *Difference Distribution Table (DDT)* consists of the entries

$$\mathrm{DDT}_F[\alpha, \beta] := \left| \Delta_\alpha(F)^{-1}(\beta) \right| = \left| \left\{ x \in \mathbb{F}_2^n \;\middle|\; \Delta_\alpha(F)(x) = \beta \right\} \right| ,$$

where we leave out the subscript, if it is clear from the context.

Using the DDT, Eq. (2.10) becomes

$$\Pr_x \left[\alpha \overset{F}{\underset{x}{\rightrightarrows}} \beta \right] = \frac{\mathrm{DDT}_F[\alpha, \beta]}{2^n} .$$

Some basic observations on the DDT and derivatives are the following. Every entry in the DDT is even, as $\mathbb{F}_2^n$ has characteristic two and thus $\Delta_\alpha(F)(x) = \Delta_\alpha(F)(x+\alpha)$ for all α and x. The sum over any row α of the DDT is $\sum_{\beta \in \mathbb{F}_2^n} \mathrm{DDT}_F[\alpha, \beta] = 2^n$, as we sum up all possible output differences. Similarly, the sum over any column of the DDT equals 2^n if F is a permutation.

The main security property an S-box S should exhibit against differential cryptanalysis, is a low maximal probability for any differential over it, denoted p_S. We relate p_S to the DDT's maximum entry.

Definition 53 (Uniformity)**.** Given a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$. The *uniformity* of F is

$$\delta(F) := \max_{\substack{\alpha \in \mathbb{F}_2^n \setminus \{0\} \\ \beta \in \mathbb{F}_2^m}} \mathrm{DDT}_F[\alpha, \beta] = \max_{\substack{\alpha \in \mathbb{F}_2^n \setminus \{0\} \\ \beta \in \mathbb{F}_2^m}} \left| \Delta_\alpha(F)^{-1}(\beta) \right| .$$

We exclude the input difference $\alpha = 0$, as this is the trivial case giving no information. The uniformity for vectorial Boolean Functions is bound by $2 \leqslant \delta(F) \leqslant 2^n$, due to the above discussed facts. Functions with uniformity reaching the lower bound are called *Almost Perfect Nonlinear (APN)* (perfect nonlinear functions $F : A \to B$ achieve uniformity $|B|/|A|$, but do not exist for vectorial Boolean functions).

Linear functions obviously have uniformity 2^n.

Using the uniformity results in $p_S = \delta(S)/2^n$ for Heuristic 30.

A special case of derivatives are the linear structures of a (vectorial) Boolean function.

2.4.2 *Linear Structures*

Linear structures were studied in e. g. [Eve88; Lai95; CV05], and quite recently, see [Bei+17], appeared in the context of resistance against invariant attacks; in particular the linear structures of the invariants are studied.

[Bei+17] Beierle et al., "Proving Resistance Against Invariant Attacks: How to Choose the Round Constants"

Definition 54 (Linear Structures of Boolean functions)**.** Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$. The set of *linear structures of f* is

$$\mathrm{LS}(f) = \left\{ \alpha \in \mathbb{F}_2^n \,\middle|\, \exists c \in \mathbb{F}_2, \forall x \in \mathbb{F}_2^n : \Delta_\alpha(f)(x) = c \right\}.$$

We call $\alpha \in \mathrm{LS}(f)$ for which the derivative is constant zero (or one) a *0-linear structure* (or *1-linear structure*).

$\mathrm{LS}(f)$ is also called the linear space of f and it is partitioned by the above distinction into $\mathrm{LS}(f) = \mathrm{LS}^0(f) \cup \mathrm{LS}^1(f)$, where $\mathrm{LS}^1(f)$ is either empty or a coset of $\mathrm{LS}^0(f)$ (we have for any $\alpha \in \mathrm{LS}^0(f)$ and $\beta \in \mathrm{LS}^1(f)$ that $\alpha + \beta \in \mathrm{LS}^1(f)$). Note that $\mathrm{LS}(f)$ and $\mathrm{LS}^0(f)$ are subspaces.

Lemma 55 (Linear Structure subspaces)**.** *Given $f : \mathbb{F}_2^n \to \mathbb{F}_2$. Then $\mathrm{LS}(f)$ and $\mathrm{LS}^0(f)$ are both subspaces of $\mathbb{F}_2^n$.*

Proof. The trivial 0-linear structure $\alpha = 0$ is contained in $\mathrm{LS}(f)$ and $\mathrm{LS}^0(f)$.

Now assume, $\gamma, \theta \in \mathrm{LS}(f)$. Then $\Delta_\gamma(f)(x) + \Delta_\theta(f)(x)$ is constant and so is $f(x) + f(x+\gamma) + f(x) + f(x+\theta)$. By substituting $x' = x + \gamma$ this implies $f(x') + f(x' + \gamma + \theta)$ is constant and thus $\gamma + \theta \in \mathrm{LS}(f)$.

The same argument holds for $\gamma, \theta \in \mathrm{LS}^0(f)$. ∎

This definition can be naturally extended to vectorial Boolean functions.

Definition 56 (Linear Structures of vectorial Boolean functions)**.** Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$. The set of *linear structures of* F is

$$\mathrm{LS}(F) := \left\{(\alpha,\beta) \in \mathbb{F}_2^m \times \mathbb{F}_2^n \;\middle|\; \exists c \in \mathbb{F}_2, \forall x \in \mathbb{F}_2^n : \alpha \cdot \Delta_\beta(F)(x) = c\right\} \quad (2.11)$$

We define the set L_β as

$$\mathrm{L}_\beta(F) := \left\{\alpha \in \mathbb{F}_2^m \;\middle|\; \beta \in \mathrm{LS}(F_\alpha)\right\},$$

thus the set of all components F_α that have β as a linear structure. Just like $\mathrm{LS}(f)$, we can partition $\mathrm{L}_\beta(F) = \mathrm{L}_\beta^0(F) \cup \mathrm{L}_\beta^1(F)$ by fixing the constant c in Eq. (2.11) to either 0 or 1, and where the later is either empty or a coset of the first. Finally, like with $\mathrm{LS}(f)$, linear structures with constant $c = 0$ are called *0-linear structures*, and *1-linear structures* if $c = 1$.

Lemma 57 (0-Linear Structures are subspaces)**.** *Given* $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ *and* $\beta \in \mathbb{F}_2^n$. *Then* $\mathrm{L}_\beta^0(F)$ *is a subspace of* $\mathbb{F}_2^n$.

Proof. As $\alpha = 0$ is a trivial 0-linear structure for any F, it is contained in $\mathrm{L}_\beta^0(F)$. For all $\gamma, \theta \in \mathrm{L}_\beta^0(F)$:

$$\forall x : (\gamma + \theta) \cdot \Delta_\beta(F)(x) = \gamma \cdot \Delta_\beta(F)(x) + \theta \cdot \Delta_\beta(F)(x) = 0$$

and thus $\gamma + \theta \in \mathrm{L}_\beta^0(F)$. ∎

Let us now turn to the linear properties of (vectorial) Boolean functions.

2.4.3 *Walsh transform*

In Section 2.3.3 we have already discussed the relation between a linear approximation over F and the Walsh transform of F. Definition 32 gave the definition of the Walsh transform for vectorial Boolean functions. The special case of the Walsh transform of a Boolean function is the following.

Definition 58 (Walsh transform)**.** Given a Boolean function $f \in \mathrm{BF}_n$. Its *Walsh transform* at point α is defined as

$$\widehat{f}(\alpha) := \sum_{x \in \mathbb{F}_2^n} (-1)^{\alpha \cdot x + f(x)}.$$

Thus the Walsh transform of vectorial Boolean functions consists of the Walsh transforms of its components. We encounter this generalisation several times, e. g. in the following and in Chapter 3.

Analogously to the DDT, we define the LAT.

Definition 59. Given a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$. The *Linear Approximation Table (LAT)* contains the Walsh coefficients

$$\mathrm{LAT}_F[\alpha,\beta] := \widehat{F}(\alpha,\beta).$$

Again we leave out the subscript, if it is clear from the context.

A fundamental equality is the following.

Lemma 60 (Parseval's equality)**.** *Given a Boolean function* $f : \mathbb{F}_2^n \to \mathbb{F}_2$*. Then*

$$\sum_{\alpha \in \mathbb{F}_2^n} \widehat{f}(\alpha)^2 = 2^{2n} .$$

Proof. Reordering the sum results in

$$\sum_{\alpha \in \mathbb{F}_2^n} \widehat{f}(\alpha)^2 = \sum_{\alpha, x, y \in \mathbb{F}_2^n} (-1)^{\alpha \cdot x + f(x) + \alpha \cdot y + f(y)} = \sum_{x,y} (-1)^{f(x)+f(y)} \sum_{\alpha} (-1)^{\alpha \cdot (x+y)}$$

and applying Eq. (2.1) again, in

$$= 2^n \sum_{x=y} (-1)^{f(x)+f(x)} = 2^{2n}$$

as claimed. ∎

The main property of a (vectorial) Boolean function to resist linear cryptanalysis is its (non-) linearity.

Definition 61 (Linearity and Nonlinearty)**.** Given a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$. The *linearity* of f is

$$\mathcal{L}(f) := \max_{\alpha \in \mathbb{F}_2^n} \left| \widehat{f}(\alpha) \right| ,$$

and the *nonlinearity* is $\mathcal{NL}(f) := 2^n - \mathcal{L}(f)$.

For a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, the linearity is again defined over all components, except the trivial F_0 component which is constant zero:

$$\mathcal{L}(F) := \max_{\beta \in \mathbb{F}_2^m \setminus \{0\}} \mathcal{L}\left(F_\beta\right) .$$

Typically the nonlinearity is defined as $\mathcal{NL}(f) := 2^{n-1} - \frac{1}{2}\mathcal{L}(f)$. In the remainder of this thesis, we only work with the linearity.

The nonlinearity can also be interpreted as the minimal Hamming-distance of F to the set of all affine functions, because for these the following result holds.

Lemma 62 (Walsh coefficient of affine functions)**.** *Given an affine Boolean function* $f : \mathbb{F}_2^n \to \mathbb{F}_2, x \mapsto \beta \cdot x + c$*. Then*

$$\widehat{f}(\alpha) = \begin{cases} (-1)^c 2^n & \text{if } \alpha = \beta \\ 0 & \text{else} \end{cases} .$$

Proof. We can simply compute

$$\widehat{f}(\alpha) = \sum_{x \in \mathbb{F}_2^n} (-1)^{\alpha \cdot x + f(x)} = \sum_{x \in \mathbb{F}_2^n} (-1)^{\alpha \cdot x + \beta \cdot x + c} = (-1)^c \sum_{x \in \mathbb{F}_2^n} (-1)^{(\alpha+\beta) \cdot x}$$

Using the fundamental fact Eq. (2.1), the statement follows. ∎

With the linearity, we can compute $|c_S| = \mathcal{L}(S)/2^n$ for Heuristic 44.

Using Parseval's equality we can deduce a lower bound on the linearity.

Proposition 63 (Covering Radius Bound)**.** *Given a Boolean function* $f : \mathbb{F}_2^n \to \mathbb{F}_2$*. Then* $\mathcal{NL}(f) \leqslant 2^n - 2^{n/2}$ *or equivalently* $\mathcal{L}(f) \geqslant 2^{n/2}$.

Proof. Assume for all $\alpha \in \mathbb{F}_2^n$ it holds that $\widehat{f}(\alpha) < 2^{n/2}$. Then this would imply $\sum_\alpha \widehat{f}(\alpha)^2 < 2^{2n}$, contradicting Lemma 60. ∎

[Car10a] Carlet, "Boolean Functions for Cryptography and Error-Correcting Codes"

This bound is also called *Covering Radius* bound, as it can be related to the covering radius of the Reed-Muller code of order one if n is even, see e. g. [Car10a, p. 297]. Functions reaching this lower bound are called *bent*, we discuss them at the end of this section. Additionally, this bound can only be reached by functions in n variables with n even.

For n odd, we can bound the lowest linearity. We use the following lemma for the proof.

Lemma 64 (Walsh coefficients of direct sums). *Let $f : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2$ be of the form $f(x, y) = g(x) + h(y)$. Then for f's Walsh coefficients and linearity it holds that*

$$\widehat{f}(\alpha, \beta) = \widehat{g}(\alpha) \cdot \widehat{h}(\beta), \quad \textit{and} \quad \mathcal{L}(f) = \mathcal{L}(g) \cdot \mathcal{L}(h).$$

Proof. We can simply compute f's Walsh transform as

$$\begin{aligned}\widehat{f}(\alpha, \beta) &= \sum_{x \in \mathbb{F}_2^n, y \in \mathbb{F}_2^m} (-1)^{\alpha \cdot x + \beta \cdot y + f(x,y)} \\ &= \sum_{x,y} (-1)^{\alpha \cdot x + g(x) + \beta \cdot y + h(y)} = \widehat{g}(\alpha) \cdot \widehat{h}(\beta).\end{aligned}$$

Noting that

$$\max_{\alpha,\beta} \left|\widehat{f}(\alpha, \beta)\right| = \max_\alpha \left|\widehat{g}(\alpha)\right| \cdot \max_\beta \left|\widehat{h}(\beta)\right|$$

concludes the proof. ∎

Proposition 65 (Lowest possible linearity). *The lowest possible linearity of a Boolean function in m variables, $m = 2n + 1$ odd, is bounded by*

$$2^{m/2} < \min_{f \in \mathrm{BF}_m} \mathcal{L}(f) \leqslant 2^{\frac{m+1}{2}}.$$

Proof. As we see below in Lemma 66, the inner product $x \cdot y$ over $\mathbb{F}_2^{2n}$ is bent. From this, We construct the following function over $\mathbb{F}_2^m$ with $m = 2n + 1$:

$$\begin{aligned} f(x, y, z) &: \mathbb{F}_2^n \times \mathbb{F}_2^n \times \mathbb{F}_2 \to \mathbb{F}_2 \\ f(x, y, z) &= x \cdot y + z.\end{aligned}$$

Using Lemma 64, the linearity of f is given by

$$\mathcal{L}(f) = \mathcal{L}(g) \cdot \mathcal{L}(h) = 2^n \cdot 2 = 2^{n+1} = 2^{\frac{m+1}{2}},$$

as claimed. ∎

[Nyb91] Nyberg "Perfect Nonlinear S-Boxes"

Nyberg [Nyb91] showed that a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ can only be bent if $n \geqslant 2m$. Such functions are also called *perfect nonlinear*. As we are most often interested in permutations of $\mathbb{F}_2^n$, we cannot use bent functions in this case. Instead, the best linearity we can then achieve (i. e. for $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$) is

$$\mathcal{L}(F) \geqslant 2^{\frac{n+1}{2}},$$

where functions reaching this bound are called *Almost Bent (AB)*. Obviously AB functions only exist for n odd. This bound is also called *Sidelnikov-Chabaud-Vaudenay* bound, as it was first shown by Sidelnikov [Sid71] and later by Chabaud and Vaudenay [CV95].

[Sid71] Sidelnikov "On the mutual correlation of sequences"

[CV95] Chabaud and Vaudenay "Links Between Differential and Linear Cryptanalysis"

2.4.4 *Bent Functions*

Bent functions have been introduced by Rothaus [Rot76] and studied ever since, see also [Car10a, Section 8.6].

[Rot76] Rothaus "On 'bent' functions"

Using Parseval's equality and following from the Covering Radius bound, it is easy to see that a function is bent if and only if all its Walsh coefficients are $\pm 2^{\frac{n}{2}}$. Moreover, an alternative classification that will be of importance in Chapter 4, is that a function is bent if and only if all (non-trivial) derivatives $\Delta_\alpha(f)$ are balanced Boolean functions [MS90].

[MS90] Meier and Staffelbach, "Nonlinearity Criteria for Cryptographic Functions"

While there are many different primary and secondary constructions[7] of bent functions known, for simplicity (and for the sake of ease of implementation), we focus on the simplest known bent function in Chapter 4. We recall this function next, see also [Car10a, Section 6.2].

Lemma 66 ([Dil72]). *Let $n = 2m$ be an even integer. The function*

$$\begin{aligned} f : \mathbb{F}_2^m \times \mathbb{F}_2^m &\to \mathbb{F}_2 \\ f(x, y) &:= x \cdot y \end{aligned}$$

corresponding to the canonical scalar product over $\mathbb{F}_2^m$, is a quadratic bent function.

In other words $\Delta_{(\alpha,\beta)}(f)$ is balanced for all $\alpha, \beta \in \mathbb{F}_2^m$.

Proof. By induction over m. For $m = 1$ we can simply compute the linearity of $f(x, y) = xy$ as $\mathcal{L}(f) = \widehat{f}(0) = \widehat{f}(1) = \widehat{f}(2) = -\widehat{f}(3) = 2$. Now, assume that $\mathcal{L}(f) = 2^m$ hold. For $m+1$, i. e. $n = 2m+2$, write

$$\begin{aligned} &f(x_1, \dots, x_m, x_{m+1}, y_1, \dots, y_m, y_{m+1}) \\ &= g(x_1, \dots, x_m, y_1, \dots, y_m) + h(x_{m+1}, y_{m+1}) \,. \end{aligned}$$

Applying Lemma 64 and the induction argument gives

$$\mathcal{L}(f) = \mathcal{L}(g) \cdot \mathcal{L}(h) = 2^m \cdot 2 = 2^{m+1} = 2^{n/2}$$

which implies f is bent.

Even though bent functions achieve the highest possible nonlinearity, their direct use in symmetric cryptography is so far very limited. This is mainly due to the fact that bent functions are not balanced, i. e. the distribution of zeros and ones is (slightly) biased, and is due to the requirement that $\widehat{f}(0) = 2^{n/2}$.

Tokareva [Tok15, Chapter 4] discusses some applications of bent functions in cryptography. Most notably, the S-boxes used in the block cipher CAST [Ada97] are designed such that all 32 coordinates are bent functions (in 8 variables). The stream cipher GRAIN [Hel+06] and the hash function HAVAL [Zhe+93] use nonlinear functions which are constructed from bent functions. However, e. g. using AB round functions in a block cipher might also come with caveats, see the discussion by Canteaut and Videau [CV02]. Interestingly, Tokareva discusses bent functions only to thwart linear cryptanalysis – whereas our application for bent functions in Chapter 4 exploits the fact that all derivatives of a bent function are balanced to show resistance against differential cryptanalysis.

[Tok15] Tokareva *Bent Functions*

[CV02] Canteaut and Videau "Degree of Composition of Highly Nonlinear Functions and Applications to Higher Order Differential Cryptanalysis"

[7] Primary constructions give bent functions from scratch, while secondary constructions build new bent functions from previously defined ones.

Part II

SECURITY ARGUMENTS FOR BLOCK CIPHERS

3

On Autocorrelation Tables and Differential-Linear Cryptanalysis

“Attacks only get better, never get worse.”
—Unknown

▸ SYNOPSIS We observe that the DLCT coincides with the ACT and provide a detailed analysis of it. This chapter is based on the preprint

> Anne Canteaut, Lukas Kölsch, and Friedrich Wiemer. *Observations on the DLCT and Absolute Indicators*. 2019. IACR PREPRINT: 2019/848.

All authors contributed equally.

After discussing the equivalence of DLCT and ACT, we analyse a link between the ACT, DDT and LAT, which corresponds to the well known relation between differential and linear cryptanalysis already exhibited by Chabaud and Vaudenay [CV95] and later by Blondeau and Nyberg [BN13]. We then show that the (extended) ACT spectrum is invariant under (extended) affine equivalence, but not under CCZ equivalence. The last observation is based on the fact that the ACT of a function and that of its inverse can behave quite differently. This is of particular interest, as it might lead to the case where a differential-linear attack work better for the decryption than the encryption of a block cipher. It is also in stark contrast to standard differential or linear cryptanalysis, which both behave the same in both directions.

[CV95] Chabaud and Vaudenay “Links Between Differential and Linear Cryptanalysis”

[BN13] Blondeau and Nyberg “New Links between Differential and Linear Cryptanalysis”

We extend our analysis of the ACT by exhibiting a lower bound on the absolute indicator for generic vectorial Boolean functions. Eventually we prove some more specific properties for some specific types of vectorial functions.

3.1 ON THE DLCT AND ACT

Let us define the DLCT as follows.

Definition 67 (Differential-Linear Connectivity Table)**.** Given a permutation $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, the corresponding *Differential-Linear Connectivity Table (DLCT)* consists of the following elements:

$$\mathrm{DLCT}_F[\alpha, \beta] := \sum_{x \in \mathbb{F}_2^n} (-1)^{\beta \cdot (F(x) + F(x+\alpha))} \tag{3.1}$$

We leave out the subscript, if it is clear from the context.

[BO+19] Bar-On et al. *DLCT: A New Tool for Differential-Linear Cryptanalysis*

While Bar-On et al. [BO+19] defined the entry at position (α, β) as

$$\mathrm{DLCT}_F[\alpha,\beta] = |\{x \mid \beta \cdot (F(x) + F(x+\alpha)) = 0\}| - 2^{n-1},$$

it is easy to see that both definitions only differ in a factor of 2 for each entry:

$$\begin{aligned} &2 \cdot \left(\left|\left\{x \in \mathbb{F}_2^n \mid \beta \cdot (F(x) + F(x+\alpha)) = 0\right\}\right| - 2^{n-1}\right) = |M_0| + |M_0| - 2^n \\ &= |M_0| + (2^n - |M_1|) - 2^n = |M_0| - |M_1| = \sum_{x \in \mathbb{F}_2^n} (-1)^{\beta \cdot (F(x) + F(x+\alpha))}, \end{aligned}$$

where we define $M_i = \left\{x \in \mathbb{F}_2^n \mid \beta \cdot (F(x) + F(x+\alpha)) = i\right\}$ for the sake of readability.

Our first observation on the DLCT is that it basically contains the autocorrelation spectra of the component functions of F. Recall that the *autocorrelation of f* is defined as, see e. g. Carlet [Car10a, p. 277],

[Car10a] Carlet "Boolean Functions for Cryptography and Error-Correcting Codes"

$$\mathcal{A}_f(\alpha) := \widehat{\Delta_\alpha(f)}(0).$$

Similar to Walsh coefficients, this notion can naturally be generalised to vectorial Boolean functions as

$$\mathcal{A}_F(\alpha,\beta) := \mathcal{A}_{F_\beta}(\alpha) = \widehat{\Delta_\alpha(F_\beta)}(0),$$

and we name the $\mathcal{A}_F(\alpha,\beta)$ the *autocorrelation coefficients* of F. In other words, the autocorrelation coefficients of a vectorial Boolean function consist of the autocorrelation of its component functions. To easily see the correspondence, we only have to conclude that

$$\widehat{\Delta_\alpha(F_\beta)}(0) = \sum_x (-1)^{\beta \cdot (F(x) + F(x+\alpha))}.$$

[Zha+00] Zhang et al. "Relating Differential Distribution Tables to Other Properties of of Substitution Boxes"

Zhang et al. [Zha+00, Section 3] introduced the term Autocorrelation Table (ACT) for a vectorial Boolean function which, analogously to the Walsh coefficient and LAT again, contains the autocorrelation spectra of F's component functions. This implies for the DLCT that

$$\mathrm{DLCT}_F[\alpha,\beta] = \mathcal{A}_F(\alpha,\beta) = \mathrm{ACT}_F[\alpha,\beta].$$

For the remainder of this paper we thus stick to the established notion of the autocorrelation.

[ZZ96] Zhang and Zheng "GAC — the Criterion for Global Avalanche Characteristics of Cryptographic Functions"

Zhang and Zheng [ZZ96] termed the *absolute indicator* $\mathcal{M}(f)$ as the maximum absolute value of the autocorrelation (of a Boolean function f). Analogously for a vectorial Boolean function, we define

Definition 68 (Absolute indicator)**.** Given a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. The *absolute indicator* of F is

$$\mathcal{M}(F) := \max_{\beta \in \mathbb{F}_2^n \setminus \{0\}} \mathcal{M}(F_\beta) = \max_{\alpha,\beta \in \mathbb{F}_2^n \setminus \{0\}} |\mathcal{A}_F(\alpha,\beta)|,$$

that is the maximum absolute indicator of F's non-trivial component functions.

We call the multiset $\left\{\mathcal{A}_F(\alpha,\beta) \,\middle|\, \alpha,\beta \in \mathbb{F}_2^n\right\}$ the *autocorrelation* or ACT *spectrum* of F.

Zhang et al. [Zha+00, Section 3] further showed that

$$\mathrm{ACT} = \mathrm{DDT} \cdot H$$

where H is the Walsh matrix of order n. In other words, the ACT is the Walsh transformed DDT of F:

$$\begin{aligned}(\mathrm{DDT}_F \cdot H)[\alpha,\beta] &= \sum_{\gamma\in\mathbb{F}_2^n} \left|\left\{x \in \mathbb{F}_2^n \,\middle|\, F(x)+F(x+\alpha)=\gamma\right\}\right| \cdot H[\gamma,\beta] \\ &= \sum_{\gamma\in\mathbb{F}_2^n} \left|\left\{x \in \mathbb{F}_2^n \,\middle|\, F(x)+F(x+\alpha)=\gamma\right\}\right| \cdot (-1)^{\gamma\cdot\beta} \\ &= \sum_{x\in\mathbb{F}_2^n} (-1)^{\beta\cdot(F(x)+F(x+\alpha))} \\ &= \mathcal{A}_F(\alpha,\beta)\,.\end{aligned}$$

Because of the correspondence between the ACT and the DLCT, this corresponds to [Bar+19, Proposition 1].

[Bar+19] Bar-On et al., “DLCT: A New Tool for Differential-Linear Cryptanalysis”

Let us now recall some properties of and links between the above discussed notations.

3.1.1 *Links between the ACT, DDT, and Walsh transformation*

► Sum of the ACT entries, within a row or a column

It is well-known that the entries $\mathcal{A}_F(\alpha,\beta)$, $\beta \neq 0$ in each nonzero row in the ACT of F sum to zero if and only if F is a permutation (see e. g. [Ber+06, Proposition 2]). The same property holds when the entries $\mathcal{A}_F(\alpha,\beta)$, $\alpha \neq 0$ in each nonzero column in the ACT are considered (see e. g. [Ber+06, Eq. (9)]).

[Ber+06] Berger et al., “On Almost Perfect Nonlinear Functions Over $\mathbf{F}_2^n$”

► Link between differential and linear cryptanalysis

The following proposition shows that the restriction of the autocorrelation function $\alpha \mapsto \mathcal{A}_F(\alpha,\beta)$ can be seen as the discrete Fourier transform of the squared Walsh transform of F_β: $\gamma \mapsto \widehat{F}^2(\gamma,\beta)$. As previously mentioned, $\beta \mapsto \mathcal{A}_F(\alpha,\beta)$ similarly corresponds to the Fourier transform of the row of index α in the DDT: $\theta \mapsto \delta_F(\alpha,\theta)$. It is worth noticing that this correspondence points out the well-known relationship between the Walsh transform of F and its DDT, which was exhibited in [CV95; BN13].

Proposition 69. *Let F be a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$. Then, for all $\alpha,\beta \in \mathbb{F}_2^n$, we have*

$$\begin{aligned}\mathcal{A}_F(\alpha,\beta) &= 2^{-n}\sum_{\gamma\in\mathbb{F}_2^n}(-1)^{\alpha\cdot\gamma}\widehat{F}^2(\gamma,\beta) &(3.2)\\ &= \sum_{\theta\in\mathbb{F}_2^n}(-1)^{\beta\cdot\gamma}\delta_F(\alpha,\theta)\,. &(3.3)\end{aligned}$$

Conversely, the inverse Fourier transform leads to

$$\widehat{F}^2(\alpha,\beta) = \sum_{\gamma\in\mathbb{F}_2^n}(-1)^{\alpha\cdot\gamma}\mathcal{A}_F(\gamma,\beta) \tag{3.4}$$

$$\delta_F(\alpha,\beta) = 2^{-n}\sum_{\theta\in\mathbb{F}_2^n}(-1)^{\beta\cdot\theta}\mathcal{A}_F(\alpha,\theta)\,, \tag{3.5}$$

for all $\alpha,\beta\in\mathbb{F}_2^n$.

Proof. We first prove Relation (3.2) which involves the squared Walsh transform. For all $\alpha,\beta\in\mathbb{F}_2^n$, we have

$$\begin{aligned}\sum_{\gamma\in\mathbb{F}_2^n}(-1)^{\alpha\cdot\gamma}\widehat{F}^2(\gamma,\beta) &= \sum_{\gamma\in\mathbb{F}_2^n}(-1)^{\alpha\cdot\gamma}\sum_{x\in\mathbb{F}_2^n}(-1)^{F_\beta(x)+\gamma\cdot x}\sum_{y\in\mathbb{F}_2^n}(-1)^{F_\beta(y)+\gamma\cdot y}\\ &= \sum_{x,y\in\mathbb{F}_2^n}(-1)^{F_\beta(x)+F_\beta(y)}\left(\sum_{\gamma\in\mathbb{F}_2^n}(-1)^{\gamma\cdot(\alpha+x+y)}\right)\\ &= 2^n\sum_{x\in\mathbb{F}_2^n}(-1)^{F_\beta(x)+F_\beta(x+\alpha)} = 2^n\mathcal{A}_F(\alpha,\beta)\end{aligned}$$

where the last equality comes from the fact that

$$\sum_{\gamma\in\mathbb{F}_2^n}(-1)^{\gamma\cdot(\alpha+x+y)} = \begin{cases}2^n & \text{for } \alpha+x+y=0\\ 0 & \text{otherwise.}\end{cases}$$

Obviously, Eq. (3.4) can be directly derived from Eq. (3.2) by applying the inverse Fourier transform. We now prove the relation involving the DDT, namely Eq. (3.5). For all $\alpha,\beta\in\mathbb{F}_2^n$, we have

$$\begin{aligned}\sum_{\theta\in\mathbb{F}_2^n}(-1)^{\beta\cdot\theta}\mathcal{A}_F(\alpha,\theta) &= \sum_{\theta\in\mathbb{F}_2^n}(-1)^{\beta\cdot\theta}\sum_{x\in\mathbb{F}_2^n}(-1)^{\theta\cdot\Delta_\alpha(F)(x)}\\ &= \sum_{x\in\mathbb{F}_2^n}\sum_{\theta\in\mathbb{F}_2^n}(-1)^{\theta\cdot(\beta+\Delta_\alpha(F)(x))}\\ &= \sum_{x\in\Delta_\alpha(F)^{-1}(\beta)}2^n = 2^n\delta_F(\alpha,\beta)\,.\end{aligned}$$

Eq. (3.3) then follows directly by the inverse Fourier transform.

As a corollary, Parseval's equality leads to an expression of the sum of all squared entries in each row, and in each column of the autocorrelation table.

Corollary 70. *Let F be a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$. Then, for all $\alpha,\beta\in\mathbb{F}_2^n$, we have*

$$\sum_{\alpha\in\mathbb{F}_2^n}\mathcal{A}_F^2(\alpha,\beta) = 2^{-n}\sum_{\gamma\in\mathbb{F}_2^n}\widehat{F}^4(\gamma,\beta) \quad \textit{and} \quad \sum_{\beta\in\mathbb{F}_2^n}\mathcal{A}_F^2(\alpha,\beta) = 2^n\sum_{\theta\in\mathbb{F}_2^n}\delta_F^2(\alpha,\theta)\,.$$

In the following, we show that the ACT spectrum is affine invariant, the extended ACT spectrum is extended affine invariant and the ACT spectrum is not invariant under CCZ equivalence.

3.1.2 *Invariance under Equivalence Relations*

Having an equivalence relation on the set of all n-bit functions, allows us to partition these functions into equivalence classes. Properties which are invariant under this equivalence notion can then be tested on only one representative of each class – resulting in a massive decrease of complexity, if we want to characterise the whole set of functions under this property. Three well-known equivalences are the following:

Definition 71 (Equivalence relations for vectorial Boolean functions)**.** Given two functions F and $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$. We say these functions are

1. *affine equivalent* ($F \overset{\mathrm{A}}{\sim} G$) if there exist two affine permutations A and B such that $G = B \circ F \circ A$,
2. *extended affine equivalent* ($F \overset{\mathrm{EA}}{\sim} G$) if there exist two affine permutations A and B, and an affine function C such that $G = B \circ F \circ A + C$,
3. *CCZ equivalent* ($F \overset{\mathrm{CCZ}}{\sim} G$) if their graphs are affine equivalent.

Here, the graph of a function is defined as $\{(x, F(x)) \mid x \in \mathbb{F}_2^n\}$.

A nice property of the ACT is that its spectrum is invariant under affine equivalence, and further its extended ACT spectrum, that is the multiset of absolute autocorrelation coefficients, $\{|\mathcal{A}_F(\alpha,\beta)| \mid \alpha,\beta \in \mathbb{F}_2^n\}$, is invariant under extended affine equivalence.

Proposition 72 (Affine and Extended Affine Invariance)**.** *Given two permutations F, and G on $\mathbb{F}_2^n$. Let further $A = L_a + a$, $B = L_b + b$ be two affine permutations and $C = L_c + c$ be an affine function. Then*

$$G = B \circ F \circ A \quad \Rightarrow \quad \mathcal{A}_G(\alpha,\beta) = \mathcal{A}_F(L_a(\alpha), L_b^\top(\beta))$$

and

$$G = B \circ F \circ A + C \quad \Rightarrow \quad |\mathcal{A}_G(\alpha,\beta)| = \left|\mathcal{A}_F(L_a(\alpha), L_b^\top(\beta))\right|,$$

where $L^\top$ denotes the adjoint of the linear mapping L.

Proof. From the affine equivalence of F and G, we have

$$G(x) = L_b(F(L_a(x) + a)) + b\ .$$

Each entry $\mathrm{ACT}_F[\alpha,\beta]$ corresponds to the number of solutions x for the equation

$$\beta \cdot (F(x) + F(x+\alpha)) = 0.$$

Thus for each entry of G's ACT at position (α,β), we count the number of solutions for

$$\beta \cdot [L_b(F(L_a(x) + a)) + b + L_b(F(L_a(x+\alpha) + a)) + b] = 0.$$

Substituting $x' = L_a(x) + a$, this simplifies to

$$\begin{aligned} &\beta \cdot L_b(F(x') + F(x' + L_a(\alpha))) = 0 \\ \Leftrightarrow \quad &L_b^\top(\beta) \cdot (F(x') + F(x' + L_a(\alpha))) = 0 \end{aligned}$$

thus the number of solutions for this equation is nothing else as the ACT entry of F at position $(L_a(\alpha), L_b^\top(\beta))$.

For the second point, we now only have to show how $G = F + C$ behaves. By definition of the autocorrelation we have

$$\begin{aligned}\mathcal{A}_G(\alpha,\beta) &= \sum_{x\in\mathbb{F}_2^n} (-1)^{\beta\cdot(G(x)+G(x+\alpha))} \\ &= \sum_{x\in\mathbb{F}_2^n} (-1)^{\beta\cdot(F(x)+L_c(x)+c+F(x+\alpha)+L_c(x+\alpha)+c)}\end{aligned}$$

where $C(x) = L_c(x) + c$

$$\begin{aligned}&= \sum_{x\in\mathbb{F}_2^n} (-1)^{\beta\cdot(F(x)+F(x+\alpha)+L_c(\alpha))} \\ &= (-1)^{\beta\cdot(L_c(\alpha))}\mathcal{A}_F(\alpha,\beta),\end{aligned}$$

and thus the affine map C only influences the sign of the autocorrelation value. ∎

The (extended) affine invariance of the (extended) ACT spectra follows directly from this proposition.

Corollary 73. *Given two permutations F, and G on $\mathbb{F}_2^n$.*

- *If $F \overset{A}{\sim} G$, the ACT spectrum of F equals that of G.*
- *If $F \overset{EA}{\sim} G$, the extended ACT spectrum of F equals that of G.*

To examine the behaviour under CCZ equivalence, let us first recall that the ACT is related to linear structures in the following way, see also [Zha+00; MT14].

Lemma 74 (Linear Structures). *Given $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, then*

$$\mathcal{A}_F(\alpha,\beta) = \pm 2^n$$

if and only if (α,β) forms a linear structure for F.

Proof. This follows from the fact that, for a linear structure, by definition,

$$\beta\cdot(F(x) + F(x+\alpha))$$

is constant zero or one. The sign of the entry thus determines, if the linear structure is constant one (negative) or zero (positive). ∎

One consequence of this is the next corollary.

Corollary 75 (Inversion). *Given a permutation $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, then the ACT spectrum of F is in general not equal to the ACT spectrum of F^{-1}.*

Proof. Counterexamples are the S-boxes used in the ciphers SAFER [Mas94], SC2000 [Shi+02], and FIDES [Bil+13], where the S-boxes have linear structures in one direction but non in the other direction, and the Gold permutations as analysed in Section 3.2.4. ∎

An interesting implication of this is that it might be advantageous when doing a differential-linear cryptanalysis, to look at both directions, encryption and decryption, of a cipher.

Another direct consequence of Corollary 75 is the following result.

Corollary 76. *Given two CCZ-equivalent permutations $F \overset{CCZ}{\sim} G$ on $\mathbb{F}_2^n$. Their ACT spectrum is in general not invariant.*

Proof. A function and its inverse are always CCZ equivalent. Thus Corollary 75 gives a counterexample.

Zhang et al. further showed how the ACT of F and its inverse F^{-1} are related, see [Zha+00, Corollary 1]. In particular they showed that

[Zha+00] Zhang et al., "Relating Differential Distribution Tables to Other Properties of of Substitution Boxes"

$$\mathrm{ACT}_{F^{-1}} = H^{-1} \cdot \mathrm{ACT}_F^{\top} \cdot H,$$

which in our notation is

$$\mathcal{A}_{F^{-1}}(\alpha,\beta) = \frac{1}{2^n} \sum_{\gamma,\theta \in \mathbb{F}_2^n} (-1)^{\alpha\cdot\gamma+\beta\cdot\theta} \mathcal{A}_F(\gamma,\theta).$$

3.2 LOWER BOUND ON THE ABSOLUTE INDICATOR

Finding the smallest possible absolute indicator for a *Boolean function* is an open question investigated by many authors. Zhang and Zheng conjectured [ZZ96, Conjecture 1] that the absolute indicator of a balanced Boolean function of n variables was at least $2^{\frac{n+1}{2}}$. But this was later disproved first for odd values of $n \geqslant 9$ by modifying the Patterson-Wiedemann construction, namely for $n \in \{9, 11\}$ in [Kav+07], for $n = 15$ in [MS02; Kav16] and for $n = 21$ in [Gan+06]. For the case n even, Tang and Maitra [TM18] gave a construction for balanced Boolean functions with absolute indicator strictly less than $2^{n/2}$ when $n \equiv 2 \bmod 4$. Very recently, similar examples for $n \equiv 0 \bmod 4$ were exhibited by Kavut et al. [Kav+19]. However, we now show that such small values for the absolute indicator cannot be achieved for *vectorial Boolean functions*.

[ZZ96] Zhang and Zheng, "GAC — the Criterion for Global Avalanche Characteristics of Cryptographic Functions"

3.2.1 *General Case*

Parseval's equality leads to the following lower bound on the sum of all *squared* autocorrelation coefficients in each row. This result can be found in [Nyb95] (see also [Ber+06, Theorem 2]), but we recall the proof for the sake of completeness.

[Nyb95] Nyberg, "S-boxes and Round Functions with Controllable Linearity and Differential Uniformity"

[Ber+06] Berger et al., "On Almost Perfect Nonlinear Functions Over $\mathbf{F}_2^n$"

Proposition 77. *Let F be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^n$. Then, for all $\alpha \in \mathbb{F}_2^n$, we have*

$$\sum_{\beta \in \mathbb{F}_2^n \setminus \{0\}} \mathcal{A}_F^2(\alpha,\beta) \geqslant 2^{2n} .$$

Moreover, equality holds for all nonzero $\alpha \in \mathbb{F}_2^n$ if and only if F is APN.

Proof. From Corollary 70, we have that, for all $\alpha \in \mathbb{F}_2^n$,

$$\sum_{\beta \in \mathbb{F}_2^n} \mathcal{A}_F^2(\alpha,\beta) = 2^n \sum_{\theta \in \mathbb{F}_2^n} \delta_F^2(\alpha,\theta)$$

Cauchy-Schwarz inequality implies that

$$\left(\sum_{\theta\in\mathbb{F}_2^n}\delta_F(\alpha,\theta)\right)^2 \leqslant \left(\sum_{\theta\in\mathbb{F}_2^n}\delta_F^2(\alpha,\theta)\right)\cdot\left|\left\{\theta\in\mathbb{F}_2^n \,\middle|\, \delta_F(\alpha,\theta)\neq 0\right\}\right|,$$

with equality if and only if all nonzero elements in $\left\{\delta_F(\alpha,\theta) \,\middle|\, \theta\in\mathbb{F}_2^n\right\}$ are equal. Using that

$$\left|\left\{\theta\in\mathbb{F}_2^n \,\middle|\, \delta_F(\alpha,\theta)\neq 0\right\}\right| \leqslant 2^{n-1}$$

with equality for all nonzero α if and only if F is APN, we deduce that

$$\sum_{\theta\in\mathbb{F}_2^n}\delta_F^2(\alpha,\theta) \geqslant 2^{2n}\times 2^{-(n-1)} = 2^{n+1}$$

again with equality for all nonzero α if and only if F is APN. Equivalently, we deduce that

$$\sum_{\beta\in\mathbb{F}_2^n}\mathcal{A}_F^2(\alpha,\beta) \geqslant 2^{2n+1}$$

with equality for all nonzero a if and only if F is APN. Then the result follows from the fact that

$$\sum_{\beta\in\mathbb{F}_2^n}\mathcal{A}_F^2(\alpha,\beta) = 2^{2n} + \sum_{\beta\in\mathbb{F}_2^n\setminus\{0\}}\mathcal{A}_F^2(\alpha,\beta)\,.$$

From the lower bound on the sum of all squared coefficients within a row of the ACT, we deduce the following lower bound on the absolute indicator.

Proposition 78 (Lowest possible absolute indicator). *Let F be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^n$. Then,*

$$\mathcal{M}(F) \geqslant \frac{2^n}{\sqrt{2^n-1}} > 2^{n/2}\,.$$

Proof. From the facts that

$$\sum_{\beta\in\mathbb{F}_2^n\setminus\{0\}}\mathcal{A}_F^2(\alpha,\beta) \geqslant 2^{2n}$$

and

$$\sum_{\beta\in\mathbb{F}_2^n\setminus\{0\}}\mathcal{A}_F^2(\alpha,\beta) \leqslant \mathcal{M}(F)^2(2^n-1)$$

the result directly follows.

We can get a more precise lower bound on the absolute indicator by using the fact that all autocorrelation coefficients are divisible by 8. We even have a stronger property for functions having a lower algebraic degree as shown in the following proposition.

Proposition 79 (Divisibility). *Let $n > 2$ and $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a permutation with algebraic degree at most d. Then, for any $\alpha,\beta\in\mathbb{F}_2^n$, $\mathcal{A}_F(\alpha,\beta)$ is divisible by $2^{\lceil \frac{n}{d-1}\rceil+1}$.*

Most notably, the autocorrelation coefficients of a permutation are divisible by 8.

Proof. From the definition of the autocorrelation, we know that

$$\mathcal{A}_F(\alpha,\beta) = \widehat{\Delta_\alpha\big(F_\beta\big)}(0).$$

For the sake of readability, we define $h_{\alpha,\beta} = \Delta_\alpha\big(F_\beta\big)$. We can derive two properties of this Boolean function $h_{\alpha,\beta}$. First, as F has degree at most d, $\deg(h_{\alpha,\beta}) \leqslant d-1$. Second, $h_{\alpha,\beta}(x) = h_{\alpha,\beta}(x+\alpha)$.

We now focus on the divisibility of $\widehat{h_{\alpha,\beta}}(0)$. First, assume for simplicity that $\alpha = e_n$, we discuss the general case afterwards. Then we can write $h_{e_n,\beta}$ as $h_{e_n,\beta}(x) = g(x_1,\dots,x_{n-1})$ with $g : \mathbb{F}_2^{n-1} \to \mathbb{F}_2$, because $h_{e_n,\beta}(x+e_n) = h_{e_n,\beta}(x)$. The Walsh coefficient of $h_{e_n,\beta}$ at point 0 can then be computed as

$$\widehat{h_{e_n,\beta}}(0) = \sum_{x\in\mathbb{F}_2^{n-1}, x_n\in\mathbb{F}_2} (-1)^{f(x,x_n)} = 2\cdot \sum_{x\in\mathbb{F}_2^{n-1}} (-1)^{g(x)} = 2\cdot\widehat{g}(0)$$

Now $\deg(g) \leqslant d-1$. It is well-known that the Walsh coefficients of a Boolean function f are divisible by $2^{\left\lceil \frac{n}{\deg(f)}\right\rceil}$ (see [McE72] or [Car10a, Section 3.1]). We then deduce that $\widehat{g}(0)$ is divisible by $2^{\left\lceil \frac{n-1}{d-1}\right\rceil}$, implying that $\widehat{h_{e_n,\beta}}(0)$ is divisible by $2^{\left\lceil \frac{n-1}{d-1}\right\rceil+1}$. Most notably, if F is bijective, $d \leqslant n-1$. We then have that

$$\left\lceil \frac{n-1}{d-1} \right\rceil + 1 \geqslant 3,$$

implying that $\widehat{h_{e_n,\beta}}(0)$ is divisible by 8.

[McE72] McEliece, “Weight congruences for p-ary cyclic codes”

[Car10a] Carlet, “Boolean Functions for Cryptography and Error-Correcting Codes”

In the case that $\alpha \neq e_n$, we can find a linear transformation L, such that $L(e_n) = \alpha$, with which we have the affine equivalent function $G = F \circ L \overset{\mathrm{A}}{\sim} F$. Now for G the same argument as above holds and thus $\mathcal{A}_G(\alpha,\beta)$ is divisible by $2^{\left\lceil \frac{n}{d-1}\right\rceil+1}$. Due to the affine invariance of G’s and F’s ACT spectra the same holds for $\mathcal{A}_F(\alpha,\beta)$ in this case.

The absolute indicator is known for very few permutations only, except in the case of permutations of degree less than or equal to 2, where the result is trivial. To our best knowledge, one of the only functions whose absolute indicator is known is the inverse mapping $F(x) = x^{2^n-2}$ over $\mathbb{F}_{2^n}$ [Cha+07]: $\mathcal{M}(F) = 2^{\frac{n}{2}+1}$ when n is even. When n is odd, $\mathcal{M}(F) = \mathcal{L}(F)$ when $\mathcal{L}(F) \equiv 0 \bmod 8$, and $\mathcal{M}(F) = \mathcal{L}(F) \pm 4$ otherwise (see Section 3.2.3 for an alternative proof).

[Cha+07] Charpin et al., “Propagation characteristics of $x \mapsto x^{-1}$ and Kloosterman sums”

We now study the absolute indicator of some types of vectorial Boolean functions.

3.2.2 *Case of power permutations*

For power permutations on $\mathbb{F}_{2^n}$ we can show that the autocorrelation spectrum is invariant for all component functions, analogously to the same well known fact for the Walsh spectrum. In this case, the trace function is used as the inner product, i. e. $\alpha\cdot\beta = \mathrm{tr}_n(\alpha\beta)$ with

$$\mathrm{tr}_n : \mathbb{F}_{2^n} \to \mathbb{F}_2$$
$$\mathrm{tr}_n(x) := \sum_{i=0}^{n-1} x^{2^i}$$

where we leave out the subscript, if it is clear from the context. Thus we have the following corollary.

Corollary 80. *Let F be a power permutation on $\mathbb{F}_{2^n}$, i. e. $F = x^k$ where for k it holds that* $\gcd(k, 2^n - 1) = 1$. *Then, for all non-zero α and β in $\mathbb{F}_{2^n}$,*

$$\mathcal{A}_F(\alpha,\beta) = \mathcal{A}_F(1,\alpha^k\beta) = \mathcal{A}_F(\alpha\beta^{\frac{1}{k}},1)\,.$$

Most notably, all non-zero component functions F_β of F have the same (Boolean) absolute indicator: $\mathcal{M}(F) = \mathcal{M}\left(F_\beta\right)$ for all $\beta \in \mathbb{F}_{2^n} \setminus \{0\}$ and $\mathcal{M}\left(F_\beta\right) > 2^{n/2}$.

[Ber+06] Berger et al., "On Almost Perfect Nonlinear Functions Over $\mathbf{F}_2^n$"

Proof. The fact that $\mathcal{A}_F(\alpha,\beta) = \mathcal{A}_F(1,\alpha^k\beta)$ has been proved for instance in [Ber+06, Proposition 4]. The second equality comes from the fact that

$$\alpha\cdot\left(x^k + (x+\alpha)^k\right) = 1\cdot\left(\left(\beta^{\frac{1}{k}}x\right)^k + \left(\beta^{\frac{1}{k}}x + \alpha\beta^{\frac{1}{k}}\right)^d\right),$$

where $\beta^{\frac{1}{k}}$ only exists if $\gcd(k, 2^n - 1) = 1$. It follows that it is enough to compute only one column of the ACT, as the remaining ones are just permutations of each other. In other words, all Boolean functions F_β, $\beta \neq 0$, have the same absolute indicator.

3.2.3 *Case of APN functions*

In the specific case of APN functions, we can also exhibit a stronger condition than Proposition 78 on the lowest possible absolute indicator.

Proposition 81 (Lowest possible indicator for APN functions)**.** *Let n be a positive integer. If there exists an APN function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$ with absolute indicator M, then there exists a balanced Boolean function of n variables with linearity M.*

Proof. If F is APN, then $\delta_F(\alpha,\beta) \in \{0,2\}$ for all α,β, $\alpha \neq 0$. It follows that, for each nonzero α, we can define a Boolean function g_α of n variables such that

$$g_\alpha(\beta) = \begin{cases} 0 & \text{if } \delta_F(\alpha,\beta) = 0 \\ 1 & \text{if } \delta_F(\alpha,\beta) = 2\,. \end{cases}$$

Equivalently,

$$\delta_F(\alpha,\beta) = 1 - (-1)^{g_\alpha(\beta)}\,.$$

Obviously, all g_α are balanced. Moreover, we deduce from Eq. (3.3) that, for all nonzero α,β,

$$\begin{aligned}\mathcal{A}_F(\alpha,\beta) &= \sum_{\theta\in\mathbb{F}_2^n}(-1)^{\beta\cdot\theta}\delta_F(\alpha,\theta)\\ &= \sum_{\theta\in\mathbb{F}_2^n}(-1)^{\beta\cdot\theta}\left(1-(-1)^{g_\alpha(\theta)}\right)\\ &= -\sum_{\theta\in\mathbb{F}_2^n}(-1)^{\beta\cdot\theta+g_\alpha(\theta)} = -\widehat{g_\alpha}(\beta)\end{aligned}$$

where $\widehat{g_\alpha}(\beta)$ denotes the value of the Walsh transform of g_α at point β. This implies that

$$\max_{\beta\neq 0}|\mathcal{A}_F(\alpha,\beta)| = \mathcal{L}(g_\alpha)\,.$$

The result then directly follows.

To our best knowledge, the smallest known linearity for a balanced Boolean function is obtained by Dobbertin's recursive construction [Dob95]. For instance, for $n = 9$, the smallest possible linearity for a balanced Boolean function is known to belong to the set $\{24, 28, 32\}$, which implies that exhibiting an APN function over $\mathbb{F}_2^9$ with absolute indicator 24 would determine the smallest linearity for such a function.

[Dob95] Dobbertin, "Construction of Bent Functions and Balanced Boolean Functions with High Nonlinearity"

It is worth noticing that the proof of the previous proposition shows that the knowledge of g directly determines the ACT. This explains why the absolute indicator of the inverse mapping over $\mathbb{F}_{2^n}$, n odd, is derived from its linearity as proved in [Cha+07, Theorem 1] and detailed in the following example.

[Cha+07] Charpin et al., "Propagation characteristics of $x \mapsto x^{-1}$ and Kloosterman sums"

Example 82 (ACT of the inverse mapping, n odd)**.** For any $\alpha \in \mathbb{F}_{2^n} \setminus \{0\}$, the Boolean function g_α which characterises the support of Row α in the DDT of the inverse mapping $F : x \mapsto x^{-1}$ coincides with $(1 + F_{\alpha^{-1}})$ except on two points:

$$g_\alpha(\beta) = \begin{cases} 1 + \mathrm{tr}(\alpha^{-1}\beta^{-1}) & \text{if } \beta \notin \left\{0, \alpha^{-1}\right\} \\ 0 & \text{if } \beta = 0 \\ 1 & \text{if } \beta = \alpha^{-1} \end{cases} .$$

This comes from the fact that the equation

$$(x + \alpha)^{-1} + x^{-1} = \beta$$

for $\beta \neq \alpha^{-1}$ can be rewritten as

$$x + (x + \alpha) = \beta(x + \alpha)x$$

or equivalently when $\beta \neq 0$, by setting $y = \alpha^{-1}x$,

$$y^2 + y = \alpha^{-1}\beta^{-1} .$$

It follows that this equation has two solutions if and only if $\mathrm{tr}(\alpha^{-1}\beta^{-1}) = 0$. From the proof of the previous proposition, we deduce

$$\begin{aligned} \mathcal{A}_F(\alpha, \beta) &= -\widehat{g_\alpha}(\beta) \\ &= \widehat{F_{\alpha^{-1}}}(\beta) + 2\left(1 - (-1)^{\mathrm{tr}(\alpha^{-1}\beta)}\right) , \end{aligned}$$

where the additional term corresponds to the value of the sum defining the Walsh transform $\widehat{F_{\alpha^{-1}}}(\beta)$ at points 0 and α^{-1}.

It can be observed that Propositions 77 and 81 are actually more general and apply as soon as the DDT of F contains one row composed of 0s and 2s only.

Proposition 83. *Let F be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^n$. Then, for any fixed $\alpha \in \mathbb{F}_2^n \setminus \{0\}$, the following properties are equivalent:*

(i) $\displaystyle\sum_{\beta \in \mathbb{F}_2^n \setminus \{0\}} \mathcal{A}_F^2(\alpha, \beta) = 2^{2n}$

(ii) For all $\beta \in \mathbb{F}_2^n$, $\delta_F(\alpha, \beta) \in \{0, 2\}$.

Moreover, if these properties hold, then there exists a balanced Boolean function $g : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$ such that

$$\mathcal{L}(g) = \max_{\beta \in \mathbb{F}_2^n \setminus \{0\}} |\mathcal{A}_F(\alpha, \beta)| \,.$$

Let us now take a closer look at the absolute indicator of some specific APN permutations.

3.2.4 *Case of APN permutations*

As previously observed, the ACT and the absolute indicator are not invariant under inversion. Then, while the absolute indicator of a quadratic permutation is trivially equal to 2^n, computing the absolute indicator of the inverse of a quadratic permutation is not obvious at all. Indeed, the absolute indicator depends on the considered function, as we will see next.

► INVERSES OF QUADRATIC APN PERMUTATIONS, n ODD
For instance, for $n = 9$, the inverses of the two APN Gold permutations x^3 and x^5, namely x^{341} and x^{409}, do not have the same absolute indicator: the absolute indicator of x^{341} is 56 while the absolute indicator of x^{409} is 72.

Nevertheless, the specificity of quadratic APN permutations for n odd is that they are *crooked* [BF98], which means that the image sets of their derivatives $\Delta_\alpha(F)$, $\alpha \neq 0$, is the complement of a hyperplane $\mathrm{Span}\{\pi(\alpha)\}^\perp$. Moreover, it is known (see e. g. [CC03, Proof of Lemma 5]) that all these hyperplanes are distinct, which implies that π is a permutation of $\mathbb{F}_2^n$ when we add to the definition that $\pi(0) = 0$. Then, the following proposition shows that, for any quadratic APN permutation F, the ACT of F^{-1} corresponds to the Walsh transform of π.

[BF98] Bending and Fon-Der-Flaass, "Crooked Functions, Bent Functions, and Distance Regular Graphs"

[CC03] Canteaut and Charpin, "Decomposing bent functions"

Proposition 84. *Let n be an odd integer and F be a quadratic APN permutation over $\mathbb{F}_2^n$. Let further π be the permutation of $\mathbb{F}_2^n$ defined by*

$$\mathrm{Im}\,\Delta_\alpha(F) = \mathbb{F}_2^n \setminus \mathrm{Span}\{\pi(\alpha)\}^\perp, \textit{when } \alpha \neq 0 \,,$$

and $\pi(0) = 0$. Then, for any nonzero α and β in $\mathbb{F}_2^n$, we have

$$\mathcal{A}_{F^{-1}}(\alpha, \beta) = -\widehat{\pi}(\beta, \alpha) \,.$$

It follows that

$$\mathcal{M}\left(F^{-1}\right) \geqslant 2^{\frac{n+1}{2}}$$

with equality if and only if π is an AB permutation.

Proof. Let α and β be two nonzero elements in $\mathbb{F}_2^n$. Then, from Relation (3.3), we deduce

$$\begin{aligned} \mathcal{A}_{F^{-1}}(\alpha, \beta) &= \sum_{\gamma \in \mathbb{F}_2^n} (-1)^{\beta \cdot \gamma} \delta_{F^{-1}}(\alpha, \gamma) \\ &= \sum_{\gamma \in \mathbb{F}_2^n} (-1)^{\beta \cdot \gamma} \delta_F(\gamma, \alpha) \,. \end{aligned}$$

By definition of π, we have that, for any nonzero γ,

$$\delta_F(\gamma, \theta) = \begin{cases} 2 & \text{if } \theta \cdot \pi(\gamma) = 1 \\ 0 & \text{if } \theta \cdot \pi(\gamma) = 0 \end{cases} .$$

It then follows that

$$\delta_F(\gamma,\theta) = 1 - (-1)^{\pi(\gamma)\cdot\theta}$$

where this equality holds for all $(\gamma,\theta) \neq (0,0)$ by using that $\pi(0) = 0$. Therefore, we have, for any nonzero α and β,

$$\mathcal{A}_{F^{-1}}(\alpha,\beta) = \sum_{\gamma\in\mathbb{F}_2^n} (-1)^{\beta\cdot\gamma}\left(1-(-1)^{\pi(\gamma)\cdot\alpha}\right) = -\widehat{\pi}(\beta,\alpha)\,.$$

As a consequence, $\mathcal{M}\left(F^{-1}\right)$ is equal to the linearity of π, which is at least $2^{\frac{n+1}{2}}$ with equality for AB functions.

It is worth noticing that the previous proposition is valid, not only for quadratic APN permutations, but for all *crooked* permutations, which are a particular case of AB functions. However, the existence of crooked permutations of degree strictly higher than 2 is an open question.

As a corollary of the previous proposition, we get some more precise information on the autocorrelation spectrum of the quadratic power permutations corresponding to Gold exponents, i. e. $F(x) = x^{2^i+1}$. Recall that x^{2^i+1} and $x^{2^{n-i}+1}$ are affine equivalent since the two exponents belong to the same cyclotomic coset modulo $(2^n - 1)$. This implies that their inverses share the same autocorrelation spectrum.

Corollary 85. *Let $n > 5$ be an odd integer and $0 < i < n$ with $\gcd(i,n) = 1$. Let F be the APN power permutation over $\mathbb{F}_{2^n}$ defined by $F(x) = x^{2^i+1}$. Then, for any nonzero α and β in $\mathbb{F}_2^n$, we have*

$$\mathcal{A}_{F^{-1}}(\alpha,\beta) = -\widehat{\pi}(\beta,\alpha) \quad \textit{where} \quad \pi(x) = x^{2^n-2^i-2}\,.$$

Most notably, the absolute indicator of F^{-1} is strictly higher than $2^{\frac{n+1}{2}}$.

Proof. The result comes from the form of the function π which defines the DDT of x^{2^i+1}. Indeed, for any nonzero $\alpha \in \mathbb{F}_{2^n}$, the number $\delta_F(\alpha,\beta)$ of solutions of

$$(x+\alpha)^{2^i+1} + x^{2^i+1} = \beta$$

is equal to the number of solutions of

$$x^{2^i} + x = 1 + \beta\alpha^{-(2^i+1)}$$

which is nonzero if and only if $\mathrm{tr}(\beta\alpha^{-(2^i+1)}) = 1$. It follows that

$$\pi(x) = x^{2^n-2^i-2}\,.$$

The autocorrelation spectrum of F^{-1} then follows from Proposition 84. Moreover, this function π cannot be AB since AB functions have algebraic degree at most $\frac{n+1}{2}$ [Car+98, Theorem 1], while π has degree $(n-2)$. It follows that π cannot be AB when $n > 5$. Therefore, the absolute indicator of the inverse of F^{-1}, i. e. the linearity of π, is strictly higher than $2^{\frac{n+1}{2}}$.

[Car+98] Carlet et al., “Codes, Bent Functions and Permutations Suitable For DES-like Cryptosystems”

In the specific case $n = 5$, it can easily be checked that the inverses of all Gold APN permutations $F(x) = x^{2^i+1}$ have absolute indicator 8.

▸ Cubic APN permutations

In the case of APN permutations of degree 3, we have a more precise result.

Proposition 86 (Cubic APN permutations). *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be APN with degree 3. Then we have that for non-zero α and λ*

$$|\mathcal{A}_F(\alpha,\lambda)| \in \left\{0, 2^{\frac{n+d(\alpha,\lambda)}{2}}\right\},$$

where $d(\alpha,\lambda) = \dim \mathsf{LS}(\Delta_\alpha(F_\lambda)) = \dim\left\{\beta \;\middle|\; \Delta_{\alpha,\beta}(F_\lambda) = c\right\}$ and $c \in \mathbb{F}_2$ is constant. Moreover, $\mathcal{A}_F(\alpha,\lambda) = 0$ if and only if $\Delta_\alpha(F_\lambda)$ is balanced, which equivalently means that it has an all-one derivative.

From this proposition, if n is odd, we obviously have that $\mathcal{M}(F) \geqslant 2^{\frac{n+1}{2}}$ with equality if and only if, for any nonzero $\alpha, \lambda \in \mathbb{F}_2^n$, either $\mathsf{LS}(\Delta_\alpha(F_\lambda)) = \{0,\alpha\}$ or there exists β such that $\Delta_\beta(\Delta_\alpha(F_\lambda)) = 1$. Moreover, if F is APN and $\mathcal{M}(F) = 2^{\frac{n+1}{2}}$, it follows from Proposition 77 that the number of nonzero λ such that $\mathsf{LS}(\Delta_\alpha(F_\lambda)) = \{0,\alpha\}$ is exactly 2^{n-1}.

Additionally, an upper bound on the absolute indicator can be established for two cubic APN permutations, namely the first Kasami power function and the Welch function. We denote the Kasami power functions K_i and the Welch power function W by

$$\begin{aligned} K_i &: \mathbb{F}_{2^n} \to \mathbb{F}_{2^n} \\ K_i &: x \mapsto x^{(2^{3i}+1)/(2^i+1)} \\ &= x^{4^i-2^i+1} \end{aligned} \quad \text{and} \quad \begin{aligned} W &: \mathbb{F}_{2^n} \to \mathbb{F}_{2^n} \\ W &: x \mapsto x^{2^{(n-1)/2}+3} \end{aligned}.$$

Proposition 87 ([Car08], Lemma 1). *The absolute indicator for W on $\mathbb{F}_{2^n}$ is bounded from above by*

$$\mathcal{M}(W) \leqslant 2^{\frac{n+5}{2}}$$

As long as the (regular) degree of the derivatives is small compared to the field size, the Weil bound gives a nontrivial upper bound for the absolute indicator of a vectorial Boolean function. This is particularly interesting for the Kasami functions as the Kasami exponents do not depend on the field size (contrary to for example the Welch exponent).

Proposition 88. *The absolute indicator of K_i on $\mathbb{F}_{2^n}$ is bounded from above by*

$$\mathcal{M}(K_i) \leqslant (4^i - 2^{i+1}) \cdot 2^{\frac{n}{2}}.$$

In particular,

$$\mathcal{M}(K_2) \leqslant 2^{\frac{n+5}{2}}.$$

Proof. Note that the two exponents with the highest degree of any derivative of K_i are $4^i - 2^i$ and $4^i - 2^{i+1} + 1$. The first exponent is even, so it can be reduced using the relation $\mathrm{tr}(y^2) = \mathrm{tr}(y)$. The result then follows from the Weil bound. Combining the bound with Proposition 86 yields the bound on K_2.

In the two cases of W and K_2, we deduce that the absolute indicator belongs to $\left\{2^{\frac{n+1}{2}}, 2^{\frac{n+3}{2}}, 2^{\frac{n+5}{2}}\right\}$. We actually conjecture the following.

Conjecture 89. *Let $n \geqslant 9$ be odd. Then $\mathcal{M}(K_2) \geqslant 2^{\frac{n+3}{2}}$ and $\mathcal{M}(W) \geqslant 2^{\frac{n+3}{2}}$.*

Conjecture 90. *If n odd and $n \not\equiv 0 \bmod 3$, then $\mathcal{M}(K_i) = 2^{\frac{n+1}{2}}$.*

Some other results on the autocorrelations of the Boolean functions $\mathrm{tr}(x^k)$ are known in the literature, which can be trivially extended to the vectorial power functions x^k if $\gcd(k,n) = 1$, see [GK04, Theorem 5], [Car08] and [SW09, Lemmata 2 and 3]. In the case $n = 6r$ and $k = 2^{2r} + 2^r + 1$, the power monomial x^k is not a permutation, but results for all component functions of x^k were derived by Canteaut et al. [Can+08]. We summarise the results in the following proposition.

[GK04] Gong and Khoo, "Additive Autocorrelation of Resilient Boolean Functions"

[Car08] Carlet, "Recursive Lower Bounds on the Nonlinearity Profile of Boolean Functions and Their Applications"

[SW09] Sun and Wu, "The lower bounds on the second order nonlinearity of three classes of Boolean functions with high nonlinearity"

[Can+08] Canteaut et al. "A new class of monomial bent functions"

Proposition 91. *Let $F(x) = x^k$ be a function on $\mathbb{F}_{2^n}$.*

1. *If n is odd and $k = 2^r + 3$ with $r = \frac{n+1}{2}$ then $\mathcal{M}(F) \in \{2^{\frac{n+1}{2}}, 2^{\frac{n+3}{2}}\}$.*
2. *If n is odd and k is the i-th Kasami exponent, where $3i \equiv \pm 1 \pmod{n}$, then $\mathcal{M}(F) = 2^{\frac{n+1}{2}}$.*
3. *If $n = 2m$ and $k = 2^{m+1} + 3$ then $\mathcal{M}(F) \leqslant 2^{\frac{3m}{2}+1}$.*
4. *If $n = 2m$, m odd and $k = 2^m + 2^{\frac{m+1}{2}} + 1$ then $\mathcal{M}(F) \leqslant 2^{\frac{3m}{2}+1}$.*
5. *If $n = 6r$ and $k = 2^{2r} + 2^r + 1$ then $\mathcal{M}(F) = 2^{4r}$.*

We now provide a different proof of the second case in the previous proposition that additionally relates the autocorrelation table of K_i with the Walsh spectrum of a Gold function.

Proposition 92 ([Dil99])**.** *Let n be odd, not divisible by 3 and $3i \equiv \pm 1 \pmod{n}$. Set $f = \mathrm{tr}(x^k)$ where $k = 4^i - 2^i + 1$ is the i-th Kasami exponent. Then*

$$\mathrm{Supp}\left(\widehat{f}\right) = \left\{\alpha \;\middle|\; \mathrm{tr}(\alpha^{2^k+1}) = 1\right\}.$$

Proposition 93. *Let n be odd, not divisible by 3 and $3i \equiv \pm 1 \pmod{n}$. Then*

$$\mathcal{A}_{K_i}(\alpha,\beta) = -\sum_{\gamma\in\mathbb{F}_{2^n}} (-1)^{\mathrm{tr}(\alpha\beta^{1/k}\gamma + \gamma^{2^k+1})},$$

where $k = 4^i - 2^i + 1$ is the i-th Kasami exponent and $1/k$ denotes the inverse of k in $\mathbb{Z}_{2^n-1}$. In particular, $\mathcal{M}(K_i) = 2^{\frac{n+1}{2}}$.

Proof. It is well-known that, if F is a power permutation over a finite field, its Walsh spectrum is uniquely defined by the entries $\widehat{F}(1,\beta)$. Indeed, for $\beta \neq 0$,

$$\begin{aligned}\widehat{K_i}(\gamma,\beta) &= \sum_{x\in\mathbb{F}_{2^n}} (-1)^{\mathrm{tr}(\beta x^k + \gamma x)} \\ &= \sum_{x\in\mathbb{F}_{2^n}} (-1)^{\mathrm{tr}(x^k + \gamma\beta^{-1/k}x)} = \widehat{K_i}(1,\gamma\beta^{-1/k}) \in \left\{0, \pm 2^{\frac{n+1}{2}}\right\},\end{aligned}$$

where the last fact follows because the Kasami function is AB. Then, by Eq. (3.2) and Proposition 92, for any nonzero α and β,

$$\begin{aligned}\mathcal{A}_{K_i}(\alpha,\beta) &= 2^{-n}\sum_{\gamma\in\mathbb{F}_{2^n}} (-1)^{\mathrm{tr}(\alpha\gamma)}\widehat{K_i}^2(\gamma,\beta) = 2^{-n}\sum_{\gamma\in\mathbb{F}_{2^n}} (-1)^{\mathrm{tr}(\alpha\gamma)}\widehat{K_i}(1,\gamma\beta^{-1/k})^2 \\ &= 2^{-n}\sum_{\gamma\beta^{-1/k}\in\mathrm{Supp}\left(\widehat{\mathrm{tr}(x^k)}\right)} (-1)^{\mathrm{tr}(\alpha\gamma)}2^{n+1} = 2\sum_{\gamma\in B}(-1)^{\mathrm{tr}(\alpha\gamma)}, \qquad (3.6)\end{aligned}$$

where $B = \left\{\gamma \in \mathbb{F}_{2^n} \,\middle|\, \operatorname{tr}((\gamma\beta^{-1/k})^{2^k+1}) = 1\right\}$. We have

$$\begin{aligned} 0 &= \sum_{\gamma \notin B} (-1)^{\operatorname{tr}(\alpha\gamma)} + \sum_{\gamma \in B} (-1)^{\operatorname{tr}(\alpha\gamma)} \\ &= \sum_{\gamma \notin B} (-1)^{\operatorname{tr}(\alpha\gamma + (\gamma\beta^{-1/k})^{2^k+1})} - \sum_{\gamma \in B} (-1)^{\operatorname{tr}(\alpha\gamma + (\gamma\beta^{-1/k})^{2^k+1})} , \end{aligned}$$

so

$$\sum_{\gamma \in \mathbb{F}_{2^n}} (-1)^{\operatorname{tr}(\alpha\gamma + (\gamma\beta^{-1/k})^{2^k+1})} = 2 \sum_{\gamma \in B} (-1)^{\operatorname{tr}(\alpha\gamma + (\gamma\beta^{-1/k})^{2^k+1})} = -2 \sum_{\gamma \in B} (-1)^{\operatorname{tr}(\alpha\gamma)} .$$

Plugging this into Eq. (3.6), we obtain

$$\mathcal{A}_{K_i}(\alpha, \beta) = - \sum_{\gamma \in \mathbb{F}_{2^n}} (-1)^{\operatorname{tr}(\alpha\gamma + (\gamma\beta^{-1/k})^{2^k+1})} = - \sum_{\gamma \in \mathbb{F}_{2^n}} (-1)^{\operatorname{tr}(\alpha\beta^{1/k}\gamma + \gamma^{2^k+1})} .$$

It can be easily checked that the equation also holds for $\beta = 0$. Observe that $\gcd(k, n) = 1$, so the Gold function is AB and

$$\mathcal{M}(K_i) = 2^{\frac{n+1}{2}} .$$

Note that the cases $3i \equiv 1 \pmod{n}$ and $3i \equiv -1 \pmod{n}$ are essentially only one case because the i-th and $(n-i)$-th Kasami exponents belong to the same cyclotomic coset. Indeed, $(4^{(n-i)} - 2^{n-i} + 1)2^{2i} \equiv 4^i - 2^i + 1 \pmod{2^n - 1}$.

[Li+19a] Li et al. *On the Differential Linear Connectivity Table of Vectorial Boolean Functions*

▶ NOTE Li et al. [Li+19a] independently observed similar results. In [Can+19c] we merged both drafts; the paper is currently under submission.

4

Instantiating the WSN construction

—Lucas Hartmann

► SYNOPSIS Our main contribution is a block cipher instance of the WSN construction which allows us to give a strong bound on the best differential over several rounds of the cipher. This chapter is based on the paper

> Anne Canteaut, Virginie Lallemand, Gregor Leander, Patrick Neumann, and Friedrich Wiemer. "BISON – Instantiating the Whitened Swap-Or-Not Construction". In: *EUROCRYPT 2019, Part III*. ed. by Yuval Ishai and Vincent Rijmen. Vol. 11478. LNCS. Springer, Heidelberg, May 2019, pp. 585–616. DOI: `10.1007/978-3-030-17659-4_20`. IACR PREPRINT: `2018/1011`.

All authors contributed equally.

After recalling the WSN construction, we identify inherent restrictions and define a family of WSN instances fulfilling these restrictions. Subsequently, we discuss the differential cryptanalysis of these "BISON-like" instances. Following this analysis, we give two concrete species of BISON-like block ciphers: BISON (with odd block length) and WISENT (with even block length). For both we report on further cryptanalysis and give some remarks on implementation figures.

4.1 INTRODUCTION

In the context of Problem 3 (Block Cipher Constructions) an interesting result is the construction by Tessaro [Tes15]. His construction is based on the Swap-or-Not construction by Hoang et al. [Hoa+12], which was designed for the setting where the component functions are secret. Instead of being based on random permutations, this construction requires only a set of random (Boolean) functions. Tessaro's construction, coined Whitened Swap-Or-Not (WSN), requires only two public random (Boolean) functions f_i with n-bit input, and can be proven to achieve full security, see Section 4.2 for more details.

[Tes15] Tessaro "Optimally Secure Block Ciphers from Ideal Primitives"

[Hoa+12] Hoang et al. "An Enciphering Scheme Based on a Card Shuffle"

However, and this is the main motivation for our work, *no instance of this construction is known*. This situation is in sharp contrast to the case of the iterated Even-Mansour construction, where many secure instances are known for a long time already, as discussed in the previous chapter.

Without such a concrete instance, the framework of [Tes15] remains of no avail. As soon as one wants to use the framework in any way, one fundamentally has to instantiate the Boolean functions modeled as ideal functionalities by efficiently computable functions. Clearly, the above mentioned bound in the ideal model does not say anything about any concrete instance. Tessaro phrases this situation as follows:

> Heuristically, however, one hopes for even more: Namely, that under a careful implementation of the underlying component, the construction retains the promised security level. [Tes15]

[Tes15] Tessaro, "Optimally Secure Block Ciphers from Ideal Primitives"

[Hoa+12] Hoang et al., "An Enciphering Scheme Based on a Card Shuffle"

[Vau12] Vaudenay, *The End of Encryption based on Card Shuffling*

There has actually been one instance [Hoa+12] of the previous construction, but this has been broken almost instantaneously and completely, as parts of the encryption function were actually linear, see [Vau12]. This failure to securely instantiate the construction points to an important hurdle. Namely, proving the generic bounds and analysing the security of an instance are technically very different tasks. The security of any block cipher is with the current state of knowledge always the security against known attacks. In particular, when designing any concrete block cipher, one has to argue why linear and differential attacks do not threaten the construction.

▸ Our Contribution

Consequently, we investigate the important, but so far overlooked, aspect of instantiating the WSN construction with a practical secure instance. Practical secure meaning, just like in the case of AES, that the block cipher resists all known attacks. We denote this instance as Bison.[1] Our insights presented here are twofold.

First, we derive some inherent restrictions on the choice of the round function f_i. In a nutshell, we show that f_i has to be rather strong, in the sense that its output bit has to basically depend on all input bits. Moreover, we show that using less than n rounds will always result in an insecure construction. Those, from a cryptanalytic perspective rather obvious, results are presented in Section 4.3. Again, but from a different angle, this situation is in sharp contrast to key-alternating ciphers. In the case of key-alternating ciphers, even with a rather small number of rounds (e. g. ten in the case of AES-128) and rather weak round functions (in case of the AES round function any output bit depends on 32 input bits only and the whole round function decomposes into four parallel functions on 32 bits each) we get ciphers that provide, to the best of our knowledge today and after a significant amount of cryptanalysis, full security.

Second, despite those restrictions of the WSN construction, that have significant impact on the performance of any instance, there are very positive aspects of the WSN construction as well. In Section 4.4, we first define a family of WSN instances which fulfill our initial restrictions.

As we will show in detail, this allows to argue very convincingly that our instance is secure against differential attacks. Indeed, under standard assumptions, we can show that the probability of any (non-trivial) *differential* is upper bounded by 2^{-n+1} where n is the block size, a value that is close to the ideal case. This significantly improves upon what is the state of the art for key-alternating ciphers. As finding a solution to Problem 1 (Differentials) is notoriously hard, normally one therefore has to restrict to bounding the

[1] For "Bent whItened Swap Or Not"

probability of *differential characteristics* only. Whereas our results for differential cryptanalysis gives a much simpler solution to Problem 1. Additionally they can be of independent interest in the analysis of maximally unbalanced Feistel networks or nonlinear feedback shift registers.

We specify our concrete instance as a family of block ciphers for varying input length in Section 4.5. In our instance, we attach importance to simplicity and mathematical clarity. It is making use of bent functions for instantiating f and Linear Feedback Shift Registers (LFSRs) for generating the round keys. Another advantage of Bison is that it defines a whole family of block ciphers, one for any odd block size. In particular it allows the straightforward definition of small scale variants to be used for experiments.

Finally we discuss various other attacks and argue why they do not pose a threat for Bison in Section 4.6. Particularly the discussion on algebraic attacks might be of independent interest. For this we analyse the growth of the algebraic degree over the rounds. In contrast to what we intuitively expect – an exponential growth (until a certain threshold) as in the case for SPNs [Bou+11] – the degree of the WSN construction grows linearly in the degree of the round function f_i. This result can also be applied in the analysis of maximally unbalanced Feistel networks or nonlinear feedback shift registers.

[Bou+11] Boura et al., "Higher-order differential properties of Keccak and Luffa"

Additionally to the results in [Can+19a], we given an even block length instance, Wisent, of the WSN construction, see Section 4.8. We conclude this section by adjusting our security and implementation analysis to this new instance.

[Can+19a] Canteaut et al., "Bison – Instantiating the Whitened Swap-Or-Not Construction"

► Related Work

The first cipher, a Feistel structure, that allowed similarly strong arguments against differential attacks was presented by Nyberg and Knudsen [NK95], see also [Nyb12] for a nice survey on the topic. This cipher was named CRADIC, as Cipher Resistant Against DIfferential Cryptanalysis but is often simply referenced as the KN cipher. However, the cipher has been broken quickly afterwards, with the invention of interpolation attacks [JK97]. Another, technically very different approach to get strong results on resistance against attacks we would like to mention is the decorrelation theory [Vau98]. Interestingly, both previous approaches rely rather on one strong component, i. e. round function, to ensure security, while the WSN approach, and in particular Bison, gains its resistance against differential attacks step by step.

[NK95] Nyberg and Knudsen, "Provable Security Against a Differential Attack"

[Nyb12] Nyberg, ""Provable" Security against Differential and Linear Cryptanalysis (Invited Talk)"

[JK97] Jakobsen and Knudsen, "The Interpolation Attack on Block Ciphers"

[Vau98] Vaudenay, "Provable Security for Block Ciphers by Decorrelation"

Regarding the analysis of differentials, extensive efforts have been expended to evaluate the MEDP/MELP of SPN ciphers, and in particular of the AES. Some remarkable results were published by [Par+03] and then subsequently improved by [KS07] with a sophisticated pruning algorithm. Interestingly, further work by [DR06] and later by [CR15] revealed that such bounds are not invariant under affine transformations. All these works stress out how difficult it is to evaluate the MEDP/MELP of SPNs, even for a small number of rounds. On the contrary, and as we are going to elaborate in the remaining of this paper, computing the MEDP of Bison is rather straightforward and independent of the exact details of the components. This can be compared to the wide trail strategy that, making use of the branch number and the superbox argument, allows bounding the probability

[Par+03] Park et al., "Improving the Upper Bound on the Maximum Differential and the Maximum Linear Hull Probability for SPN Structures and AES"

[KS07] Keliher and Sui, "Exact maximum expected differential and linear probability for two-round Advanced Encryption Standard"

[DR06] Daemen and Rijmen, "Understanding Two-Round Differentials in AES"

[CR15] Canteaut and Roué, "On the Behaviors of Affine Equivalent Sboxes Regarding Differential and Linear Attacks" of any differential characteristic for a large class of SPNs. Our arguments allow to bound the differential probability for a large class of WSN instances.

Before discussing our analysis in detail, we shortly recapitulate the WSN construction.

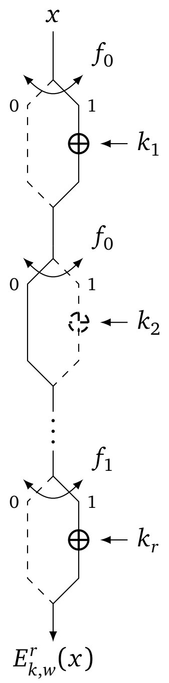


FIGURE 4.1: Schematic view of the WSN construction.

4.2 THE WHITENED SWAP-OR-NOT CONSTRUCTION

The WSN construction is defined as follows. Given two round keys k_i, w_i, the i-th round R_{k_i,w_i} computes

$$R_{k_i,w_i} : \mathbb{F}_2^n \to \mathbb{F}_2^n$$
$$R_{k_i,w_i}(x) := x + f_{b(i)}(w_i + \max\{x, x + k_i\}) \cdot k_i$$

where $f_{0,1} : \mathbb{F}_2^n \to \mathbb{F}_2$ are modeled as two ideal random functions, the max function returns the lexicographic biggest value in the input set. The index $b(i)$ equals zero for the first half of the rounds and one for the second half (see Figure 4.1 for a graphical overview of the encryption process).

In the remainder of the paper, we denote by $E^r_{k,w}(x)$ the application of r rounds of the construction to the input x with round keys k_i and w_i derived from the master key (k, w). Every round is involutory, thus for decryption one only has to reverse the order of the round keys.

Note that the usage of the maximum function is not decisive; it can be replaced by any function Φ_k that returns a unique representative of the set $\{x, x + k\}$, see [Tes15]. In other words it can be replaced by any function such that $\Phi_k(x) = \Phi_k(y)$ if and only if $y \in \{x, x + k\}$.

The main result given by Tessaro on the security of the WSN is the following (we only cite the informal statement, as our focus is not on the provable security aspect, but on practical instances).

Proposition 94 (Security of the WSN construction (Informal) [Tes15]). *The WSN construction with $\Theta(n)$ rounds*[2] *is $(2^{n-\mathcal{O}(\log n)}, 2^{n-\mathcal{O}(1)})$-secure.*

Thus, any adversary trying to distinguish the WSN construction from a random permutation and making at most $2^{n-\mathcal{O}(\log n)}$ queries to the block cipher and $2^{n-\mathcal{O}(1)}$ queries to the underlying function has negligible advantage. Here, the round keys are modeled as independent and uniformly distributed random variables.

While Tessaro's result gives us a strong security guarantee, the generic construction is not a practical instance. Any practical block cipher has to provide security arguments against known attacks and, even more important, it has to give concrete details about the building blocks involved – in this case the key schedule, the Boolean functions f_0 and f_1, and the actual number of rounds used. To fill this gap between theory and practice here, we now analyse which generic properties any WSN instance has to fulfill, and then give concrete instances of it.

4.3 INHERENT RESTRICTIONS

In this section we point out two inherent restrictions on any practical secure instance. Those restrictions result in general conditions on both the

[2] Tessaro claims security for $\mathcal{O}(n)$ rounds and notes that $\Omega(n)$ rounds are necessary to achieve information theoretically indistinguishability, as for a single query the internal Boolean function f can only supply at most one bit of randomness.

minimal number of rounds to be used and general properties of the round functions $f_{b(i)}$. In particular, those insights are taken into account for BISON and WISENT. While these restrictions are rather obvious from a cryptanalytic point of view, they have a severe impact on the performance of any concrete instance. We discuss performance in more detail in Section 4.7.

4.3.1 *Number of Rounds*

As in every round of the cipher, we simply add (or not) the current round key k_i, the ciphertext can always be expressed as the addition of the plaintext and a (message dependent) linear combination of all round keys k_i. The simple but important observation to be made here is that, as long as the round keys do not span the full space, the block cipher is easily attackable.

Phrased in terms of linear cryptanalysis we start with the following lemma.

Lemma 95. *For any number of rounds $r < n$ there exists an element $u \in \mathbb{F}_2^n \setminus \{0\}$ such that*

$$\widehat{E^r_{k,w}}(u,u) = 2^n,$$

that is the equation

$$u \cdot x = u \cdot E^r_{k,w}(x)$$

holds for all $x \in \mathbb{F}_2^n$.

Proof. Let $k_1, \ldots, k_r$ be the round keys derived from k and denote by

$$U = \operatorname{Span}\{k_1, \ldots, k_r\}^\perp$$

the dual space of the space spanned by the round keys. As $r < n$ by assumption, the dimension of $\operatorname{Span}\{k_1, \ldots, k_r\}$ is smaller than n and thus $U \neq \{0\}$. Therefore, U contains a non-zero element

$$u \in \operatorname{Span}\{k_1, \ldots, k_r\}^\perp$$

and it holds that

$$u \cdot E^r_{k,w}(x) = u \cdot (x + \sum_{i=1}^{r} \lambda_i k_i) = u \cdot x + u \cdot \sum_{i=1}^{r} \lambda_i k_i = u \cdot x$$

concluding the proof.

Even more importantly, this observation leads directly to a known plaintext attack with very low data-complexity. Given a set of t plaintext/ciphertext (p_i, c_i) pairs, an attacker simply computes

$$V = \operatorname{Span}\{p_i + c_i \mid 1 \leqslant i \leqslant t\} \subseteq \operatorname{Span}\left\{k_j \,\middle|\, 1 \leqslant j \leqslant r\right\}.$$

Given $t > r$ slightly more pairs than rounds, and assuming that $p_i + c_i$ is uniformly distributed in $\operatorname{Span}\left\{k_j\right\}$ (otherwise the attack only gets even stronger)[3] implies that

$$V = \operatorname{Span}\left\{k_j\right\}$$

with high probability and V can be efficiently computed. Furthermore, as above $\dim(\operatorname{Span}\left\{k_j\right\})$ is at most r, we have $V^\perp \neq \{0\}$. Given any $u \neq 0$ in

[3] For example if, with high probability, the $p_i + c_i$ do not depend on one or more k_j's, the described attack can be extended to one or more rounds with high probability.

$V^{\perp}$ allows to compute one bit of information on the plaintext given only the ciphertext and particularly distinguish the cipher from a random permutation in a chosen-plaintext setting efficiently.

A similar argument shows the following:

Lemma 96. *For any number of rounds r smaller than $2n-3$ there exist nonzero α and β, such that*

$$\widehat{E^r_{k,w}}(\alpha,\beta)=0\,.$$

Proof. We restrict to the case $r \geqslant n$ as otherwise the statement follows directly from the lemma above. Indeed, from Parseval's equality, the fact that $\widehat{E^r_{k,w}}(\alpha,\alpha)=2^n$ implies that $\widehat{E^r_{k,w}}(\alpha,\beta)=0$ for all $\beta \neq \alpha$. Let $k_1,\dots,k_r$ be the round keys derived from k and choose non-zero elements $\alpha \neq \beta$ such that

$$\alpha \in \operatorname{Span}\{k_1,\dots,k_{n-2}\}^{\perp} \qquad \text{and} \qquad \beta \in \operatorname{Span}\{k_{n-1},\dots,k_r\}^{\perp}.$$

Note that, as $r \leq 2n-3$ by assumption such elements always exist. Next, we split the encryption function in two parts, the first $n-2$ rounds E_1 and the remaining $r-(n-2)<n$ rounds E_2, i.e.

$$E^r_{k,w}=E_2 \circ E_1.$$

We can compute the Fourier coefficient of $E^r_{k,w}$ as

$$\widehat{E^r_{k,w}}(\alpha,\beta)=\sum_{\gamma\in\mathbb{F}_2^n}\frac{\widehat{E_1}(\alpha,\gamma)}{2^n}\cdot\frac{\widehat{E_2}(\gamma,\beta)}{2^n}.$$

Now, the above lemma and the choices of α and β imply that $\widehat{E_1}(\alpha,\gamma)=0$ for $\gamma \neq \alpha$ and $\widehat{E_2}(\gamma,\beta)=0$ for $\gamma \neq \beta$. Recalling that $\alpha \neq \beta$ by construction concludes the proof.

However, as the masks α and β depend on the key, and unlike above there does not seem to be an efficient way to compute those, we do not see a direct way to use this observation for an attack.

Summarising the observations above, we get the following conclusion:

Rationale 1. *Any practical instance must iterate at least n rounds. Furthermore, it is beneficial if any set of n consecutive round keys are linearly independent.*[4]

After having derived basic bounds on the number of rounds for any secure instance, we move on to criteria on the round function itself.

4.3.2 *Round Function*

Here, we investigate a very basic criterion on the round function, namely dependency on all input bits, when the round function of $E^r_{k,w}$ is defined by

$$R_{k_i,w_i}(x)=x+f_{b(i)}(w_i+\max\{x,x+k_i\})\cdot k_i\,.$$

Given the Boolean functions $f_{b(i)}:\mathbb{F}_2^n\to\mathbb{F}_2$, the question we would like to discuss is if it is necessary that the output bit of $f_{b(i)}$ has to depend on all

[4] If (some) round keys are linearly dependent, Lemmata 95 and 96 can easily be extended to more rounds.

input bits. The function $f_{b(i)}$ depends on an input bit j if there are two inputs x, x' differing only in the jth bit such that $f_{b(i)}(x) \neq f_{b(i)}(x')$. Otherwise the function is independent of the jth bit and we get

$$f_{b(i)}(x) = f_{b(i)}(x + e_j)$$

for all x.

We denote by $N(x) := \{i \mid x[i] = 1\}$ the index set of 1-bits in x, and by $\nu(x) := \max N(x)$ the index of the highest 1-bit in x, in other words $\nu(x) = \lfloor \log_2(x) \rfloor$, when interpreting $x \in \mathbb{F}_2^n$ as an integer. For the main observation on this criterion, we first need the following lemma.

Lemma 97. *Let $x, \delta \in \mathbb{F}_2^n$ and k uniformly randomly drawn from $\mathbb{F}_2^n$. Then*

$$\Pr[\max\{x + \delta, x + \delta + k\} = \max\{x, x + k\} + \delta] \geqslant 1 - 2^{\nu(\delta) - n}.$$

Proof. The equality depends on the highest bit of δ where x and $x + k$ differ, which is basically $\nu(k)$. We have

$$\Pr[\max\{x + \delta, x + \delta + k\} = \max\{x, x + k\} + \delta] = \Pr[\delta[\nu(k)] = 0],$$

which can also be written as

$$\Pr[\delta[\nu(k)] = 0] = 1 - \Pr[\nu(k) \in N(\delta)] = 1 - \sum_{i \in N(\delta)} \Pr[\nu(k) = i].$$

Further we have $\Pr[\nu(k) = i] = 2^{i-n-1}$ and thus

$$1 - \sum_{i \in N(\delta)} \Pr[\nu(k) = i] = 1 - \sum_{i \in N(\delta)} 2^{i-n-1} \geqslant 1 - 2^{\nu(\delta) - n},$$

which concludes the proof.

As we will see next, the functions $f_{b(i)}$ have to depend virtually on all *linear combinations of bits*. In other words, it is required that the functions $f_{b(i)}$ have no (non-trivial) derivative equal to the all-zero function.

Lemma 98. *If there exists a $\delta \in \mathbb{F}_2^n$ such that*

$$f_{b(i)}(x) = f_{b(i)}(x + \delta)$$

for all x and $i \in \{0, 1\}$ then

$$\Pr\left[E^r_{k,w}(x) + E^r_{k,w}(x + \delta) = \delta\right] \geqslant \left(1 - 2^{\nu(\delta) - n}\right)^r$$

where the probability is over the input x and the keys k and w.

Proof. From Lemma 97 we have that

$$\Pr[\max\{x + \delta, x + \delta + k\} = \max\{x, x + k\} + \delta] \geqslant 1 - 2^{\nu(\delta) - n}.$$

Now, we get for one round

$$R_{k_i, w_i}(x) = x + f_{b(i)}(w_i + \max\{x, x + k_i\}) \cdot k_i$$

by the assumption that $f_{b(i)}(x) = f_{b(i)}(x + \delta)$ for all x

$$R_{k_i, w_i}(x + \delta) = R_{k_i, w_i}(x) + \delta$$

with the same probability. Thus, for r rounds and uniformly chosen keys, we get

$$\Pr\left[E^r_{k,w}(x) + E^r_{k,w}(x+\delta) = \delta\right] \geqslant \left(1 - 2^{\nu(\delta)-n}\right)^r$$

by induction.

As an example, considering the case $n = 128$, r rounds, and both f_0 and f_1 that do not depend on the most significant byte. Thus, we can choose $\delta = e_{121}$ as a unit vector with $\nu(\delta) = 121$ and get a differential probability of

$$\Pr\left[E^r_{k,w}(x) + E^r_{k,w}(x+\delta) = \delta\right] \geqslant \left(1 - 2^{121-128}\right)^r \approx (0.36)^{r/n}$$

which would completely compromise the scheme for a reasonable number of rounds. In general this shows that as long as both $f_{b(i)}$ do not depend on almost all bits, the scheme is immediately broken by differential cryptanalysis. Now, one might hope that one could craft functions f_0 and f_1 where, e. g. f_0 depends only on the first $\frac{n}{2}$ bits and f_1 on the last $\frac{n}{2}$ bits to overcome this restriction. However, while such a construction might be secure against basic differential cryptanalysis, it would still be completely broken by boomerang attacks [Wag99]. The main idea of boomerang attacks is to split the whole block cipher in two parts such that one has a high probable differential for the first part and a second high probable differential for the second part, which is exactly the situation one would end up here.

[Wag99] Wagner, "The Boomerang Attack"

Thus, both functions independently have to virtually depend on all input bits, and we deduce the following.

Rationale 2. *For a practical instance, the functions $f_{b(i)}$ has to depend on all bits. Even more, for any $\delta \in \mathbb{F}_2^n$ the probability of*

$$f_{b(i)}(x) = f_{b(i)}(x+\delta)$$

should be close to $\frac{1}{2}$.

It is worth noticing that the analysis leading to this rationale applies to the original round function. However, as pointed out in [Tes15, Section 3.1], in the definition of the round function, we can replace the function

[Tes15] Tessaro, "Optimally Secure Block Ciphers from Ideal Primitives"

$$x \mapsto \max\{x, x+k\}$$

by any function Φ_k such that $\Phi_k(x) = \Phi_k(x+k)$ for all x. While the following sections will focus on the case when Φ_k is linear, we proved that Rationale 2 is also valid in this other setting.

Again, this should be compared to key-alternating ciphers, where usually not all output bits of a single round function depend on all input bits. For example for AES any output bit after one round depends only on 32 input bits and for PRESENT any output bit only depends on 4 input bits. However, while for key-alternating ciphers this does not seem to be problematic, and indeed allows rather weak round functions to result in a secure scheme, for the WSN construction the situation is very different.

Before specifying our exact instance, we want to discuss differential cryptanalysis of a broader family of instances.

4.4 DIFFERENTIAL CRYPTANALYSIS OF BISON-LIKE INSTANCES

We coin an instance of the WSN construction "BISON-like", if it iterates at least n rounds with linearly independent round keys $k_1, \dots, k_n$ and applies Boolean functions $f_{b(i)}$ that depend on all bits, i. e. fulfill Rationale 2. As explained in [Tes15, Section 3.1], in order to enable decryption it is required that the Boolean functions $f_{b(i)}$ return the same result for both x and $x + k$. In the original proposition by Tessaro, this is achieved by using the max function in the definition of the round function. Using this technique reduces the number of possible inputs for the $f_{b(i)}$ to 2^{n-1}. To simplify the analysis and to ease notation, we replace the max function with a *linear function* $\Phi_k : \mathbb{F}_2^n \to \mathbb{F}_2^{n-1}$ with $\operatorname{Ker} \Phi_k = \{0, k\}$. From now on, we assume that any BISON-like instance uses such a Φ_k instead of the max function. The corresponding round function has then the following form

$$R_{k_i,w_i}(x) := x + f_{b(i)}\big(w_i + \Phi_{k_i}(x)\big)k_i. \tag{4.1}$$

With the above conditions, any BISON-like instance of the WSN construction inherits f's resistance to differential cryptanalysis, as we show in the remainder of this section.

For our analysis, we make two standard assumptions in symmetric cryptanalysis as mentioned above: the *independence of whitening round keys* w_i and the *hypothesis of stochastic equivalence* with respect to these round keys. That is, we assume round keys w_i to be independently uniformly drawn and the resulting EDP to equal the differential probabilities averaged over all w. In the following sections, we will argue why these assumptions do fit to our design and back up the results by practical experiments (see EXPERIMENTAL RESULTS in Section 4.6.3 and [Can+18, Appendix B]). For the round keys k_i we do not have to make such assumptions.

[Can+18] Canteaut et al., BISON – *Instantiating the Whitened Swap-Or-Not Construction (Full Version)*

We first discuss the simple case of differential behaviour for one round only and then move up to an arbitrary number of rounds and devise the number of possible output differences and their probabilities.

4.4.1 *From One-Round Differential Characteristics*

Looking only at one round, we can compute the DDT explicitly:

Proposition 99. *Let $R_{k_i,w_i} : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be the WSN round function as in Eq. (4.1). Then its DDT consists of the entries*

$$DDT[\alpha,\beta] = \begin{cases} 2^{n-1} + \mathcal{A}_f(\Phi_k(\alpha)) & \text{if } \beta = \alpha \\ 2^{n-1} - \mathcal{A}_f(\Phi_k(\alpha)) & \text{if } \beta = \alpha + k \\ 0 & \text{otherwise} \end{cases}. \tag{4.2}$$

$\mathcal{A}_f(\cdot)$ denotes the autocorrelation of f.

Most notably, if f is bent, we have

$$DDT[\alpha,\beta] = \begin{cases} 2^n & \text{if } \alpha = \beta = k \text{ or } \alpha = \beta = 0 \\ 2^{n-1} & \text{if } \beta \in \{\alpha, \alpha + k\} \text{ and } \alpha \notin \operatorname{Span}\{k\} \\ 0 & \text{otherwise} \end{cases}.$$

Proof. We have to count the number of solutions of $R(x)+R(x+\alpha)=\beta$:

$$\begin{aligned}\mathrm{DDT}[\alpha,\beta] &= \left|\left\{x\in\mathbb{F}_2^n \,\middle|\, R(x)+R(x+\alpha)=\beta\right\}\right| \\ &= \left|\left\{x\in\mathbb{F}_2^n \,\middle|\, \alpha+[f(w+\Phi_k(x))+f(w+\Phi_k(x+\alpha))]\cdot k=\beta\right\}\right|\end{aligned}$$

Since f takes its values in $\mathbb{F}_2$, the only output differences β such that $\mathrm{DDT}[\alpha,\beta]$ may differ from 0 are $\beta=\alpha$ and $\beta=\alpha+k$. When $\beta=\alpha$, we have

$$\begin{aligned}\mathrm{DDT}[\alpha,\alpha] &= \left|\left\{x\in\mathbb{F}_2^n \,\middle|\, f(w+\Phi_k(x))+f(w+\Phi_k(x+\alpha))=0\right\}\right| \\ &= \left|\left\{x\in\mathbb{F}_2^n \,\middle|\, f(w+\Phi_k(x))+f(w+\Phi_k(x)+\Phi_k(\alpha))=0\right\}\right| \\ &= 2\cdot\left|\left\{x'\in\mathbb{F}_2^{n-1} \,\middle|\, f(x')+f(x'+\Phi_k(\alpha))=0\right\}\right| \\ &= 2\left(2^{n-2}+\frac{1}{2}\widehat{\Delta_{\Phi_k(\alpha)}(f)}(0)\right)=2^{n-1}+\mathcal{A}_f(\Phi_k(\alpha))\,.\end{aligned}$$

Similarly,

$$\begin{aligned}\mathrm{DDT}[\alpha,\alpha+k] &= \left|\left\{x\in\mathbb{F}_2^n \,\middle|\, f(w+\Phi_k(x))+f(w+\Phi_k(x+\alpha))=1\right\}\right| \\ &= 2\left(2^{n-2}-\frac{1}{2}\widehat{\Delta_{\Phi_k(\alpha)}(f)}(0)\right)=2^{n-1}-\mathcal{A}_f(\Phi_k(\alpha))\,.\end{aligned}$$

Most notably, when $\alpha\in\{0,k\}$, $\mathcal{A}_f(\Phi_k(\alpha))=\mathcal{A}_f(0)=2^{n-1}$. Moreover, when f is bent, $\mathcal{A}_f(\Phi_k(\alpha))=0$ for all other values of α.

4.4.2 *To Differentials over more Rounds*

As previously explained, it is possible to estimate the probability of a differential characteristic over several rounds, averaged over the round keys, when the cipher is a Markov cipher. We now show that this assumption holds for any BISON-like instance of the WSN construction.

Lemma 100. *Let $R_{k,w}:\mathbb{F}_2^n\to\mathbb{F}_2^n$ be the WSN round function as in Eq. (4.1). For any fixed $k\in\mathbb{F}_2^n$ and any differential $\alpha\rightrightarrows_{\beta}$, we have that*

$$\Pr_w\left[R_{k,w}(x+\alpha)+R_{k,w}(x)=\beta\right]$$

is independent of x. More precisely

$$\Pr_w\left[R_{k,w}(x+\alpha)+R_{k,w}(x)=\beta\right]=\Pr_x\left[R_{k,w}(x+\alpha)+R_{k,w}(x)=\beta\right]\,.$$

Proof. We have

$$\begin{aligned}&\left\{w\in\mathbb{F}_2^{n-1} \,\middle|\, \Delta_\alpha(R_{k,w})(x)=\beta\right\} \\ &=\left\{w\in\mathbb{F}_2^{n-1} \,\middle|\, \left(\Delta_{\Phi_k(\alpha)}(f)(w+\Phi_k(x))\right)\cdot k=\alpha+\beta\right\} \\ &=\begin{cases}\emptyset & \text{if } \beta\notin\{\alpha,\alpha+k\} \\ \Phi_k(x)+\mathrm{Supp}\left(\Delta_{\Phi_k(\alpha)}(f)\right) & \text{if } \beta=\alpha+k \\ \Phi_k(x)+\left(\mathbb{F}_2^{n-1}\setminus\mathrm{Supp}\left(\Delta_{\Phi_k(\alpha)}(f)\right)\right) & \text{if } \beta=\alpha\end{cases}\,.\end{aligned}$$

Clearly, the cardinality of this set does not depend on x. Moreover, this cardinality, divided by 2^{n-1}, corresponds to the value of

$$\Pr_x\left[R_{k,w}(x+\alpha)+R_{k,w}(x)=\beta\right]$$

computed in the previous proposition.

By induction on the number of rounds, we then directly deduce that any BISON-like instance of the WSN construction is a Markov cipher in the sense of the following corollary.

Corollary 101. *Let $E^i_{k,w}$ denote i rounds of a* BISON*-like instance of the WSN construction with round function R_{k_i,w_i}. For any number of rounds r and any round keys $(k_1,\ldots,k_r)$, the probability of an r-round characteristic θ satisfies*

$$\begin{aligned}&\Pr_w\left[E^i_{k,w}(x)+E^i_{k,w}(x+\theta_0)=\theta_i, \forall 1\leqslant i\leqslant r\right]\\ = &\prod_{i=1}^{r}\Pr_x\left[R_{k_i,w_i}(x)+R_{k_i,w_i}(x+\theta_{i-1})=\theta_i\right].\end{aligned}$$

For many ciphers several differential characteristics can cluster in a *differential* over more rounds. This is the main reason why bounding the probability of differentials is usually very difficult if possible at all. For BISON-like instances the situation is much nicer; we can actually compute the EDP, i. e. the probabilities of the differentials averaged over all whitening key sequences $(w_1,\ldots,w_r)$. This comes from the fact that any differential for less than n rounds contains at most one differential characteristic with non-zero probability. To understand this behaviour, let us start by analysing the EDP (averaged over the w_i) and by determining the number of possible output differences.

In the following, we assume that the input difference α is fixed, and we calculate the number of possible output differences. We show that this quantity depends on the relation between α and the k_i.

Lemma 102. *Let us consider r rounds of a* BISON*-like instance of the WSN construction with round function involving Boolean functions $f_{b(i)}$ having no (non-trivial) constant derivative. Assume that the first n round keys $k_1,\ldots,k_n$ are linearly independent, and that $k_{n+1}=k_1+\sum_{i=2}^{n}\gamma_i k_i$ for $\gamma_i\in\mathbb{F}_2$. For any non-zero input difference α, the number of possible output differences β such that*

$$\Pr_{w,x}\left[E^r_{k,w}(x+\alpha)+E^r_{k,w}(x)=\beta\right]\neq 0$$

is

$$\begin{cases}2^r & \text{if } \alpha\notin\operatorname{Span}\{k_i\} \text{ and } r<n,\\ 2^r-2^{r-\ell} & \text{if } \alpha=k_\ell+\sum_{i=1}^{\ell-1}\lambda_i^\alpha k_i \text{ and } r\leqslant n,\\ 2^n-1 & \text{if } r>n.\end{cases}$$

Proof. By combining Corollary 101 and Proposition 99, we obtain that the average probability of a characteristic $(\theta_0,\theta_1,\ldots,\theta_{r-1},\theta_r)$ can be non-zero only if $\theta_i\in\{\theta_{i-1},\theta_{i-1}+k_i\}$ for all $1\leqslant i\leqslant r$. Therefore, the output difference θ_r must be of the form $\theta_r=\theta_0+\sum_{i=1}^{r}\lambda_i k_i$ with $\lambda_i\in\mathbb{F}_2$. Moreover, for those characteristics, the average probability is non-zero unless there exists $1\leqslant i<r$ such that $\left|\mathcal{A}_{f_{b(i)}}(\Phi_{k_i}(\theta_i))\right|=2^{n-1}$, i. e. $\Delta_{\Phi_{k_i}(\theta_i)}(f_{b(i)})$ is constant. By hypothesis, this only occurs when $\theta_i\in\{0,k_i\}$, and the impossible characteristics correspond to the case when either $\theta_i=0$ or $\theta_{i+1}=0$. It follows that the valid characteristics are exactly the characteristics of the form

$$\theta_i=\theta_0+\sum_{j=1}^{i}\lambda_j k_j$$

where none of the θ_i vanishes.

- When the input difference $\alpha \notin \operatorname{Span}\{k_i\}$, for any given output difference $\beta = \alpha + \sum_{i=1}^{r} \lambda_i k_i$, the r-round characteristic

  $$(\alpha, \alpha + \lambda_1 k_1, \alpha + \lambda_1 k_1 + \lambda_2 k_2, \ldots, \alpha + \sum_{i=1}^{r} \lambda_i k_i)$$

  is valid since none of the intermediate differences vanishes.

- When $r \leqslant n$ and $\alpha = k_\ell + \sum_{i=1}^{\ell-1} \lambda_i^\alpha k_i$, the only possible characteristic from α to $\beta = \alpha + \sum_{i=1}^{r} \lambda_i k_i$ satisfies

  $$\theta_j = \begin{cases} \sum_{i=1}^{j} (\lambda_i + \lambda_i^\alpha) k_i + \sum_{i=j+1}^{\ell} \lambda_i^\alpha k_i & \text{if } j \leqslant \ell \\ \sum_{i=1}^{\ell} (\lambda_i + \lambda_i^\alpha) k_i + \sum_{i=\ell+1}^{j} \lambda_i k_i & \text{if } j > \ell \ . \end{cases}$$

  Since the involved round keys are linearly independent, we deduce that $\theta_j = 0$ only when $j = \ell$ and $\lambda_i = \lambda_i^\alpha$ for all $1 \leqslant i \leqslant \ell$. It then follows that there exists a valid characteristic from α to β unless $\lambda_i = \lambda_i^\alpha$ for all $1 \leqslant i \leqslant \ell$. The number of possible outputs β is then

  $$(2^\ell - 1) 2^{r-\ell} = 2^r - 2^{r-\ell}.$$

- If we increase the number of rounds to more than n, we have $\alpha = k_\ell + \sum_{i=1}^{\ell-1} \lambda_i^\alpha k_i$ for some $\ell \leqslant n$. If $\beta = \alpha + \sum_{i=1}^{n} \lambda_i k_i$ with $\sum_{i=1}^{\ell} \lambda_i k_i \neq \alpha$, then we can obviously extend the previous n-round characteristic to

  $$(\alpha, \alpha + \lambda_1 k_1, \ldots, \alpha + \sum_{i=1}^{n-1} \lambda_i k_i, \beta, \beta, \ldots, \beta).$$

  If $\sum_{i=1}^{\ell} \lambda_i k_i = \alpha$, β cannot be the output difference of an n-round characteristic. However, the following $(n+1)$-round characteristic starting from $\theta_0 = \alpha$ is valid:

  $$\theta_j = \begin{cases} k_1 + \sum_{i=2}^{j} \gamma_i k_i + \sum_{i=j+1}^{\ell} \lambda_i^\alpha k_i & \text{if } j \leqslant \ell \\ k_1 + \sum_{i=2}^{j} \gamma_i k_i + \sum_{i=\ell+1}^{j} \lambda_i k_i & \text{if } \ell < j \leqslant n \\ \beta & \text{if } j = n+1 \end{cases}$$

  Indeed, $\theta_n = \beta + k_n$ implying that the last transition is valid. Moreover, it can be easily checked that none of these θ_j vanishes, unless $\beta = 0$. This implies that all non-zero output differences β are valid.

The last case in the above lemma is remarkable, as it states any output difference is possible after $n+1$ rounds. To highlight this, we restate it as the following corollary.

Corollary 103. *For* BISON*-like instances with more than n rounds whose round keys $k_1, \ldots, k_{n+1}$ satisfy the hypothesis of Lemma 102, and for any non-zero input difference, every non-zero output difference is possible.*

We now focus on a reduced version of the cipher limited to exactly n rounds and look at the probabilities for every possible output difference. Most notably, we exhibit in the following lemma an upper-bound on the

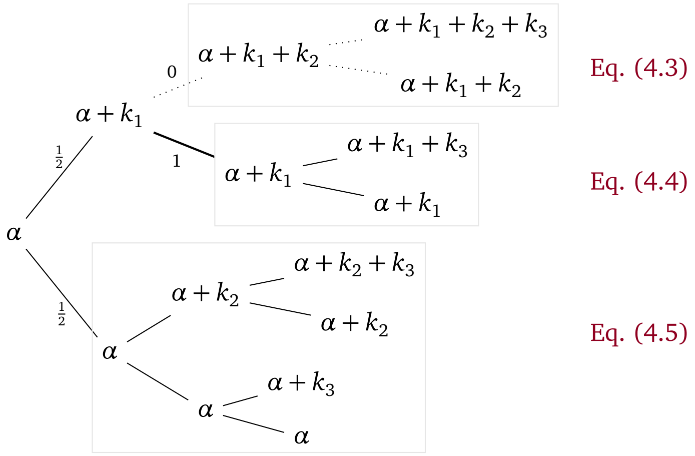


FIGURE 4.2: Probabilities of output differences for three rounds and the cases of the input difference $\alpha = k_1 + k_2$, thus $\ell = 2$. Dotted transitions are impossible.

MEDP which is minimised when n is odd and the involved Boolean functions $f_{b(i)}$ are bent. In other words, Rationale 2 which was initially motivated by the analysis in Section 4.3 for the original round function based on $x \mapsto \max(x, x + k)$ [Tes15] is also valid when a linear function Φ_k is used.

[Tes15] Tessaro, "Optimally Secure Block Ciphers from Ideal Primitives"

Lemma 104. *Let us consider n rounds of a* BISON*-like instance of the WSN construction with round function involving Boolean functions $f_{b(i)}$. Let $k_1, \dots, k_n$ be any linearly independent round keys. Then, for any input difference $\alpha \neq 0$ and any β, we have*

$$\begin{aligned} EDP(\alpha, \beta) &= \Pr_{w,x}\left[E_{k,w}(x + \alpha) + E_{k,w}(x) = \beta\right] \\ &\leqslant \left(\frac{1}{2} + 2^{-n} \max_{1 \leqslant i \leqslant n} \mathcal{M}\left(f_{b(i)}\right)\right)^{n-1} . \end{aligned}$$

$\mathcal{M}(f)$ denotes the absolute indicator of f.

More precisely, if all $f_{b(i)}$ are bent*,*

$$EDP(\alpha, \beta) = \begin{cases} 0 & \text{if } \beta = \sum_{i=\ell+1}^{n} \lambda_i k_i, & (4.3) \\ 2^{-n+1} & \text{if } \beta = k_\ell + \sum_{i=\ell+1}^{n} \lambda_i k_i, & (4.4) \\ 2^{-n} & \text{otherwise}, & (4.5) \end{cases}$$

where ℓ denotes as previously the latest round key that appears in the decomposition of α into the basis $(k_1, \dots, k_n)$, that is $\alpha = k_\ell + \sum_{i=1}^{\ell-1} \lambda_i k_i$.

The case of bent functions is visualised in Figure 4.2, where we give an example of the three possibilities for three rounds.

Proof. As proved in Lemma 102, (α, β) is an impossible differential if and only if $\beta = \sum_{i=\ell+1}^{n} \lambda_i k_i$. For all other values of $\beta = \alpha + \sum_{i=1}^{n} \lambda_i k_i$, we have

$$\mathrm{EDP}(\alpha, \beta) = \prod_{i=1}^{n} \left(\frac{1}{2} + (-1)^{\lambda_i} 2^{-n} \mathcal{A}_{f_{b(i)}}(\Phi_{k_i}(\theta_i))\right)$$

where $\theta_i = \alpha + \sum_{j=1}^{i} \lambda_j k_j$. The i-th term in the product is upper bounded by

$$\frac{1}{2} + 2^{-n} \max_{1 \leqslant i \leqslant n} \mathcal{M}\left(f_{b(i)}\right)$$

except if $\Phi_{k_i}(\theta_i) = 0$, i. e. $\theta_i \in \{0, k_i\}$. As seen in Lemma 102, the case $\theta_i = 0$ cannot occur in a valid characteristic. The case $\theta_i = k_i$ occurs if and only if $i = \ell$ and $\beta = k_\ell + \sum_{j=\ell+1}^{n} \lambda_j k_j$. In this situation, the ℓ-th term in the product equals one. In the tree of differences this is visible as the collapsing of the two branches from two possible succeeding differences into only one, which then of course occurs with probability one, see upper branch of Figure 4.2.

Most notably, all $f_{b(i)}$ are bent if and only if

$$\max_{1 \leqslant i \leqslant n} \mathcal{M}\left(f_{b(i)}\right) = 0\,,$$

leading to the result.

This can be seen on Figure 4.2: the $2^{n-\ell}$ possible differences coming from the collapsed branch have a transition of probability one in that round, resulting in an overall probability of 2^{-n+1}, see Eq. (4.4). For the lower part of Figure 4.2, all the other differences are not affected by this effect and have a probability of 2^{-n}, see Eq. (4.5).

Because they allow us to minimise the MEDP, we now concentrate on the case of bent functions for the sake of simplicity, which implies that the block size is odd. However, for implementation reasons an even block size is typically preferred. We discuss our BISON-like instance with even block sizes, WISENT, at the end of this chapter, see Section 4.8.

It would be convenient to assume in differential cryptanalysis that the EDP of a differential does not increase when adding more rounds, while this does not hold in general. However, this argument can easily be justified for BISON-like instances using bent functions, when averaging over the whitening keys w.

Proposition 105. *Let us consider $r \geqslant n$ rounds of a* BISON*-like instance of the WSN construction with bent functions $f_{b(i)}$. Let $k_1, \ldots, k_n$ be any linearly independent round keys. Then the probability of any non-trivial differential, averaged over all whitening key sequences w is upper bounded by 2^{-n+1}.*

In other words, the MEDP of BISON*-like instances with bent $f_{b(i)}$ for $r \geqslant n$ rounds is 2^{-n+1}.*

Proof. By induction over r. The base case for $r = n$ rounds comes from Lemma 104. In the induction step, we first consider the case when the output difference β after r rounds differs from k_r. Then the output difference $\theta_r = \beta$ can be reached if and only if the output difference after $(r-1)$ rounds θ_{r-1} belongs to $\{\beta, \beta + k_r\}$. Then,

$$\begin{aligned}
\mathrm{EDP}^r(\alpha,\beta) = {} & \Pr_{w_r}\left[R_{k_r,w_r}(x_r) + R_{k_r,w_r}(x_r+\beta) = \beta\right]\mathrm{EDP}^{r-1}(\alpha,\beta) \\
& + \Pr_{w_r}\left[R_{k_r,w_r}(x_r) + R_{k_r,w_r}(x_r+\beta+k_r) = \beta\right]\mathrm{EDP}^{r-1}(\alpha,\beta+k_r) \\
= {} & \frac{1}{2}\Big(\mathrm{EDP}^{r-1}(\alpha,\beta) + \mathrm{EDP}^{r-1}(\alpha,\beta+k_r)\Big) \leqslant 2^{-n+1}\,.
\end{aligned}$$

When the output difference β after r rounds equals k_r, it results from $\theta_{r-1} = k_r$ with probability 1. In this case

$$\mathrm{EDP}^r(\alpha,\beta) = \mathrm{EDP}^{r-1}(\alpha,\beta) \leqslant 2^{-n+1}$$

as required.

This bound is close to the ideal case, in which each differential has probability $1/(2^n - 1)$.

We now give a detailed description of our instance BISON.

4.5 SPECIFICATION OF BISON

As BISON should obviously be a specialisation of BISON-like instances, this concrete instance inherits the already specified parts. Thus BISON, replaces the max function by Φ_k, and uses a key schedule that generates round keys, where all n consecutive round keys are linearly independent. Further, as assumed at the end of the prior section, BISON uses two bent functions $f_{b(i)}$. The resulting instance for n bits iterates the WSN round function as defined below over $3 \cdot n$ rounds. The chosen number of rounds mainly stems from the analysis of the algebraic degree that we discuss in Section 4.6.

4.5.1 *Round function*

For any nonzero round key k, we define $\Phi_k : \mathbb{F}_2^n \to \mathbb{F}_2^{n-1}$ as

$$\Phi_k(x) := (x_{i(k)} \cdot k + x)[1, \dots, i(k) - 1, i(k) + 1, \dots, n], \qquad (4.6)$$

where $i(k)$ denotes the index of the lowest bit set to 1 in k, and the notation $x[1, \dots, j-1, j+1, \dots, n]$ returns the $(n-1)$-bit vector, consisting of the bits of x except the jth bit.

Lemma 106. *The function $\Phi_k : \mathbb{F}_2^n \to \mathbb{F}_2^{n-1}$ is linear and satisfies*

$$\operatorname{Ker} \Phi_k = \{0, k\}.$$

The proof can be done by simply computing both outputs for x and $x + k$. For the preimage of $y \in \mathbb{F}_2^{n-1}$ and $j = i(k)$ we have

$$\Phi_k^{-1}(y) \in \left\{ \begin{matrix} (y[1:j-1], 0, y[j:n-1]) + k[1:n], \\ (y[1:j-1], 0, y[j:n-1]) \end{matrix} \right\}. \qquad (4.7)$$

Due to the requirement for the $f_{b(i)}$ being bent, we are limited to functions taking an even number of bits as input. The simplest example of a bent function is the inner product.

Eventually we end up with the following instance of the WSN round.

BISON's Round Function

For round keys $k_i \in \mathbb{F}_2^n$ and $w_i \in \mathbb{F}_2^{n-1}$ the round function computes

$$R_{k_i,w_i}(x) := x + f_{b(i)}\big(w_i + \Phi_{k_i}(x)\big)k_i. \qquad (4.8)$$

where

- Φ_{k_i} is defined as in Eq. (4.6),
- $f_{b(i)}$ is defined as

$$f_{b(i)} : \mathbb{F}_2^{n-1} \to \mathbb{F}_2$$
$$f_{b(i)}(x) := x[1:(n-1)/2] \cdot x[(n+1)/2 : n-1] + b(i),$$

- and $b(i)$ is 0 if $i \leqslant \frac{r}{2}$ and 1 otherwise.

Security Claim 1 (BISON)**.** *We claim n-bit security for* BISON *in the single-key model. We emphasise that we do not claim any security in the related-key, chosen-key or known-key model.*

4.5.2 *Key schedule*

In the i-th round, the key schedule has to compute two round keys: $k_i \in \mathbb{F}_2^n$ and $w_i \in \mathbb{F}_2^{n-1}$. We compute those round keys as the states of LFSRs after i clocks, where the initial states are given by a master key K. The master key consists of two parts of n and $n-1$ bits, i. e.

$$K = (k, w) \in \mathbb{F}_2^n \times \mathbb{F}_2^{n-1}.$$

As the all-zero state is a fixed point for any LFSR, we *exclude the zero key* for both k and w. In particular $k = 0$ is obviously a weak key that would result in a ciphertext equal to the plaintext $p = E_{0,w}^r(p)$ for all p, independently of w or of the number of rounds r.

It is well-known that choosing a feedback polynomial of an LFSR to be primitive results in an LFSR of maximal period. Clocking the LFSR then corresponds to multiplication of its state with the companion matrix of this polynomial. Interpreted as elements from the finite field, this is the same as multiplying with a primitive element.

In order to avoid structural attacks, e. g. invariant attacks [Lea+11; Tod+16; Gra+16], as well as the propagation of low-weight inputs, we add round constants c_i to the round key w_i.

These round constants are also derived from the state of an LFSR with the same feedback polynomial as the w_i LFSR, initialised to the unit vector with the least significant bit set. To avoid synchronization with the w_i LFSR, the c_i LFSR clocks backwards.

BISON's Key Schedule

For two primitive polynomials $p_w(x), p_k(x) \in \mathbb{F}_2[x]$ with degrees $\deg(p_w) = n-1$ and $\deg(p_k) = n$ and the master key $K = (k, w) \in \mathbb{F}_2^n \times \mathbb{F}_2^{n-1}$, $k, w \neq 0$ the key schedule computes the i-th round keys as

$$\begin{aligned} &\mathrm{KS}_i : \mathbb{F}_2^n \times \mathbb{F}_2^{n-1} \to \mathbb{F}_2^n \times \mathbb{F}_2^{n-1} \\ &\qquad \mathrm{KS}_i(k, w) := (C(p_k)^i k, C(p_w)^{-i} e_1 + C(p_w)^i w) = (k_i, c_i + w_i) \end{aligned}$$

where $C(\cdot)$ is the companion matrix of the corresponding polynomial, and $0 \leqslant i < r$.

In [Can+18, Appendix A] we give concrete polynomials for $5 \leqslant n \leqslant 129$-bit block sizes.

As discussed above, this key schedule has the following property, see also Rationale 1.

Lemma 107. *For* BISON*'s key schedule, the following property holds: Any set of n consecutive round keys k_i are linearly independent. Moreover there exist coefficients λ_i such that*

$$k_{n+i} = k_i + \sum_{j=i+1}^{n+i-1} \lambda_j k_j.$$

Proof. To prove this, we start by showing that the above holds for the first n round keys, the general case then follows from a similar argumentation.

We need to show that there exists no non-trivial $\lambda_i \in \mathbb{F}_2$, $1 \leqslant i \leqslant n$, so that

$$\sum_{i=1}^{n} \lambda_i k_i = \sum_{i=1}^{n} \lambda_i C(p_k)^i k = 0\,,$$

which is equivalent to showing that there exists no non-trivial $\lambda_i \in \mathbb{F}_2$ such that

$$\sum_{i=0}^{n-1} \lambda_i C(p_k)^i k = 0\,.$$

In this regard, we recall the notion of *minimal polynomial of k with respect to* $C(p_k)$, defined as the monic polynomial of smallest degree $Q_L(k)(x) = \sum_{i=0}^{d} q_i x^i \in \mathbb{F}_2[x]$ such that $\sum_{i=0}^{d} q_i C(p_k)^i k = 0$. Referring to a discussion that has been done for instance in [Bei+17], we know that the minimal polynomial of k is a divisor of the minimal polynomial of $C(p_k)$. Since in our case our construction has been made so that this later is equal to p_k which is a primitive polynomial, we deduce that the minimal polynomial of $k \neq 0$ is p_k itself. Since the degree of p_k is equal to n, this prove that the first n keys are linearly independent.

[Bei+17] Beierle et al., "Proving Resistance Against Invariant Attacks: How to Choose the Round Constants"

The equation holds, since $p_k(0) = 1$.

4.6 SECURITY ANALYSIS OF BISON

As we have already seen, BISON is resistant to differential cryptanalysis. We next argue why BISON is also resistant to other known attacks. In particular we bound the potential of any linear trail over r rounds, analyse and bound the algebraic degree, discuss resistance against impossible differentials, zero correlation and invariant attacks, and analyse the related-key behaviour.

4.6.1 *Linear Cryptanalysis*

For linear cryptanalysis, given the fact that BISON is based on a bent function, i. e. a maximally non-linear function, arguing that no linear characteristic with high correlation exist is rather easy. Again, we start by looking at the Fourier coefficients for one round.

► FROM ONE ROUND

Using the properties of f being bent, we get the following.

Proposition 108. *Let $R_{k,w} : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be the round function as defined in* Eq. (4.8). *Then its LAT consists of the entries*

$$\widehat{R_{k,w}}(\alpha,\beta) = \begin{cases} 2^n & \text{if } \alpha = \beta \text{ and } \beta \cdot k = 0 \\ \pm 2^{\frac{n+1}{2}} & \text{if } \alpha \cdot k = 1 \text{ and } \beta \cdot k = 1 \\ 0 & \text{if } (\alpha + \beta) \cdot k = 1 \text{ or } (\alpha \neq \beta \text{ and } \beta \cdot k = 0) \end{cases} . \qquad (4.9)$$

Proof. First, we show the upper part, that is the values of the diagonal. Then

$$\begin{aligned}\widehat{R_{k,w}}(\alpha,\beta) &= \sum_{x\in\mathbb{F}_2^n} (-1)^{(\alpha+\beta)\cdot x+\beta\cdot k\cdot f(w+\Phi_k(x))}\\ &= \sum_{\substack{y\in\mathbb{F}_2^{n-1}\\ \Phi_k^{-1}(y)\in\{x_0,x_1\}}} (-1)^{(\alpha+\beta)\cdot x_0+\beta\cdot k\cdot f(w+y)} + (-1)^{(\alpha+\beta)\cdot x_1+\beta\cdot k\cdot f(w+y)}\end{aligned}$$

As we look at the diagonal elements, $\alpha=\beta$, we have

$$\widehat{R_{k,w}}(\alpha,\alpha) = 2\cdot\sum_{y}(-1)^{\beta\cdot k\cdot f(w+y)} = 2\cdot\sum_{y'\in\mathbb{F}_2^{n-1}}(-1)^{\beta\cdot k\cdot f(y')}\,.$$

Two possibilities remain: If $\beta\cdot k=0$, the exponent is always zero and thus $\widehat{R_{k,w}}(\alpha,\alpha)=2^n$. In the other case, $\beta\cdot k=1$ and

$$\widehat{R_{k,w}}(\alpha,\alpha) = 2\cdot\sum_{y'\in\mathbb{F}_2^{n-1}}(-1)^{f(y')} = \pm 2\cdot 2^{\frac{n-1}{2}}\,, \tag{4.10}$$

since Parseval's relation implies that all Fourier coefficients of an $(n-1)$-variable bent function have the same magnitude, namely $2^{\frac{n-1}{2}}$.

Now for the second part we have $\alpha\neq\beta$.

$$\begin{aligned}&\widehat{R_{k,w}}(\alpha,\beta)\\ &= \sum_{\substack{y\in\mathbb{F}_2^{n-1}\\ \Phi_k^{-1}(y)\in\{x_0,x_1\}}} (-1)^{(\alpha+\beta)\cdot x_0+\beta\cdot k\cdot f(w+y)} + (-1)^{(\alpha+\beta)\cdot x_1+\beta\cdot k\cdot f(w+y)}\end{aligned}$$

By definition of Φ_k, we saw in Eq. (4.7) that the preimages x_0 and x_1 of y are equal to y' and $y'+k$, where y' is the same as y with an additional bit set to zero injected at position $i(k)$. Thus, using the bilinearity of the scalar product,

$$\widehat{R_{k,w}}(\alpha,\beta) = \left(1+(-1)^{(\alpha+\beta)\cdot k}\right)\sum_{y}(-1)^{(\alpha+\beta)\cdot y'+\beta\cdot k\cdot f(w+y)}$$

and this is equal to zero, if $(\alpha+\beta)\cdot k=1$ or $\beta\cdot k=0$. In the other case, $(\alpha+\beta)\cdot k=0$ and $\beta\cdot k=1$, we have

$$\begin{aligned}\widehat{R_{k,w}}(\alpha,\beta) &= 2\cdot\sum_{y}(-1)^{(\alpha+\beta)\cdot y'+f(w+y)} = 2\cdot\sum_{x}(-1)^{(\alpha+\beta)\cdot(x'+w')+f(x)}\\ &= 2\cdot(-1)^{(\alpha+\beta)\cdot w'}\cdot\sum_{x}(-1)^{(\alpha+\beta)\cdot x'+f(x)}\\ &= 2\cdot(-1)^{(\alpha+\beta)\cdot w'}\cdot\widehat{f}(\Phi_k(\alpha+\beta)) \qquad (4.11)\end{aligned}$$

Finally, again because f is bent, we get

$$\widehat{R_{k,w}}(\alpha,\beta) = 2\cdot(-1)^{(\alpha+\beta)\cdot w'}\cdot(\pm 2^{\frac{n-1}{2}}) = \pm 2^{\frac{n+1}{2}}\,.$$

Note that the sign of the LAT entries is uniformly distributed and thus, when averaging over the w's, the non-diagonal entries cancel out.

▸ To more Rounds

When we look at more than one round, we try to approximate the linear hull by looking at the strongest linear trail. As already discussed in Lemma 95, for $r<n$ there are trails with probability one. We now show that any trail's correlation for $r\geqslant n$ rounds is actually upper bounded by $2^{-\frac{n-1}{2}}$:

Proposition 109. *For $r \geqslant n$ rounds, the absolute correlation of any non-trivial linear trail for* Bison *is upper bounded by* $2^{-\frac{n-1}{2}}$.

Proof. It is enough to show the above for any n-round trail. By contradiction, assume there exists a non-trivial trail $\theta = (\theta_0, \dots, \theta_n)$ with correlation one. Following Proposition 108, for every round i the intermediate mask θ_i has to fulfill $\theta_i \cdot k_i = 0$. Further $\theta_i = \theta_{i+1}$ for $0 \leqslant i < n$. Because all n round keys are linearly independent, this implies that $\theta_i = 0$, which contradicts our assumption. Thus, in at least one round the second or third case of Eq. (4.9) has to apply. ∎

It would be nice to be able to say more about the linear hull, analogously to the differential case. However, for the linear cryptanalysis this looks much harder, due to the denser LAT. In our opinion developing a framework where bounding linear hulls is similarly easy as it is for Bison with respect to differentials is a fruitful future research topic.

4.6.2 *Higher-Order Differentials and Algebraic Attacks.*

High-order differential attacks, cube attacks, algebraic attacks and integral attacks all make use of non-random behaviour of the ANF of parts of the encryption function. In all these attacks the algebraic degree of (parts of) the encryption function is of particular interest. In this section, we argue that those attacks do not pose a threat to Bison.

We next elaborate in more detail on the algebraic degree of the WSN construction. In particular, we are going to show that the algebraic degree increases at most linearly with the number of rounds. More precisely, if the round function is of degree d, the algebraic degree after r rounds is upper bounded by $r(d-1)+1$.

Actually, we are going to consider a slight generalization of the WSN construction and prove the above statement for this generalization.

► General Setting

Consider an initial state of n bits given as $x = (x_0, \dots, x_{n-1})$ and a sequence of Boolean functions

$$f_i : \mathbb{F}_2^{n+i} \to \mathbb{F}_2$$

for $0 \leqslant i < r$. We define a sequence of values y_i by setting $y_0 = f_0(x)$ and

$$y_i = f_i(x_0, \dots, x_{n-1}, y_0, \dots, y_{i-1}),$$

for $1 \leqslant i < r$. Independently of the exact choice of f_i the degree of any y_ℓ, as a function of x can be upper bounded as stated in the next proposition.

Proposition 110. *Let f_i be a sequence of functions as defined above, such that* $\deg(f_i) \leqslant d$. *The degree of y_ℓ at step ℓ seen as a function of the bits of the initial state $x_0, \dots, x_{n-1}$ satisfies*

$$\deg(y_\ell) \leqslant (d-1)(\ell+1)+1.$$

Moreover, for any $I \subseteq \{0, \dots, \ell\}$,

$$\deg(\prod_{i \in I} y_i) \leqslant (d-1)(\ell+1)+|I|.$$

Proof. The first assertion is of course a special case of the second one, but we add it for the sake of clarity. We prove the second, more general, statement by induction on ℓ.

Starting with $\ell = 0$, we have to prove that $\deg(y_0) \leqslant d$, which is obvious, as

$$y_0 = f_0(x_0, \dots, x_{n-1})$$

and $\deg(f_0) \leq d$.

Now, we consider some $I \subseteq \{0, \dots, \ell\}$ and show that

$$\deg(\prod_{i \in I} y_i) \leqslant (d-1)(\ell+1) + |I| \, .$$

We assume that $\ell \in I$, otherwise the result directly follows the induction hypothesis.

Since f_ℓ depends both on $y_0, \dots, y_{\ell-1}$ and x, we decompose it as follows:

$$y_\ell = f_\ell(y_0, \dots, y_{\ell-1}, x) = \sum_{\substack{J \subseteq \{0,\dots,\ell-1\} \\ 0 \leqslant |J| \leqslant \min(d,\ell)}} \left(\prod_{j \in J} y_j \right) g_J(x)$$

with $\deg(g_J) \leqslant d - |J|$ for all J since $\deg(f_\ell) \leqslant d$.

Then, for $I = \{\ell\} \cup I'$, we look at

$$y_\ell \left(\prod_{i \in I'} y_i \right) = \sum_{\substack{J \subseteq \{0,\dots,\ell-1\} \\ 0 \leqslant |J| \leqslant \min(d,\ell)}} \left(\prod_{j \in J \cup I'} y_j \right) g_J(x) \, .$$

From the induction hypothesis, the term of index J in the sum has degree at most

$$\begin{aligned} (d-1)\ell + |J \cup I'| + \deg(g_J) &= (d-1)\ell + \left|J \cup I'\right| + d - |J| \\ &\leqslant (d-1)(\ell+1) + \left|J \cup I'\right| - |J| + 1 \\ &\leqslant (d-1)(\ell+1) + |J| + \left|I'\right| - |J| + 1 \\ &\leqslant (d-1)(\ell+1) + |I| \end{aligned}$$

concluding the proof.

Special case of Bison

In the case of Bison, we make use of quadratic functions, and thus Proposition 110 implies that after r rounds the degree is upper bounded by $r+1$ only. Thus, it will take at least $n-2$ rounds before the degree reaches the maximal possible degree of $n-1$. Moreover, due to the construction of WSN, if all component functions of $E^r_{k,w}$ are of degree at most d, there will be at least one component function of $E^{r+n-1}_{k,w}$ of degree at most d. That is, there exist a vector $\beta \in \mathbb{F}_2^n$ such that

$$\beta \cdot E^{r+n-1}_{k,w}(x)$$

has degree at most d. Namely, for

$$\beta \in \operatorname{Span}\{k_r, \dots, k_{r+s}\}^\perp$$

it holds that

$$\deg\left(\beta \cdot E_{k,w}^{r+s}(x)\right) = \deg\left(\beta \cdot E_{k,w}^{r}(x) + \sum_{i=r}^{r+s} \lambda_i(\beta \cdot k_i)\right)$$
$$= \deg\left(\beta \cdot E_{k,w}^{r}(x)\right) .$$

We conclude there exists a component function of $E_{k,w}^{r+s}$ of non-maximal degree, as long as $0 \leqslant r \leqslant n-2$ and $0 \leqslant s \leqslant n-1$. Thus for BISON there will be at least one component function of degree less than $n-1$ for any number of rounds $0 \leqslant r \leqslant 2n-3$. However, similarly to the case of zero-correlation properties as described in Lemma 96, the vector β is key dependent and thus this property does not directly lead to an attack.

Finally, so far we only discussed upper bounds on the degree, while for arguing security, lower bounds on the degree are more relevant. As it seems very hard (just like for any cipher) to prove such lower bounds, we investigated experimentally how the degree increases in concrete cases. As can be seen in Figure 4.3 the maximum degree is reached for almost any instance for $n+5$ rounds. Most importantly, the fraction of instances where it takes more than $n+2$ rounds decreases with increasing block length n. Moreover, the round function in BISON experimentally behaves with this respect as a random function, as can be seen on Figure 4.4. Thus, as the number of rounds is $3n$, we are confident that attacks exploiting the algebraic degree do not pose a threat for BISON.

Besides the WSN construction, a special case of the above proposition worth mentioning is a non linear feedback generator (NLFSR).

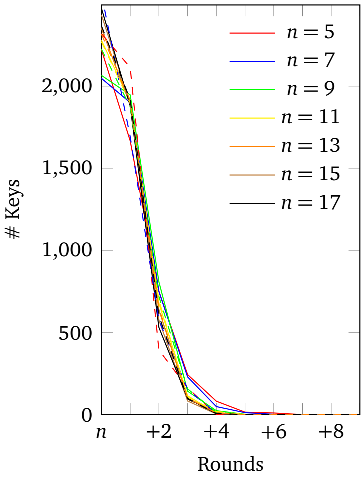


FIGURE 4.3: Number of rounds more than n needed to achieve full degree. Solid lines for random round keys, dashed lines for round keys derived from BISON's key schedule.

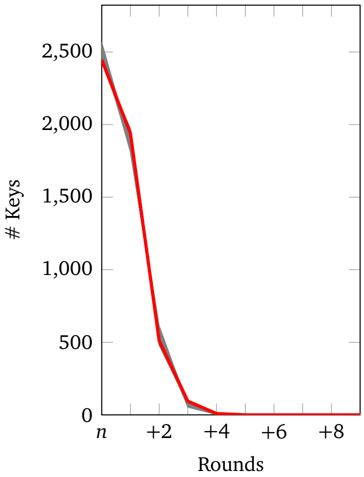


FIGURE 4.4: Behaviour of BISON's f function (red thick solid) versus random f (gray solid) with algebraic degree 2 for $n = 17$.

- DEGREE OF NLFSR

One well-known special case of the above general setting is an NLFSR or, equivalently a maximally unbalanced Feistel cipher, depicted in the margin, see Figure 4.5. Proposition 110 implies that the degree of any NLFSR increases linearly with the number of rounds. To the best of our knowledge, this is the first time this have been observed in this generality. We like to add that this is in sharp contrast to how the degree increases for SPN ciphers. For SPN ciphers the degree usually increases exponentially until a certain threshold is reached [Bou+11].

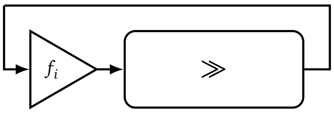


FIGURE 4.5: Non-Linear Feedback Shift Register

[Bou+11] Boura et al., "Higher-order differential properties of Keccak and Luffa"

4.6.3 *Other attacks*

We briefly discuss other cryptanalytic attacks.

- IMPOSSIBLE DIFFERENTIALS

In Lemma 102 and Corollary 103, we discuss that every output difference is possible after more than n rounds. Consequently, there are no impossible differentials for BISON.

- TRUNCATED DIFFERENTIALS

Due to our strong bounds on differentials it seems very unlikely that any strong truncated differential exists.

- Zero Correlation Linear Cryptanalysis

In Lemma 96 we already discussed generic zero correlation linear hulls for the WSN construction. Depending on the actual key used, this technique may be used to construct a one-round-longer zero-correlation trail. For this, we need two *distinct* elements $\alpha \in \operatorname{Span}\{k_1, \dots, k_{n-1}\}^\perp$, $\beta \in \operatorname{Span}\{k_n, \dots, k_{2n-2}\}^\perp$, and construct the trail analogously to Lemma 96 (which may not exist, due to the key dependency).

- Invariant Attacks

For an invariant attack, we need a Boolean function g, such that $g(x) + g(E^r_{k,w}(x))$ is constant *for all* x and some *weak keys* (k, w). As the encryption of any message is basically this message with some of the round keys added, key addition is the only operation which is performed. It has been shown in [Bei+17, Proposition 1] that any g which is invariant for a linear layer followed by the addition of the round key k_i as well as for the same up to addition of a different k_j, has a linear space containing $k_i + k_j$. In the case of the linear layer being the identity, the linear space actually contains also the k_i and k_j (by definition).

[Bei+17] Beierle et al., "Proving Resistance Against Invariant Attacks: How to Choose the Round Constants"

Thus, the linear space of any invariant for our construction has to contain $\operatorname{Span}\{k_1, \dots, k_{3n}\}$ which is obviously the full space $\mathbb{F}_2^n$. Following the results of [Bei+17], there are thus no invariant subspace or nonlinear invariant attacks on Bison.

- Related-Key Attacks

In generic related-key attacks, the attacker is also allowed to exploit encryptions under a related, that is $k' = f(k)$, key – in the following, we restrict our analysis to the case where f is the addition with a constant. That is, the attacker cannot only request $E_{k,w}(x)$, and $E_{k,w}(x + \alpha)$, but also $E_{k+\beta,w+\beta'}(x)$ or $E_{k+\beta,w+\beta'}(x + \alpha)$, for α (difference in the input x), β (difference in the key k) and β' (difference in the key w) of her choice. As $\beta = \beta' = 0$ would result in the standard differential scenario, we exclude it for the remainder of this discussion. Similar, $\beta = k$ results in $\Phi_{k+\beta} = \Phi_0$, which we did not define, thus we also skip this case and refer to the fact that if an attacker chooses $\beta = k$, she basically already has guessed the secret key correctly.

First note that, for any input difference (α, β, β'), the possible output differences after one round are

$$\begin{array}{ll} \alpha & \text{if } (u,v) = (0,0), \\ \alpha + \beta + k & \text{if } (u,v) = (0,1), \\ \alpha \qquad + k & \text{if } (u,v) = (1,0), \text{ and} \\ \alpha + \beta & \text{if } (u,v) = (1,1), \end{array}$$

where

$$u = f(w + \Phi_k(x)), \tag{4.12}$$
$$v = f(w + \beta' + \Phi_{k+\beta}(x + \alpha)). \tag{4.13}$$

Our aim is to bound both the probability that $u + v = 0$ and that $u + v = 1$ by $3/4$. This implies that the probability for any related-key differential characteristic for one round is at most $3/4$. Thus the probability for any r-round related-key differential characteristic is bounded by $(3/4)^r$. For this, we need the following lemma.

Lemma 111. *Let us consider the linear function Φ_k defined by Eq. (4.6). Given k and $\beta \notin \{0, k\}$. Then the dimension of the image of the linear function $x \mapsto \Phi_k(x) + \Phi_{k+\beta}(x)$ is either one or two.*

Proof. For the sake of simplicity, we instead consider $\Phi'_k(x)+\Phi'_{k+\beta}(x)$, where Φ' is the same as Φ but does not truncate its output. Basically the same argumentation then holds for Φ as well. This function can also be written as

$$\begin{aligned}\Phi'_k(x) + \Phi'_{k+\beta}(x) &= x + x_{i(k)}k + x + x_{i(k+\beta)}(k+\beta)\\ &= (x_{i(k)} + x_{i(k+\beta)})k + x_{i(k+\beta)}\beta.\end{aligned}$$

Thus

$$\Phi'_k(x) + \Phi'_{k+\beta}(x) \in \operatorname{Span}\{k, \beta\}$$

for all x, which upper bounds the dimension of the image by two. As e. g. $x = e_{i(k+\beta)}$ is not mapped to zero, the dimension of the image is at least one, completing the proof.

By the rank-nullity theorem (see Theorem 10), this implies that

$$\dim \operatorname{Ker} \Phi_k + \Phi_{k+\beta} \in \{n-1, n-2\}.$$

We can now show the following proposition.

Proposition 112. *For r rounds, the probability of any related-key differential characteristic for* Bison, *averaged over all whitening key sequences $(w_1, \ldots, w_r)$, is upper bounded by $\left(\frac{3}{4}\right)^r$.*

Proof. First, let us introduce the set $A_{x,k}^{\alpha,\beta,\beta'}$ defined as:

$$A_{x,k}^{\alpha,\beta,\beta'} := \left\{ w \in \mathbb{F}_2^{n-1} \mid f(w + \Phi_k(x)) + f(w + \beta' + \Phi_{k+\beta}(x+\alpha)) = 0 \right\},$$

that is the set of all round keys w, for which x, k, α, β, and β' result in $u + v = 0$ (where u and v are as at the beginning of this section). In case that

$$\Phi_k(x) = \beta' + \Phi_{k+\beta}(x+\alpha),$$

the size of $A_{x,k}^{\alpha,\beta,\beta'}$ is 2^{n-1}, while if the equality does not hold the set is of size 2^{n-2} since f is bent.

For k, α, β and β' fixed, the number of x such that the size of $A_{x,k}^{\alpha,\beta,\beta'}$ is 2^{n-1}, is just the size of the preimage of $\Phi_{k+\beta}(\alpha) + \beta'$ under the linear mapping $x \mapsto \Phi_k(x) + \Phi_{k+\beta}(x)$. The size of this preimage is either 0 or $\left|\operatorname{Ker} \Phi_k + \Phi_{k+\beta}\right|$. Denote by

$$B = \left|\operatorname{Ker} \Phi_k + \Phi_{k+\beta}\right|,$$

which, by Lemma 111, is bounded by 2^{n-1}.

Then, the probability over x and w for having an output difference of α or $\alpha+\beta$ is:

$$\begin{aligned}
&\Pr_{x,w}[u+v=0 \text{ for fixed } k,\ \alpha,\ \beta,\ \beta']\\
&\leqslant \frac{B}{2^n}+\frac{2^{n-2}}{2^{n-1}}\cdot\frac{2^n-B}{2^n} \leqslant \frac{B}{2^n}+\frac{1}{2}\left(1-\frac{B}{2^n}\right) \leqslant \frac{1}{2}+\frac{B}{2^{n+1}}\\
&\leqslant \frac{1}{2}+\frac{1}{4}=\frac{3}{4}.
\end{aligned}$$

The other case, $u+v=1$, follows with the same argument. Thus, the probability for each of the four possible cases $(u,v)\in\mathbb{F}_2\times\mathbb{F}_2$ can be upper bounded by $(3/4)$, which concludes the proof.

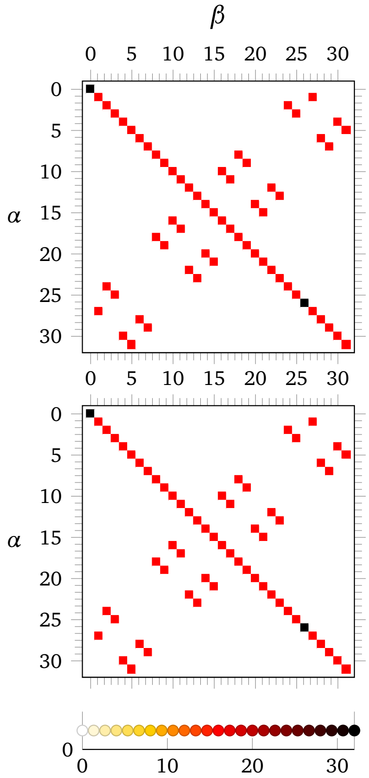


FIGURE 4.6: BISON's DDT for $n=5$, $k=13$ averaged over w (upper) resp. fixed $w=k$ (lower).

▸ FURTHER OBSERVATIONS

During the design process, we observed the following interesting point: For sparse master keys k and w and message m, e. g. $k=w=m=1$, in the first few rounds, nothing happens. This is mainly due to the choice of sparse key schedule polynomials p_w and p_k and the fact that f_0 outputs 0 if only one bit in its input is set (as $0\cdot x=0$ for any x).

To the best of our knowledge, this observation cannot be exploited in an attack.

▸ EXPERIMENTAL RESULTS

We conducted experiments on small-scale versions of BISON with $n=5$. The DDTs and LATs, depicted using the "Jackson Pollock representation" [BP15], for one (see also Figure 4.6 in the margin) to ten rounds are listed in [Can+18, Appendix B]. In [Can+18, Appendix B] one can see that the two cases of averaging over all possible w_i and choosing a fixed w_i result in very similar differential behaviours. Additionally, after $n=5$ rounds, the plots do not change much.

[BP15] Biryukov and Perrin, "On Reverse-Engineering S-Boxes with Hidden Design Criteria or Structure"

[Can+18] Canteaut et al., BISON – *Instantiating the Whitened Swap-Or-Not Construction (Full Version)*

The results in the linear case, see also [Can+18, Appendix B], are quite similar. The major difference here, is the comparable bigger entries for a fixed w_i. Nonetheless, most important is that there are no high entries in the average LAT which would imply a strong linear approximation for many keys. Additionally one also expects for a random permutation not too small LAT entries. Note that one can well observe the probability-one approximation for $4=n-1$ rounds (lower right corner of the corresponding plot).

4.7 IMPLEMENTATION ASPECTS OF BISON

As the round function is involutory, we do not need to implement a separate decryption, but instead can just use the encryption implementation with reversed round keys.

To implement the two LFSRs for the key schedule, we need two primitive polynomials of degree n and $n-1$. Clocking an LFSR with feedback polynomial $p(x)$ corresponds to multiplying the state by $x\in\mathbb{F}_2[x]/p(x)$. This can be implemented by a simple left shift and a conditional addition of the polynomial, if a modulo reduction is necessary. To keep this addition as efficient as possible it is advantageous to have all non-leading monomials of the polynomial of degree less than the word size of the implementation's underlying CPU, since in this case, we only need to add a term to the least significant word of the state. Appropriate polynomials can easily be found by enumerating possible candidates and test if the candidate is primitive. See [Can+18, Appendix A] for possible, good-to-implement, choices for $p_w(x)$ and $p_k(x)$.

For comparison and test reasons we also provide test vectors in [Can+18, Appendix C] and a SAGEMATH implementation, see [Can+18, Appendix E]. We implemented the 129-bit instance in C on a 64-bit INTEL CORE I7-8700 CPU (COFFEE LAKE architecture) running at 3.7 GHz. The corresponding source code can be found in [Can+18, Appendix G].

Utilising the CPU's `popcount` instruction, this implementation consumes a size of 701 bytes when compiled with `-Os`. The same implementation needs a bit more then 3 000 cycles per byte for the encryption of one 129-bit block.[5] Table 4.1 summarises these results. While this might be obvious, we nevertheless want to note that it is important for reliable benchmarks to turn off advanced performance capabilities of modern CPUs.

Regarding cycles per byte, this is six orders of magnitude slower than optimised implementations of AES. Even if the reference implementation is not optimised, we do not believe to come close to a competitive speed. Another point which can be seen in Table 4.1 is the deviating runtime of our implementation. The reference implementation is clearly not constant time. For any secure implementation this and other side channels have of course to be taken into account. Nevertheless, a side channel-resistant implementation is out of scope of this work but is certainly an interesting research direction. We expect the simplicity of our design to support side-channel countermeasures.

TABLE 4.1: Performance of our 129-bit BISON implementation on an INTEL CORE I7-8700 CPU, running at 3.7 GHz. Cycles per byte are measured for 1 000 000 runs, HYPER-THREADING and TURBO-BOOST were disabled.

Block size	129	(bit)
Code size	701	(byte)
	Cycles per byte	
median	mean	σ
25 124	25 044.51	1 392.39

While BISON results in a, from the cryptanalysis perspective, elegant design, the restriction on odd block length is from an implementation perspective unpleasant. Nevertheless, our analysis for BISON-like ciphers is generic enough to easily apply it to other BISON-like instances. Thus we turn our attention now to an instance with even block length: WISENT.

4.8 SPECIFICATION OF WISENT

In order to exploit our analysis of BISON-like instances, especially Lemma 104, in a helpful way, we need to find f_0, f_1 with a low absolute indicator. Then we can use these to give good bounds on the EDP. Due to the Φ_k function, the f_i have to operate on $n-1$ bits, with $n-1$ odd and thus we cannot use bent functions.

Recall from the previous chapter that finding Boolean functions with good absolute indicator is an interesting problem. It is actually not even clear what a meaningful lower bound on the absolute indicator is when bent functions are not considered. Additionally, the currently known constructions (for $n-1$ even) that reach very low autocorrelations are highly non trivial and thus quite complex to implement.

Thus, and because as long as we do not know a lower bound for n odd and a construction reaching this lower bound, we cannot give an optimal instance for WISENT anyway, we take a different approach. Instead, to get a

[5] For comparison: exploiting `AES-NI` instructions on modern CPUs results in 4.375 cycles per byte for encrypting one AES-128 block, excluding the key-schedule. When parallelism can be exploited, the speed can be even further increased, eventually tending to less than one cycle per byte. Implementing a full AES-128 encryption with `AES-NI` instructions including the key schedule uses 394 bytes.

family of Boolean functions f_i that we can use for WISENT with any even block length, we use the direct sum construction.

That is, our f_0, f_1 are of the form $f_i(x,y) = g_i(x) + h(y)$ with $g_i : \mathbb{F}_2^5 \to \mathbb{F}_2$ and $h : \mathbb{F}_2^{n-6} \to \mathbb{F}_2$.[6] For the even length part h of f, we utilise the same bent function as for BISON:

$$\begin{aligned} h &: \mathbb{F}_2^{n-6} \to \mathbb{F}_2 \\ h(x) &:= x[1:(n-6)/2] \cdot x[(n-6)/2+1:n-6]\,. \end{aligned}$$

For the odd length part, we use $g_0 : \mathbb{F}_2^5 \to \mathbb{F}_2$ with

$$\begin{aligned} g_0(x_1,\dots,x_5) := {} & x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_1x_2x_4x_5 + x_1x_3x_4x_5 + x_2x_3x_4x_5 \\ & + x_1x_2x_5 + x_1x_3x_4 + x_1x_4x_5 + x_2x_3x_4 + x_2x_3x_5 \\ & + x_1x_4 + x_1x_5 + x_2x_3 + x_2x_5 + x_3x_4 \\ & + x_1 + x_2 + x_3 + x_4 + x_5\,, \end{aligned}$$

or as truth table in hexadecimal form 00071356, and we choose g_1 as the complement of g_0.

The n-bit instance of WISENT iterates $3n$ rounds of the above specified round function, using the same key schedule as BISON.

WISENT's Round Function

Let $n \geqslant 6$ be even. For round keys $k_i \in \mathbb{F}_2^n$ and $w_i \in \mathbb{F}_2^{n-1}$ the round function computes

$$R_{k_i,w_i}(x) := x + f_{b(i)}\big(w_i + \Phi_{k_i}(x)\big)k_i. \tag{4.14}$$

where

- Φ_{k_i} is defined as in Eq. (4.6),
- $f_{b(i)}$ is defined as

$$\begin{aligned} f_{b(i)} &: \mathbb{F}_2^{n-1} \to \mathbb{F}_2 \\ f_{b(i)}(x_1,\dots,x_{n-1}) &:= g_{b(i)}(x_1,\dots,x_5) + h(x_6,\dots,x_{n-1}) \end{aligned}$$

- g_0 is defined by its truth table in hexadecimal form: 00071356, and g_1 as $g_0 + 1$,
- h is defined as

$$\begin{aligned} h &: \mathbb{F}_2^{n-6} \to \mathbb{F}_2 \\ h(x_1,\dots,x_{n-6}) &:= x[1:n/2-3] \cdot x[n/2-2:n-6]\,. \end{aligned}$$

- and $b(i)$ is 0 if $i \leqslant \frac{r}{2}$ and 1 otherwise.

Security Claim 2 (WISENT). *We claim n-bit security for* WISENT *in the single-key model. We emphasise that we do not claim any security in the related-key, chosen-key or known-key model.*

Several of the cryptanalysis results of BISON directly translate to WISENT, as we discuss in the following section.

[6] The reason why we choose this splitting of $n-1 = 5+n-6$ bits is that it is still feasible for 5 bits to search for functions with very good absolute indicator.

4.8.1 *Security Analysis of* WISENT

We first use Lemma 104 for a bound regarding differential cryptanalysis and then discuss other attacks. The only difference for these other attacks occur for linear cryptanalysis, which we thus discuss at the end of this section.

- DIFFERENTIAL CRYPTANALYSIS

For Lemma 104 we need to bound the absolute indicator of f_i. We can adapt Lemma 64 to autocorrelation properties.

Lemma 113 (Autocorrelation coefficients of direct sums)**.** *Let $f : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2$ be of the form $f(x,y) = g(x) + h(y)$. Then for f's autocorrelation coefficients and absolute indicator it holds that*

$$\mathcal{A}_f(\alpha,\beta) = \mathcal{A}_g(\alpha)\cdot\mathcal{A}_h(\beta)\,, \quad \textit{and} \quad \mathcal{M}(f) = \max\{2^n \cdot \mathcal{M}(h), 2^m \cdot \mathcal{M}(g)\}\,.$$

Proof. The first part follows analogously to the proof of Lemma 64.

However, for the second part note that, for any f, $\mathcal{A}_f(0) \geqslant \mathcal{M}(f)$. Thus for the absolute indicator of a direct sum f the critical autocorrelation coefficients are those at point $(0,\beta),(\alpha,0) \neq (0,0)$.

Applying Lemma 113 to the f_i's, we get

$$\mathcal{M}(f_i) = 2^{n-6}\cdot\mathcal{M}(g_i) = 2^{n-6}\cdot 2^3 = 2^{n-3}\,.$$

From Lemma 104 the EDP of WISENT with $r = n$ rounds is bounded by

$$\mathrm{EDP}(\alpha,\beta) \leqslant \left(\frac{1}{2} + 2^{-n} \max_{1\leqslant i\leqslant n} \mathcal{M}\left(f_{b(i)}\right)\right)^{n-1}$$

which in the case of WISENT is

$$= \left(\frac{1}{2} + 2^{-n}\cdot 2^{n-3}\right)^{n-1} = \left(\frac{1}{2} + \frac{1}{8}\right)^{n-1}\,.$$

Experimental results show the expected behaviour for WISENT's DDT, see Figure 4.7.

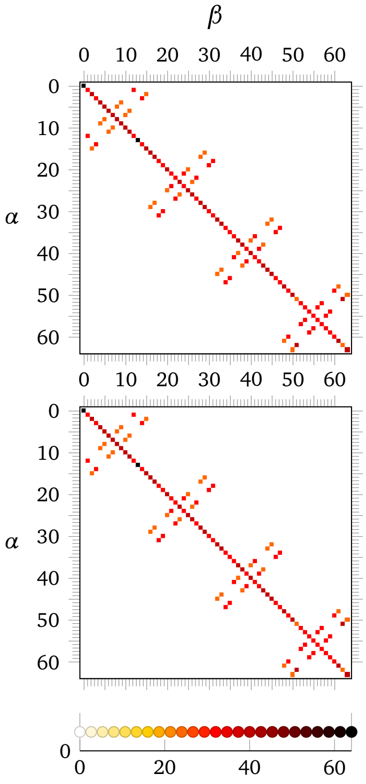


FIGURE 4.7: WISENT's DDT for $n = 6$, $k = 13$ averaged over w (upper) resp. fixed $w = k$ (lower).

- HIGHER-ORDER DIFFERENTIALS AND ALGEBRAIC ATTACKS

Our analysis of the algebraic degree of BISON does not exploit any details of the Boolean functions f_i, but is independent of the exact choice for them. Thus, our results also apply to WISENT, which is why we also choose $3n$ rounds for this instance.

- IMPOSSIBLE DIFFERENTIALS AND ZERO CORRELATION ATTACKS

Analogously, the generic analysis of the WSN construction and BISON-like instances show that there are no impossible differentials for more than n rounds, and no zero correlation attacks for $2n$ or more rounds.

- INVARIANT ATTACKS

Here, too, our argument does not depend on the exact choice of the Boolean functions f_i, and thus the argument by Beierle et al. [Bei+17] holds again.

▸ LINEAR CRYPTANALYSIS

While we can follow the same strategy as for BISON (bounding the Fourier coefficient of a single trail by first computing the one round coefficients and then using Proposition 109 for more rounds), we need to slightly adjust the proof for the new Boolean functions.

Proposition 114. *Let* $R_{k,w} : \mathbb{F}_2^n \to \mathbb{F}_2^n$ *be* WISENT*'s round function as defined in* Eq. (4.14)*. Then its LAT entries are bounded by*

$$\widehat{R_{k,w}}(\alpha,\beta) \leqslant \begin{cases} 2^n & \text{if } \alpha=\beta \text{ and } \beta\cdot k=0 \\ \pm 2^{\frac{n}{2}+2} & \begin{array}{l}\text{if } \alpha=\beta \text{ and } \beta\cdot k=1 \text{ or} \\ \text{if } \alpha\neq\beta,\ (\alpha+\beta)\cdot k=0, \text{ and } \beta\cdot k=1\end{array} \\ 0 & \text{otherwise} \end{cases} .$$

Proof. The proof is analogous to Proposition 108, with two small changes in Eqs. (4.10) and (4.11).

First, for the case $\alpha=\beta$ and $\beta\cdot k=1$, we have

$$\widehat{R_{k,w}}(\alpha,\alpha) = 2\cdot \sum_{y'\in\mathbb{F}_2^{n-1}} (-1)^{f(y')} = 2\cdot\widehat{f}(0)$$

due to Lemma 64 this is

$$= 2\cdot \pm 2^{\frac{n-6}{2}} \cdot \widehat{g}(0) = \pm 2^{\frac{n}{2}-2}\cdot 12 < \pm 2^{\frac{n}{2}+2} .$$

Second, for the case $\alpha\neq\beta$, $(\alpha+\beta)\cdot k=0$, and $\beta\cdot k=1$, it is

$$\widehat{R_{k,w}}(\alpha,\beta) = 2\cdot(-1)^{(\alpha+\beta)\cdot w'}\cdot\widehat{f}(\Phi_k(\alpha+\beta))$$

which can be bounded, with $\gamma=\Phi_k(\alpha+\beta)[1:5]$, by

$$\begin{aligned}\widehat{R_{k,w}}(\alpha,\beta) &= 2\cdot(-1)^{(\alpha+\beta)\cdot w'}\cdot\widehat{f}(\Phi_k(\alpha+\beta)) \\ &= \pm 2^{\frac{n}{2}-2}\cdot\widehat{g}(\gamma) \leqslant \pm 2^{\frac{n}{2}-2}\cdot 12 < \pm 2^{\frac{n}{2}+2} .\end{aligned}$$

The other cases are as in Proposition 108.

To use this for an upper bound for the absolute correlation of any non-linear trail over more rounds, we can simply adjust Proposition 109 with the new bound for one round, resulting in $2^{-\frac{n}{2}+2}$ as the maximal absolute correlation.

TABLE 4.2: Performance of our 128-bit WISENT implementation on an INTEL CORE I7-8700 CPU, running at 3.7 GHz. Cycles per byte are measured for 1 000 000 runs, HYPER-THREADING and TURBO-BOOST were disabled.

Block size	128	(bit)
Code size	683	(byte)
Cycles per byte		
median	mean	σ
23 920	24 392.32	1 482.97

4.8.2 *Implementation Aspects of* WISENT

Basically the same points as for BISON's implementation also apply to WISENT. Its round function is involutory, the key schedule is the same as for BISON (thus, the same polynomials can be used), and the same high number of rounds lead to a very slow implementation, see also Table 4.2. Again, for comparison and test reasons, we provide test vectors, see [Can+18, Appendix D]. We implemented WISENT in SAGEMATH ([Can+18, Appendix F]) and C ([Can+18, Appendix H]).

[Can+18] Canteaut et al., BISON – *Instantiating the Whitened Swap-Or-Not Construction (Full Version)*

4.9 CONCLUSION

Efficiency of symmetric ciphers have been significantly improved further and further, in particular within the trend of lightweight cryptography. However, when it comes to arguing about the security of ciphers, the progress is rather limited and the arguments basically did not get easier nor stronger since the development of the AES. In our opinion it might be worth shifting the focus to improving security arguments for new designs rather than (incrementally) improving efficiency. We see BISON and WISENT as a first step in this direction.

With our instance for the WSN construction and its strong resistance to differential cryptanalysis, this framework emerges as an interesting possibility to design block ciphers. Unfortunately, we are not able to give better than normal arguments for the resistance to linear cryptanalysis. It is thus an interesting question, if one can find a similar instance of the WSN construction for which comparable strong arguments for the later type of cryptanalysis exist.

Alternative designs might also be worth looking at. For example many constructions for bent functions are known and could thus be examined as alternatives for the scalar product used in BISON. One might also look for a less algebraic design – but we do not yet see how this would improve or ease the analysis or implementation of an instance.

Especially for WISENT, alternatives to the used Boolean function might be an interesting future research direction. Our construction is far from optimal with regard to its absolute indicator – mainly due to the fact that neither an optimal function nor a (tight) lower bound on the absolute indicator is known today. Thus, finding either a lower bound or – as the known constructions or examples with good absolute indicator, see e. g. [Kav+07; TM18], are quite complex from an implementation point of view – finding simpler constructions would be a valuable result.

Another line of future work is the in-depth analysis of implementation optimizations and side channel-resistance of BISON and WISENT.

Part III

AUTOMATED METHODS IN DESIGN AND ANALYSIS

5

Heuristics for XOR counts

"Premature optimization is the root of all evil"
—Donald Knuth

► SYNOPSIS This chapter discusses heuristics for XOR count optimisations and the application of these to previously published results. It is based on the article

> Thorsten Kranz, Gregor Leander, Ko Stoffelen, and Friedrich Wiemer. "Shorter Linear Straight-Line Programs for MDS Matrices". In: *IACR Trans. Symm. Cryptol.* 2017.4 (2017), pp. 188–211. ISSN: 2519-173X. DOI: 10.13154/tosc.v2017.i4.188-211. IACR PREPRINT: 2017/1151.

All authors contributed equally.

We start by recalling the notion of an MDS matrix and several special types of matrix constructions. Then we examine the different XOR count metrics discussed in the literature and discuss heuristic approaches published in another line of research. Finally we report on improvements on previously published results from the application of these heuristics. We conclude the chapter with a statistical evaluation of different matrix constructions and a summary of follow up work.

5.1 INTRODUCTION

In the context of lightweight cryptography, several researchers started to tackle Problem 7 (Optimise Building Block), with a special focus on the linear layers more recently and even more specifically the implementation of MDS matrices. That is, linear layers with an optimal branch number.

The first line of work focused solely on minimising the chip area of the implementation. This started with the block cipher PRESENT [Bog+07] and goes over to many more designs, such as LED [Guo+11a] and the hash function PHOTON [Guo+11b], where in the latter MDS matrices were constructed that are especially optimised for chip area by allowing a serialised implementation. However, there seem to be only a few practical applications where a small chip area is the only optimization goal and for those applications very good solutions are already available by now.

[Bog+07] Bogdanov et al., "PRESENT: An Ultra-Lightweight Block Cipher"

[Guo+11a] Guo et al., "The LED Block Cipher"

[Guo+11b] Guo et al., "The PHOTON Family of Lightweight Hash Functions"

Later, starting with [Kho+14], researchers focused on round-based implementations with the goal of finding MDS constructions that minimise the number of XOR operations needed for their implementation. Initially, the

[Kho+14] Khoo et al., "FOAM: Searching for Hardware-Optimal SPN Structures and Components with a Fair Comparison" number of XOR operations needed was bounded by the number of ones in the binary representation of the matrix.

However, as the number of ones only gives an upper bound on the number of required XORs, several papers started to deviate from this conceptually easier but less accurate definition of XOR count and started to consider more efficient ways of implementing MDS matrices. Considering an $n \times n$ MDS matrix over a finite field $\mathbb{F}_{2^k}$ given as $M = (m_{i,j})$ the aim was to choose the elements $m_{i,j}$ in such a way that implementing all of the multiplications $x \mapsto m_{i,j}x$ in parallel becomes as cheap as possible. In order to compute the matrix M entirely, those partial results have to be added together, for which an additional amount of XORs is required. It became common to denote the former cost as the overhead and the later cost, i. e. the cost of combining the partial results as a fixed, incompressible part. A whole series of papers [Sim+15; Bei+16a; LS16; LW16; SS16a; SS16b; Jea+17b; LW17; SS17; Zho+17] managed to reduce the overhead.

From a different viewpoint, what happened was that parts of the matrix, namely the cost of multiplication with the $m_{i,j}$, were extensively optimised, while taking the overall part of combining the parts as a given. That is, researchers have focused on local optimization instead of global optimization.

Indeed the task of globally optimising is far from trivial, and thus the local optimization is a good step forward.

Interestingly, the task to optimise the cost of implementing the multiplication with a relatively large, e. g., 32×32 binary matrix, is another extensively studied line of research. It is known that the problem is NP-hard [Boy+13; Boy+08] and thus renders quickly infeasible for increasing matrix dimension. However, quite a number of heuristic algorithms for finding the shortest linear straight-line program, which exactly corresponds to minimising the number of XORs, have been proposed in the literature [Paa97; Boy+08; BP10; FSK10; FS10; Boy+13; Boy+17; Vis+17]. Those algorithms produce very competitive results with a rather reasonable running time for arbitrary binary matrices of dimension up to at least 32.

[Boy+13] Boyar et al., "Logic Minimization Techniques with Applications to Cryptology"

[Boy+08] Boyar et al., "On the Shortest Linear Straight-Line Program for Computing Linear Forms"

Thus, the natural next step in order to optimise the cost of implementing MDS matrices is to combine those two approaches. This is exactly what we are doing in our work.

▸ Our contribution

By applying the heuristic algorithms to find a short linear straight-line program to the case of MDS matrices, we achieve the following.

First, we use several well-locally-optimised MDS matrices from the literature and apply the known algorithms to all of them. This is conceptually easy, with the main problem being the implementation of those algorithms. In order to simplify this for follow-up works, we make our implementation publicly available.

This simple application leads immediately to significant improvements. For instance, we get an implementation of the AES MixColumn matrix that outperforms all implementations in the literature, i. e. we use 97 XORs while the best implementation before used 103 XORs ([Jea+17a]). In the case of applying it to the other constructions, we often get an implementation using *less XOR operations than what was considered fixed costs before*. That is, when

[Jea+17a] Jean et al., "Bit-Sliding: A Generic Technique for Bit-Serial Implementations of SPN-based Primitives - Applications to AES, PRESENT and SKINNY" (artificially) computing it, the overhead would actually be negative. This confirms our intuition that the overhead was already very well optimised in previous work, such that now optimising globally is much more meaningful.

Second, we took a closer look at how the previous constructions compare when being globally optimised. Interestingly, the previous best construction, i. e. the MDS matrix with smallest overhead, was most of the time *not the one with the fewest XORs*. Thus, with respect to the global optimum, the natural question was, which known construction actually performs best. In order to analyze that, we did extensive experimental computations to compare the distribution of the optimised implementation cost for the various constructions. The, somewhat disappointing, result is that all known constructions behave basically the same. The one remarkable exception is the subfield construction for MDS matrices, first introduced in WHIRLWIND [Bar+10].

[Bar+10] Barreto et al., "Whirlwind: a new cryptographic hash function"

Third, we looked at finding matrices that perform exceptionally well with respect to the global optimization, i. e. which can be implemented with an exceptional low *total* number of XORs. Those results are summarised in Table 5.2. Compared to previous known matrices, ours improve on all – with the exception of one, where the best known matrix is the already published matrix from [SS16b].

[SS16b] Sarkar and Syed, "Lightweight Diffusion Layer: Importance of Toeplitz Matrices"

Finally, we like to point out two restrictions of our approach. First, we do not try to minimise the amount of temporary registers needed for the implementation. Second, in line with all previous constructions, we do not minimise the circuit depth. The later restriction is out of scope of the current work but certainly an interesting task for the future.

5.2 MDS MATRICES AND MATRIX CONSTRUCTIONS

MDS matrices are related to a Maximum Distance Separable code, hence their name, and play an important role in the design of block ciphers. From the diffusion point of view, such matrices are optimal and thus a natural choice for linear layers. We recall now if and when a matrix is MDS and what kind of constructions for MDS matrices there are. Additionally we discuss other popular constructions for matrices.

5.2.1 *MDS matrices*

Following the importance of MDS matrices for the diffusion in block cipher constructions, we define an MDS matrix as follows.

Definition 115. An $n \times n$ matrix M is *Maximum Distance Separable (MDS)* if and only if $\mathrm{bn}(M) = n + 1$.

It has been shown that a matrix is MDS if and only if all its square submatrices are invertible [MS77, page 321, Theorem 8]. MDS matrices do not exist for every choice of n, k. The exact parameters for which MDS matrices do or do not exist are investigated in the context of the famous MDS conjecture which was initiated in [Seg55].

[MS77] MacWilliams and Sloane, *The theory of Error-Correcting Codes*

[Seg55] Segre, "Curve razionali normali ek-archi negli spazi finiti"

For binary matrices, we need to modify Definition 115.

Definition 116. A binary matrix $B \in \mathbb{F}_2^{nk \times nk}$ is *MDS* (for k-bit words) if and only if $\mathrm{bn}_k(M) = n + 1$.

MDS matrices have a common application in linear layers of block ciphers, due to the wide trail strategy proposed for the AES, see [Dae95; DR02]. We typically deal with $n \times n$ MDS matrices over $\mathbb{F}_2^k$ respectively binary $\mathbb{F}_2^{nk \times nk}$ matrices that are MDS for k-bit words where $k \in \{4, 8\}$ is the size of the S-box. In either case, when we call a matrix MDS, the size of k will always be clear from the context when not explicitly mentioned.

[Dae95] Daemen, "Cipher and Hash Function Design Strategies based on linear and differential cryptanalysis"

[DR02] Daemen and Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*

It is easy to see that, if $M \in \mathbb{F}_{2^k}^{n \times n}$ is MDS, then also $\mathrm{BinMatr}(M)$ is MDS for k-bit words. On the other hand, there might also exist binary MDS matrices for k-bit words that do not have an according representation over $\mathbb{F}_2^k$.

Other, non-MDS, matrices are also common in cipher designs. To name only a few examples: PRESENT's permutation matrix [Bog+07], lightweight implementable matrices from PRINCE [Bor+12], or PRIDE [Alb+14], or recently, e. g. in MIDORI [Ban+15], or QARMA [Ava17], almost-MDS matrices.

5.2.2 *MDS Constructions*

Cauchy and Vandermonde matrices are two famous constructions for building MDS matrices. They have the advantage of being provably MDS.

However, as known from the MDS conjecture, for some parameter choices, MDS matrices are unlikely to exist. For example, we do not know how to construct MDS matrices over $\mathbb{F}_{2^2}$ of dimension 4×4.

Definition 117 (Cauchy matrix)**.** Given two disjoint sets of n elements of a field $\mathbb{F}_{2^k}$, $A = \{a_1, \ldots, a_n\}$, and $B = \{b_1, \ldots, b_n\}$. Then the matrix $M = \mathrm{cauchy}(A, B)$ consists of the entries $m_{i,j}$ with

$$m_{i,j} := \frac{1}{a_i - b_j}, \qquad \text{with } 1 \leq i, j \leq n$$

and is called a *Cauchy* matrix.

Every Cauchy matrix is MDS, e. g. see [GR13, Lemma 1].

[GR13] Gupta and Ray, "On Constructions of Involutory MDS Matrices"

Definition 118 (Vandermonde matrix)**.** Given an n-tuple $(a_1, \ldots, a_n)$ with $a_i \in \mathbb{F}_{2^k}$. Then the matrix $M = \mathrm{vandermonde}(a_1, \ldots, a_n)$ is defined by the entries

$$m_{i,j} := a_i^{j-1}, \qquad \text{with } 1 \leq i, j \leq n$$

and is called *Vandermonde* matrix.

Given two Vandermonde matrices A and B with pairwise different a_i, b_j, then the matrix AB^{-1} is MDS, see [LF04, Theorem 2]. Furthermore, if $a_i = b_i + \Delta$ for all i and an arbitrary nonzero Δ, then the matrix AB^{-1} is also involutory [LF04; Saj+12].

[LF04] Lacan and Fimes, "Systematic MDS erasure codes based on Vandermonde matrices"

[Saj+12] Sajadieh et al., "On construction of involutory MDS matrices from Vandermonde Matrices in GF(2^q)"

5.2.3 *Specially Structured Matrix Constructions*

Other constructions, such as circulant, Hadamard, or Toeplitz, are not per se MDS, but they have the advantage that they greatly reduce the search space by restricting the number of submatrices that appear in the matrix. For circulant matrices, this was e. g. already noted by Daemen et al. [Dae+97].

[Dae+97] Daemen et al. "The Block Cipher Square"

In order to generate a random MDS matrix with one of these constructions, we can choose random elements for the matrix and then check for the

MDS condition. Because of many repeated submatrices, the probability to find an MDS matrix is much higher then for a fully random matrix.

Definition 119 (Circulant matrices)**.** A *right circulant* $n \times n$ matrix is defined by the elements of its first row $a_1, \dots, a_n$ as

$$M = \mathrm{circ}_r(a_1, \dots, a_n) \coloneqq \begin{pmatrix} a_1 & a_2 & \cdots & a_n \\ a_n & a_1 & \cdots & a_{n-1} \\ \vdots & & \ddots & \vdots \\ a_2 & \cdots & a_n & a_1 \end{pmatrix}.$$

A *left circulant* $n \times n$ matrix is analogously defined as

$$M = \mathrm{circ}_\ell(a_1, \dots, a_n) \coloneqq \begin{pmatrix} a_1 & a_2 & \cdots & a_n \\ a_2 & a_3 & \cdots & a_1 \\ \vdots & & \ddots & \vdots \\ a_n & a_1 & \cdots & a_{n-1} \end{pmatrix}.$$

While in the literature circulant matrices are almost always right circulant, left circulant matrices are equally fine for cryptographic applications. The often noted advantage of right circulant matrices, the ability to implement the multiplication serialised and with shifts in order to save XORs, of course also applies to left circulant matrices. Additionally, it is easy to see that the branch number of left and right circulant matrices are equal, $\mathrm{bn}(\mathrm{circ}\, r(a_1, \dots, a_n)) = \mathrm{bn}(\mathrm{circ}_\ell(a_1, \dots, a_n))$, since the matrices only differ in a permutation of the rows. Thus, for cryptographic purposes, it does not matter if a matrix is right circulant or left circulant and we will therefore simply talk about circulant matrices in general. The common practice of restricting the matrix entries to elements from a finite field comes with the problem that circulant involutory MDS matrices over finite fields do not exist, see [JA09]. But Li and Wang [LW16] showed that this can be avoided by taking the matrix elements from the general linear group.

[JA09] Jr. and Abrahão, "A New Involutory MDS Matrix for the AES"

[LW16] Li and Wang "On the Construction of Lightweight Circulant Involutory MDS Matrices"

Definition 120 (Hadamard matrix)**.** A *(finite field) Hadamard* matrix M is of the form

$$M = \begin{pmatrix} M_1 & M_2 \\ M_2 & M_1 \end{pmatrix},$$

where M_1 and M_2 are either Hadamard matrices themselves or one-dimensional.

The biggest advantage of Hadamard matrices is the possibility to construct involutory matrices. If we choose the elements of our matrix such that the first row sums to one,[1] the resulting matrix is involutory, see [GR13].

Definition 121 (Toeplitz matrix)**.** An $n \times n$ *Toeplitz* matrix M is defined by the elements of its first row $a_1, \dots, a_n$ and its first column $a_1, a_{n+1}, \dots, a_{2n-1}$ as

$$M = \mathrm{toep}(a_1, \dots, a_n, a_{n+1}, \dots, a_{2n-1}) \coloneqq \begin{pmatrix} a_1 & a_2 & \cdots & a_n \\ a_{n+1} & a_1 & \ddots & a_{n-1} \\ \vdots & \ddots & \ddots & \vdots \\ a_{2n-1} & a_{2n-2} & \cdots & a_1 \end{pmatrix},$$

that is, every element defines one minor diagonal of the matrix.

[1] Or to the identity matrix, if the elements are from the general linear group.

To the best of our knowledge, Sarkar and Syed [SS16b] were the first to scrutinise Toeplitz matrices in the context of XOR counts.

[SS16b] Sarkar and Syed "Lightweight Diffusion Layer: Importance of Toeplitz Matrices"

Finally, the subfield construction was first used to construct lightweight linear layers in the WHIRLWIND hash function [Bar+10, Section 2.2.2] and later used in [Cho+12; Alb+14; Kho+14; Sim+15; Jea+17b]. As its name suggests, the subfield construction was originally defined only for matrices over finite fields: a matrix with coefficients in $\mathbb{F}_{2^k}$ can be used to construct a matrix with coefficients in $\mathbb{F}_{2^{2k}}$. Here, we use the natural extension to binary matrices.

[Bar+10] Barreto et al., "Whirlwind: a new cryptographic hash function"

Definition 122 (Subfield matrix)**.** Given an $n \times n$ matrix M with entries $m_{i,j} \in \mathbb{F}_2^{k\times k}$. The subfield construction of M is then an $n \times n$ matrix M' with

$$M' = \text{subfield}(M) := \left(m'_{i,j}\right),$$

where each $m'_{i,j} = \left(\begin{smallmatrix} m_{i,j} & 0 \\ 0 & m_{i,j} \end{smallmatrix}\right) \in \mathbb{F}_2^{2k\times 2k}$.

In other words, for Definition 122 we interpret the elements with which we multiply M' as elements of the product ring $\mathbb{F}_2^k \times \mathbb{F}_2^k$ and the linear layer acts as multiplying one copy of M with the first parts of the input and another copy of M with the second parts of the input. This definition is straightforward to extend for more than one copy of the matrix M.

The subfield construction has some very useful properties, see [Bar+10; Kho+14; Sim+15; Jea+17b].

Lemma 123. *For the subfield construction, the following properties hold:*

1. *Let M be a matrix that can be implemented with m XORs. Then the matrix $M' = \text{subfield}(M)$ can be implemented with $2m$ XORs.*

2. *Let M be an MDS matrix for k-bit words. Then $M' = \text{subfield}(M)$ is MDS for $2k$-bit words.*

3. *Let M be an involutory matrix. Then $M' = \text{subfield}(M)$ is also involutory.*

Proof.

(1) Due to the special structure of the subfield construction, we can split the multiplication by M' into two multiplications by M, each on one half of the input bits. Hence, the XOR count doubles.

(2) We want to show that $\text{hw}_{2k}(u)+\text{hw}_{2k}(M'u) \geq n+1$ for every nonzero u. We split u into two parts u_1 and u_2, each containing alternating halves of the elements of u. As described in [Kho+14], the multiplication of M' and u is the same as the multiplication of the original matrix M and each of the two u_i, if we combine the results according to our splitting. Let $t = \text{hw}_{2k}(u) > 0$. Then, we have $t \geq \text{hw}_k(u_1)$ and $t \geq \text{hw}_k(u_2)$. Without loss of generality, let $\text{hw}_k(u_1) > 0$. Since M is MDS for k-bit words, we have $\text{hw}_k(Mu_1) \geq n - t + 1$ which directly leads to $\text{hw}_{2k}(M'u) \geq n - t + 1$.

[Kho+14] Khoo et al., "FOAM: Searching for Hardware-Optimal SPN Structures and Components with a Fair Comparison"

(3) As in the above proof, this property is straightforward to see. We want to show that $M'M'u = u$ for any vector u. Again, we split u into two parts, u_1 and u_2, each containing alternating halves of the elements of u. Now, we need to show that $MMu_i = u_i$. This trivially holds, as M is involutory.

With respect to cryptographic designs, this means the following: assume we have an implementation with m XORs for an (involutory) $n \times n$ MDS matrix and k-bit S-boxes. By applying the subfield construction we then easily find an implementation with $2m$ XORs for an (involutory) $n \times n$ MDS matrix and $2k$-bit S-boxes.

We now turn to the XOR count metric and related work on it.

5.3 XOR COUNTS AND OPTIMISATION STRATEGIES

In 2014, Khoo et al. [Kho+14] introduced the notion of XOR count as a metric to compare the area-efficiency of matrix multiplications. Following that, there has been a lot of work [Sim+15; Bei+16a; LS16; LW16; SS16a; SS16b; SS17; Jea+17b; LW17; Zho+17] on finding MDS matrices that can be implemented with as few XOR gates as possible in the round-based scenario.

In an independent line of research, the problem of implementing binary matrix multiplications with as few XORs as possible was extensively studied [Paa97; Boy+08; BP10; FSK10; FS10; Boy+13; Boy+17; Vis+17].

In this section, we depict these two fields of research and show how they can be combined.

5.3.1 *The XOR count metric*

Let us first recall the scenario. In a round-based implementation the matrix is implemented as a fully unrolled circuit. Thus, in the XOR count metric, the goal is to find a matrix that can be implemented with a circuit of as few (2-input) XOR gates as possible. Basically every paper on this topic defines the XOR count as a property of the according matrix M. But we will see in the remainder of this chapter that the actual XOR count highly depends on the *implementation* of M. To reflect this, we use the following definition.

Definition 124 (XOR count)**.** For a given a matrix M denote by $\mathrm{Impl}(M)$ an implementation that computes the map $x \mapsto M \cdot x$. The *XOR count* of $\mathrm{Impl}(M)$ is then defined as the number of XOR operations this implementation needs.

Obviously, the implementation of a matrix, and thus the matrix' XOR count, is not unique. If we want to highlight that an implementation was derived under some specific heuristic H, we denote this by $\mathrm{Impl}_H(\cdot)$. Of course, the matrix, or its implementation, has to fulfill some criteria, typically the matrix is MDS. For finding matrices with a low XOR count, the question of how to create a circuit for a given matrix must be answered.

5.3.2 *Local Optimisations*

[Kho+14] Khoo et al., "FOAM: Searching for Hardware-Optimal SPN Structures and Components with a Fair Comparison"

The usual way for finding an implementation of $n \times n$ matrices over $\mathbb{F}_{2^k}$ was introduced in [Kho+14]. As each of the n output components of a matrix-vector multiplication is computed as a sum over n products, the implementation is divided into two parts. Namely the single multiplications on the one hand and addition of the products on the other hand. As $\mathbb{F}_{2^k} \cong \mathbb{F}_2^k$, an addition of two elements from $\mathbb{F}_2^k$ requires k XORs and thus adding up the products for all rows requires $n(n-1)k$ XORs in the case of an MDS matrix where every element is nonzero. If one implements the matrix like this, these $n(n-1)k$ XORs are a fixed part that cannot be changed. Accordingly, many papers [Bei+16a; LS16; LW16; Zho+17; Jea+17b] just state the number of XORs for the single field multiplications when presenting results. The other costs are regarded as inevitable. The goal then boils down to constructing matrices with elements for which multiplication can be implemented with few XORs. Thus, the original goal of finding a global implementation for the matrix is approached by locally looking at the single matrix elements.

[Jea+17b] Jean et al., "Optimizing Implementations of Lightweight Building Blocks"

To count the number of XORs for implementing a single multiplication with an element $\alpha \in \mathbb{F}_{2^k}$, the multiplication matrix $T_\alpha \in \mathbb{F}_2^{k \times k}$ is considered. Such a matrix can be implemented in a straightforward way with $\text{hw}(T_\alpha) - k$ XORs by simply implementing every XOR of the output components. We call this the *naïve* implementation of a matrix and when talking about the naïve XOR count of a matrix, we mean the $\text{hw}(T_\alpha) - k$ XORs required for the naïve implementation. In [Jea+17b], this is called d-XOR. It is the easiest and most frequently used method of counting XORs. Of course, in the same way we can also count the XORs of other matrices over $\mathbb{F}_2^{k \times k}$, i. e. also matrices that were not originally defined over finite fields.

[Bei+16a] Beierle et al., "Lightweight Multiplication in GF(2^n) with Applications to MDS Matrices"

For improving the XOR count of the single multiplications, two methods have been introduced. First, if the matrix is defined over some finite field, one can consider different field representations that lead to different multiplication matrices with potentially different Hamming weights, see [Sim+15; Bei+16a; SS16a]. Second, by reusing intermediate results, a $k \times k$ binary matrix might be implemented with less than $\text{hw}(M) - k$ XORs, see [Bei+16a; Jea+17b]. In [Jea+17b], this is called s-XOR. The according definitions from [Jea+17b] and [Bei+16a] require that all operations must be carried out on the input registers. If we want to compute an s-XOR count for a matrix, we are thus looking for an implementation that is constrained to use no temporary registers. However, as we consider round-based hardware implementations, there is no need to avoid temporary registers since these are merely wires between gates.

Nowadays, the XOR count of implementations is mainly dominated by the $n(n-1)k$ XORs for putting together the locally optimised multiplications. Lastly, we seem to hit a threshold and new results often improve existing results only by very few XORs. The next natural step is to shift the focus from local optimisation of the single elements to the global optimisation of the whole matrix. This was also formulated as future work in [Jea+17b]. As described in Section 5.2, we can use the binary representation to write an $n \times n$ matrix over $\mathbb{F}_{2^k}$ as a binary $nk \times nk$ matrix. First we note, that the naïve XOR count of the binary representation is exactly the naïve XOR count of implementing each element multiplication and finally adding the results. But if we look at the optimisation technique of reusing intermediate results for the whole $nk \times nk$ matrix, we now have many more degrees of freedom. For the MixColumn matrix there already exists some work that goes beyond local optimisation. An implementation with 108 XORs has been presented in [Sat+01; Ban+16a; Ban+16b] and an implementation with 103 XORs in [Jea+17a]. A first step to a global optimisation algorithm was done in [Zha+16]. However, their heuristic did not yield very good results and they finally had to go back to optimising submatrices.

Interestingly, much better algorithms for exactly this problem are already known from a different line of research.

[Jea+17a] Jean et al., "Bit-Sliding: A Generic Technique for Bit-Serial Implementations of SPN-based Primitives - Applications to AES, PRESENT and SKINNY"

[Zha+16] Zhao et al., *Designing Optimal Implementations of Linear Layers (Full Version)*

5.3.3 *Global Optimisations*

Implementing binary matrices with as few XOR operations as possible is also known as the problem of finding the *shortest linear Straight-Line Program (SLP)* [FS10; Boy+13] over the finite field with two elements. Although this problem is NP-hard [Boy+08; Boy+13], attempts have been made to find exact solutions for the minimum number of XORs. Fuhs and Schneider-Kamp [FSK10; FS10] suggested to reduce the problem to satisfiability of Boolean logic. They presented a general encoding scheme for deciding if a matrix can be implemented with a certain number of XORs. Now, for finding the optimal implementation, they repeatedly use SAT solvers for a decreasing number of XORs. Then, when they know that a matrix can be implemented with ℓ XORs, but cannot be implemented with $\ell - 1$ XORs, they are able to present ℓ as the optimal XOR count. They used this technique to search for the minimum number of XORs necessary to compute a binary matrix of size 21×8, which is the first linear part of the AES S-box, when it is decomposed into two linear parts and a minimal non-linear core. While it worked to find a solution with 23 XORs and to show that no solution with 20 XORs exists, it turned out to be infeasible to prove that a solution with 22 XORs does not exist and that 23 is therefore the minimum. In general, this approach quickly becomes infeasible for larger matrices. Stoffelen [Sto16] applied it successfully to a small 7×7 matrix, but did not manage to find a provably minimal solution with a specific matrix of size 19×5. However, there do exist heuristics to efficiently find short linear SLPs also for larger binary matrices.

[FS10] Fuhs and Schneider-Kamp, "Synthesizing Shortest Linear Straight-Line Programs over GF(2) Using SAT"

[Boy+13] Boyar et al., "Logic Minimization Techniques with Applications to Cryptology"

[Boy+08] Boyar et al., "On the Shortest Linear Straight-Line Program for Computing Linear Forms"

[FSK10] Fuhs and Schneider-Kamp, "Optimizing the AES S-Box using SAT"

[Sto16] Stoffelen "Optimizing S-Box Implementations for Several Criteria Using SAT Solvers"

Back in 1997, Paar [Paa97] studied how to optimise the arithmetic used by Reed-Solomon encoders. Essentially, this boils down to reducing the number of XORs that are necessary for a constant multiplier over the field $\mathbb{F}_{2^k}$. Paar described two algorithms that find a local optimum. Intuitively, the idea of the algorithms is to iteratively eliminate *common subexpressions*. Let T_α be the multiplication matrix, to be applied to a variable field element $x = (x_1, \ldots, x_k) \in \mathbb{F}_2^k$. The first algorithm for computing $\operatorname{Impl}(T_\alpha)$, denoted PAAR1 in the rest of this work, finds a pair (i, j), with $i \neq j$, where the bitwise AND between columns i and j of T_α has the highest Hamming weight. In other words, it finds a pair (x_i, x_j) that occurs most frequently as subexpression in the output bits of $T_\alpha x$. The XOR between those is then computed, and T_α and $\operatorname{Impl}(T_\alpha)$ are updated accordingly, with $x_i + x_j$

[Paa97] Paar "Optimized Arithmetic for Reed-Solomon Encoders" as newly available variable. This is repeated until there are no common subexpressions left.

The second algorithm, denoted PAAR2, is similar, but differs when multiple pairs are equally common. Instead of just taking the first pair, it recursively tries all of them. The algorithm is therefore much slower, but can yield slightly improved results. Compared to the naïve XOR count, Paar noted an average reduction in the number of XORs of 17.5% for matrices over $\mathbb{F}_{2^4}$ and 40% for matrices over $\mathbb{F}_{2^8}$.

[Ber09] Bernstein “Optimizing linear maps modulo 2”

[Paa97] Paar, “Optimized Arithmetic for Reed-Solomon Encoders”

[BC14] Bernstein and Chou, “Faster Binary-Field Multiplication and Faster Binary-Field MACs”

In 2009, Bernstein [Ber09] presented an algorithm for efficiently implementing linear maps modulo 2. Based on this and on [Paa97], a new algorithm was presented in [BC14]. However, the algorithms from [Ber09; BC14] have a different framework in mind and yield a higher number of XORs compared to [Paa97].

[Boy+08] Boyar et al. “On the Shortest Linear Straight-Line Program for Computing Linear Forms”

Paar’s algorithms lead to so-called *cancellation-free* implementations. This means that for every XOR operation $u + v$, none of the input bit variables x_i occurs in both u and v. Thus, the possibility that two variables cancel each other out is never taken into consideration, while this may in fact yield a more efficient solution in terms of the total number of XORs. In 2008, Boyar et al. [Boy+08] showed that cancellation-free techniques can often not be expected to yield optimal solutions for non-trivial inputs. They also showed that, even under the restriction to cancellation-free programs, the problem of finding an optimal SLP is NP-complete.

[BP10] Boyar and Peralta “A New Combinational Logic Minimization Technique with Applications to Cryptology”

Around 2010, Boyar and Peralta [BP10] came up with a heuristic that is not cancellation-free and that improved on Paar’s algorithms in most scenarios. Their idea was to keep track of a distance vector that contains, for each targeted expression of an output bit, the minimum number of additions of the already computed intermediate values that are necessary to obtain that target. To decide which values will be added, the pair that minimises the sum of new distances is picked. If there is a tie, the pair that maximises the Euclidean norm of the new distances is chosen. Additionally, if the addition of two values immediately leads to a targeted output, this can always be done without searching further. This algorithm works very well in practice, although it is slower compared to PAAR1.

Next to using the Euclidean norm as tie breaker, they also experimented with alternative criteria. For example, choosing the pair that maximises the Euclidean norm minus the largest distance, or minus the difference between the two largest distances. The results were then actually very similar. Another tie-breaking method is to flip a coin and choose a pair randomly. The algorithm is now no longer deterministic and can be run multiple times. The lowest result can then be used. This performed slightly better, but of course processing again takes longer.

[Boy+13] Boyar et al., “Logic Minimization Techniques with Applications to Cryptology”

The results of [Boy+08; BP10] were later improved and published in the journal of cryptology [Boy+13].

[Vis+17] Visconti et al. *Improved upper bounds for the expected circuit complexity of dense systems of linear equations over GF(2)*

In early 2017, Visconti et al. [Vis+17] explored the special case where the binary matrix is dense. They improved the heuristic on average for dense matrices by first computing a *common path*, an intermediate value that contains most variables. The original algorithm then starts from this common path. At BFA 2017, Boyar et al. presented an improvement that simultaneously reduces the number of XORs and the depth of the resulting circuit, see [Boy+17].

We refer to this family of heuristics [Boy+08; BP10; Boy+13; Boy+17; Vis+17] as the BP heuristics.

[Boy+17] Boyar et al., *Low-Depth, Low-Size Circuits for Cryptographic Applications*

5.4 APPLICATION AND EVALUATION OF THE HEURISTICS

Using the techniques described above, we now give XOR counts for optimised implementations of published matrices. Next, we analyze the statistical behaviour of different matrix constructions. Finally we summarise the to date best known matrices.

5.4.1 *Improved Implementations of Matrices*

Using the heuristic methods that are described in the previous section, we can easily and significantly reduce the XOR counts for many matrices that have been used in the literature. The running times for the optimisations are in the range of seconds to minutes. Table 5.3 at the end of this chapter lists results for matrices that have been suggested in previous works where it was an explicit goal to find a lightweight MDS matrix. While the constructions themselves will be compared in Section 5.4.2, this table deals with the suggested instances.

A number of issues arise from this that are worth highlighting. First of all, it should be noted that without any exception, the XOR count for every matrix could be reduced with little effort. Second, it turns out that there are many cases where the $n(n-1)k$ XORs for summing the products for all rows is not even a correct lower bound. In fact, all the 4×4 matrices over $\mathrm{GL}(4, \mathbb{F}_2)$ that we studied can be implemented in *at most* $4 \cdot 3 \cdot 4 = 48$ XORs.

What may be more interesting, is whether the XOR count, as it was used previously, is in fact a good predictor for the actual implementation cost as given by the heuristic methods. Here we see that there are some differences. For example, [LW16]'s circulant 4×4 matrices over $\mathrm{GL}(8, \mathbb{F}_2)$ first compared very favorably, but we now find that the subfield matrix of [Jea+17b] requires fewer XORs.

[LW16] Li and Wang, "On the Construction of Lightweight Circulant Involutory MDS Matrices"

[Jea+17b] Jean et al., "Optimizing Implementations of Lightweight Building Blocks"

Regarding involutory matrices, it was typically the case that there was an extra cost involved to meet this additional criterion. However, the heuristics sometimes find implementations with even fewer XORs than the non-involutory matrix that was suggested. See for example the matrices of [SS16b] in the table.

[SS16b] Sarkar and Syed, "Lightweight Diffusion Layer: Importance of Toeplitz Matrices"

Aside from these matrices, we also looked at MDS matrices that are used by various ciphers and hash functions. Table 5.4 lists their results, it can be found at the end of this chapter, too. Not all MDS matrices that are used in ciphers are incorporated here. In particular, LED [Guo+11a], PHOTON [Guo+11b], and PRIMATES [And+14] use efficient serialised MDS matrices. Comparing these to our "unrolled" implementations would be somewhat unfair.

[Guo+11a] Guo et al., "The LED Block Cipher"

[Guo+11b] Guo et al., "The PHOTON Family of Lightweight Hash Functions"

[And+14] Andreeva et al., *PRIMATEs v1.02*

The implementation of the MDS matrix used in AES with 97 XORs is, to the best of our knowledge, the most efficient implementation so far and improves on the previous implementation of 103 XORs, reported by [Jea+17a].

As a side note, cancellations do occur in this implementation, we thus conjecture that such a low XOR count is not possible with cancellation-free programs.

5.4.2 *Statistical Analysis*

Several constructions for building MDS matrices are known. But it is not clear which one is the best when we want to construct matrices which can yield an implementation with a low XOR count. In this section, we present experimental results on different constructions and draw conclusions for the designer. We also examine the correlation between naïve and heuristically improved XOR counts. When designing MDS matrices with a low XOR count, we are faced with two major questions. First, which construction is preferable? Our intuition in this case is a better construction has better statistical properties compared to an inferior construction. We are aware that the statistical behaviour of a construction might not be very important for a designer who only looks for a single, very good instance. Nevertheless we use this as a first benchmark. Second, is it a good approach to choose the matrices as sparse as possible? In order to compare the listed constructions, we construct random instances of each and then analyze their implementations with statistical means.

Building Cauchy and Vandermonde matrices is straightforward as we only need to choose the defining elements randomly from the underlying field. For the other constructions, we use the following backtracking method to build random MDS constructions of dimension 4 × 4. Choose the new random elements from $\mathrm{GL}(k, \mathbb{F}_2)$ that are needed for the matrix construction in a step-by-step manner. In each step, construct all new square submatrices. If any of these is not invertible, discard the chosen element and try a new one. In the case that no more elements are left, go one step back and replace that element with a new one, then again check the according square submatrices, and so on. Eventually, we end up with an MDS matrix because we iteratively checked that every square submatrix is invertible. The method is also trivially derandomisable, by not choosing the elements randomly, but simply enumerating them in any determined order.

Apart from applying this method to the above mentioned constructions, we can also use it to construct an *arbitrary*, i. e. unstructured, matrix that is simply defined by its 16 elements. This approach was already described in [Jea+17b].

[Jea+17b] Jean et al., "Optimizing Implementations of Lightweight Building Blocks"

In this manner, we generated 1 000 matrices for each construction and computed the XOR count distributions for the naïve, the optimised PAAR1, and BP implementations. Table 5.1 lists the statistical parameters of the resulting distributions and Figure 5.2 (at the end of the chapter) depicts them (the sample size N is the same for Table 5.1, Figures 5.1 and 5.2, and [Kra+17b, Appendix A, Figures 3 to 6]).

[Kra+17b] Kranz et al., "Shorter Linear Straight-Line Programs for MDS Matrices"

The most obvious characteristic of the statistical distributions is that the means μ do not differ much for all randomised constructions. The variance σ^2 on the contrary differs much more. This is most noticeable for the naïve implementation, while the differences get much smaller when the implementation is optimised with the PAAR1 or BP heuristic. One might think

	Naïve		Paar1		BP	
Construction	μ	σ^2	μ	σ^2	μ	σ^2
4×4 matrices over $\mathrm{GL}(4, \mathbb{F}_2)$						
Cauchy	120.7	77.3	62.9	11.0	53.1	4.0
Circulant	111.8	117.1	60.4	19.2	52.1	7.1
Hadamard	117.5	99.6	60.2	17.8	52.4	6.9
Toeplitz	112.8	39.9	59.9	7.4	51.3	3.9
Vandermonde	120.6	87.6	62.2	8.1	52.9	3.1
enumerated 4×4 matrices over $\mathrm{GL}(4, \mathbb{F}_2)$						
Circulant	82.9	53.0	54.9	13.5	50.1	6.7
Hadamard	102.1	76.0	56.7	20.6	50.6	8.0
Toeplitz	86.1	43.9	55.3	8.3	49.4	3.9
Arbitrary	80.5	8.3	49.7	3.2	44.5	1.8
4×4 matrices over $\mathrm{GL}(8, \mathbb{F}_2)$						
Cauchy	454.1	467.2	215.1	39.6	—	—
Vandermonde	487.3	597.4	220.2	44.3	—	—
4×4 subfield matrices over $\mathrm{GL}(4, \mathbb{F}_2)$						
Cauchy	241.1	312.1	125.8	44.2	—	—
Vandermonde	240.6	452.8	121.8	47.1	—	—

Table 5.1: Distributions for differently optimised implementations. By μ we denote the mean and σ^2 is the variance. The sample size N for all distributions is $N = 1000$.

that the construction with the lowest optimised average XOR count, which is for matrices over $\mathrm{GL}(4, \mathbb{F}_2)$ the arbitrary construction with enumerated elements, yields the best results. However, the best matrix we could find for this dimension was a Hadamard matrix. An explanation for this might be that the higher variance leads to some particularly bad and some particularly good results.

The graphs in Figure 5.2 convey a similar hypothesis. Looking only at the naïve implementation, we can notice some differences. For example circulant matrices seem to give better results than, e. g., Hadamard matrices. Additionally, the naïve XOR count increases step-wise as not every possible count occurs. But when optimising the implementation, the distributions get smoother and more similar.

We conclude that all constructions give similarly good matrices when we are searching for the matrix with the lowest XOR count, with one important exception. For randomly generated matrices the XOR count increases by a factor of four, if we double the parameter k. Table 5.1 covers this for Cauchy and Vandermonde matrices. We do not compute the statistical properties for Circulant, Hadamard and Toeplitz matrices with elements of $\mathrm{GL}(8, \mathbb{F}_2)$, as the probability to find a random MDS instance for these constructions is quite low. Thus, generating enough instances for a meaningful statistical

TABLE 5.2: MDS matrices with the lowest known XOR count. Matrices in the lower half are involutory.

Type	Previously Best Known		New Best Known	
$\mathrm{GL}(4,\mathbb{F}_2)^{4\times4}$	58	[SS16b; Jea+17b]	36*	Eq. (5.1)
$\mathrm{GL}(8,\mathbb{F}_2)^{4\times4}$	106	[LW16]	72	Eq. (5.2)
$(\mathbb{F}_2[x]/\mathtt{0x13})^{8\times8}$	384	[Sim+15]	196†	Eq. (5.3)
$\mathrm{GL}(8,\mathbb{F}_2)^{8\times8}$	640	[LS16]	392	Eq. (5.4)
$(\mathbb{F}_2[x]/\mathtt{0x13})^{4\times4}$	63	[Jea+17b]	42*	[SS16b]
$\mathrm{GL}(8,\mathbb{F}_2)^{4\times4}$	126	[Jea+17b]	84	Eq. (5.5)
$(\mathbb{F}_2[x]/\mathtt{0x13})^{8\times8}$	424	[Sim+15]	212†	Eq. (5.6)
$\mathrm{GL}(8,\mathbb{F}_2)^{8\times8}$	736	[Jea+17b]	424	Eq. (5.7)

* Computed with heuristic from [Boy+13].
† Computed with heuristic from [Paa97].

comparison is computationally tough and – as we deduce from a much smaller sample size – the statistical behaviour looks very similar to that of the Cauchy and Vandermonde matrices. Instead, and as already mentioned in Lemma 123, the subfield construction has a much more interesting behaviour. Its implementation simply doubles the XOR count. The lower half of Table 5.1 confirms this behaviour.

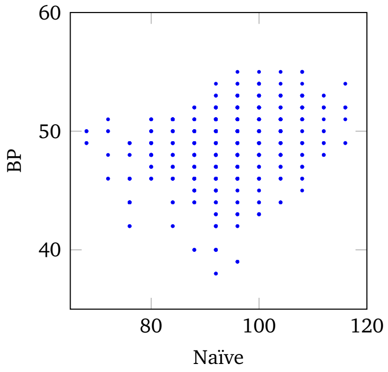


FIGURE 5.1: Correlations between naïve and BP XOR counts for enumerated Hadamard matrices.

[Kra+17b] Kranz et al., “Shorter Linear Straight-Line Programs for MDS Matrices”

Thus, when it is computationally infeasible to exhaustively search through all possible matrices, it seems to be a good strategy to use the subfield construction with the best known results from smaller dimensions. This conclusion is confirmed by the fact that our best results for matrices over $\mathrm{GL}(8,\mathbb{F}_2)$ are always subfield constructions based on matrices over $\mathrm{GL}(4,\mathbb{F}_2)$.

Next, we want to approach the question if choosing MDS matrices with low Hamming weight entries is a good approach for finding low XOR count implementations. To give a first intuition of the correlation between naïve and optimised implementations, we plot the naïve XOR count against the optimised one. For one exemplary plot see Figure 5.1, which corresponds to the construction that we used to find the best 4×4 MDS matrix for $k = 4$. The remaining plots can be found in the paper, see [Kra+17b, Appendix A, Figures 3 to 6].

In Figure 5.1 one can see that several options can occur. While there is a general tendency of higher naïve XOR counts leading to higher optimised XOR counts, the contrary is also possible. For example, there are matrices which have a low naïve XOR count (left in the plot), while still having a somewhat high optimised XOR count (top part of the plot). But there are also matrices where a higher naïve XOR count results in a much better optimised XOR count. The consequence is that we cannot restrict ourselves to very sparse matrices when searching for the best XOR count, but also have to take more dense matrices into account. A possible explanation for this behaviour is that the heuristics have more possibilities for optimisations, when the matrix is not sparse.

5.4.3 *Best results*

Let us conclude by specifying the MDS matrices with the currently best implementations. The notation $M_{n,k}$ denotes an $n \times n$ matrix with entries from $\mathrm{GL}(k, \mathbb{F}_2)$, an involutory matrix is labeled with the superscript i. Table 5.2 covers non-involutory and involutory matrices of dimension 4 × 4 and 8 × 8 over $\mathrm{GL}(4, \mathbb{F}_2)$ and $\mathrm{GL}(8, \mathbb{F}_2)$. $M_{8,4}$ and $M^i_{8,4}$ are defined over $\mathbb{F}_2[x]/\mathtt{0x13}$.

The matrices mentioned are the following:

$$M_{4,4} = \mathrm{hadamard}(\begin{pmatrix} 0&0&0&1\\0&0&1&0\\0&1&0&0\\1&0&0&0 \end{pmatrix}, \begin{pmatrix} 0&0&1&1\\1&0&0&1\\1&1&0&0\\0&1&0&0 \end{pmatrix}, \begin{pmatrix} 1&1&0&1\\1&1&0&0\\0&1&0&1\\0&0&1&0 \end{pmatrix}, \begin{pmatrix} 1&1&0&0\\0&1&0&1\\1&0&1&1\\0&0&0&1 \end{pmatrix}) \tag{5.1}$$

$$M_{4,8} = \mathrm{subfield}(M_{4,4}) \tag{5.2}$$

$$M_{8,4} = \mathrm{cauchy}\begin{pmatrix} x^3{+}x^2, x^3, x^3{+}x{+}1, x{+}1, 0, x^3{+}x^2{+}x{+}1, x^2, x^2{+}x{+}1, \\ 1, x^2{+}1, x^3{+}x^2{+}x, x^3{+}1, x^3{+}x^2{+}1, x^2{+}x, x^3{+}x, x \end{pmatrix} \tag{5.3}$$

$$M_{8,8} = \mathrm{subfield}(M_{8,4}) \tag{5.4}$$

$$M^i_{4,8} \text{ is the subfield construction applied to [SS16b, Example 3]} \tag{5.5}$$

$$M^i_{8,4} = \mathrm{vandermonde}\begin{pmatrix} x^3{+}x{+}1, x{+}1, x^3{+}x^2{+}x, x^3{+}x^2{+}1, x^3{+}1, x^3, 0, x^3{+}x \\ x^2{+}x{+}1, x^3{+}x^2{+}x{+}1, x, 1, x^2{+}1, x^2, x^3{+}x^2, x^2{+}x \end{pmatrix} \tag{5.6}$$

$$M^i_{8,8} = \mathrm{subfield}(M^i_{8,4}) \tag{5.7}$$

All these matrices improve over the previously known matrices, with the only exception being the involutory matrix from [SS16b] of dimension 4 × 4 over $\mathrm{GL}(4, \mathbb{F}_2)$. $M_{4,4}$ was found after enumerating a few thousand Hadamard matrices, while $M_{8,4}$ and $M^i_{8,4}$ are randomly generated and were found after a few seconds. Every best matrix over $\mathrm{GL}(8, \mathbb{F}_2)$ uses the subfield construction.

[SS16b] Sarkar and Syed, "Lightweight Diffusion Layer: Importance of Toeplitz Matrices"

With these results we want to highlight that, when applying global optimisations, it is quite easy to improve (almost) all currently best known results. We would like to emphasise that our results should not be misunderstood as an attempt to construct matrices which cannot be improved.

All our implementations are publicly available online.[2]

[2] https://github.com/rub-hgi/shorter_linear_slps_for_mds_matrices

5.5 FURTHER WORK

Our work mainly highlights that in order to get a tighter estimate on how few XORs are really needed for a given matrix M, we have to take optimisation strategies for its implementation into account. As already mentioned, this implementation can be subject to several constraints which we did not further investigate. One important aspect in this direction is the depth of the critical path of an implementation. The depth is an important factor for the latency of an implementation and thus subject to a different optimisation goal. There is some related work, which already takes the depth of the critical path into consideration, e. g. [Boy+17].

[Boy+17] Boyar et al., *Low-Depth, Low-Size Circuits for Cryptographic Applications*

A recent work by Duval and Leurent [DL18] also compares proposed matrices with implementation depth in mind. Additionally, Duval and Leurent propose a graph-based search algorithm to find optimal implementations that yield an MDS matrix. Using their algorithm, the authors are able to find matrices with cheaper and less deep implementations. Especially when taking the circuit depth into account, their approach is obviously favourable

[DL18] Duval and Leurent "MDS Matrices with Lightweight Circuits" to ours. For example, the best implementations for 4 × 4 matrices over $\mathrm{GL}(4, \mathbb{F}_2)$ of [DL18] need 35, 37, resp. 41 XORs (with a depth of five, four, resp. three), while our implementation of $M_{4,4}$ has depth six, see [DL18, Table 1].

An interesting observation is that their implementations can be further optimised by synthesis tools like Yosys[3] – this indicates, and is also discussed by the authors, that the XOR count is not necessarily a good metric of the actual implementation cost in silicon. The example Duval and Leurent give are Field Programmable Gate Arrays (FPGAs), where the XORs are implemented using internal programmable Look Up Tables (LUTs). These LUTs can implement not only 2-input XORs, but also 3- or 4-input or even 6- or 8-input XORs with almost no overhead compared to a simple 2-input XOR. This can of course greatly improve existing implementations. Duval and Leurent further argue that the costs for different XOR gates in Application Specific Integrated Circuit (ASIC) synthesis differ more and thus not so big improvements can be expected.

In our opinion, this is not true. Standard cell libraries for ASIC design often offers 2-input and 3-input XORs and the area cost for 3-input XOR gates might be cheap enough that it pays out to use these gates in an optimised implementation. For example in the `UMCL18G212T3` library, see e. g. [Pos09, Table 2.1], the 2-input XOR costs 2.67 GE (gate equivalents), while a 3-input XOR gate costs 4.67 GE which is ≈ 87% of the cost for two 2-input XOR gates. A recent work by Banik et al. [Ban+19] explores these differences; one of their main findings is that different matrices are optimal for different libraries.

The observation that it is crucial to take library gate costs into account was also already made by Reyhani-Masoleh et al. [Rey+18] in the context of S-box implementation optimisations. Here the authors noted that the theoretical best implementations are in practice *worse* than less optimised implementations – due to the fact that the less optimised implementation uses cheaper standard cell gates than the optimised ones. Reyhani-Masoleh et al. thus suggest that either directly the synthesis tools or the information these tools use during synthesis, that is the individual costs and characteristics (e. g. delay) of the available gates, have to be included in the theoretical optimisation process.

The more problematic (obvious) point in our opinion then is that the resulting implementations are highly dependent on the targeted cell library and, see [Ban+19], different matrices are optimal for different cell libraries. On the other hand it is questionable how problematic it is if the area requirements for a matrix implementation differs by a handful of gate equivalents.

An other interesting research question is how the existing heuristics can be improved. The already mentioned work by Reyhani-Masoleh et al. [Rey+18] proposes some improvements for the BP heuristics, but evaluate their improvements only in the setting of S-box optimisations. Li et al. [Li+19b] also improves the BP heuristics, to make them depth aware. They are also able to find new involutory matrices with cheaper implementations.

[DL18] Duval and Leurent, “MDS Matrices with Lightweight Circuits”

[Pos09] Poschmann, “Lightweight Cryptography: Cryptographic Engineering for a Pervasive World”

[Ban+19] Banik et al. “More results on Shortest Linear Programs”

[Rey+18] Reyhani-Masoleh et al. “Smashing the Implementation Records of AES S-box”

[Li+19b] Li et al. “Constructing Low-latency Involutory MDS Matrices with Lightweight Circuits”

[3] http://www.clifford.at/yosys

Matrix	Naïve	Literature	Paar1	Paar2	BP
4 × 4 matrices over GL(4, $\mathbb{F}_2$)					
[Sim+15] (H)	68	20 + 48	50	48*	48
[LS16] (C)	60	12 + 48	49	46*	44
[LW16] (C)†	60	12 + 48	48	47*	44
[Bei+16a] (C)†	64	12 + 48	48	47	**42**
[SS16b] (T)	58	**10 + 48**	46	45*	43
[Jea+17b]	61	**10 + 48**	48	47	43
[Sim+15] (H, I)	72	24 + 48	52	48*	48
[LW16] (C, I)	68	20 + 48	48	48	48
[SS16b] (I)	64	16 + 48	50	48	**42**
[Jea+17b] (I)	68	**15 + 48**	51	47*	47
4 × 4 matrices over GL(8, $\mathbb{F}_2$)					
[Sim+15] (S)	136	40 + 96	100	98*	100
[LS16] (C)	128	28 + 96[1]	116	116	112
[LW16]	106	**10 + 96**	102	102	102
[Bei+16a] (C)	136	24 + 96	116	112*	110
[SS16b] (T)	123	24 + 96[1]	110	108	107
[Jea+17b] (S)	122	20 + 96	96	95*	**86**
[Sim+15] (S, I)	144	40 + 96[1]	104	101*	100
[LW16] (H, I)	136	40 + 96	101	97*	**91**
[SS16b] (I)	160	64 + 96	110	109*	100
[Jea+17b] (S, I)	136	**30 + 96**	102	100*	**91**
8 × 8 matrices over GL(4, $\mathbb{F}_2$)					
[Sim+15] (H)	432	**160 + 224**[1]	210	209*	**194**
[SS17] (T)	410	170 + 224	212	212*	204
[Sim+15] (H, I)	512	**200 + 224**[1]	222	222*	**217**
8 × 8 matrices over GL(8, $\mathbb{F}_2$)					
[Sim+15] (H)	768	256 + 448[1]	474	—	467
[LS16] (C)	688	**192 + 448**[1]	464	—	447
[Bei+16a] (C)	784	208 + 448[1]	506	—	498
[SS17] (T)	680	232 + 448	447	—	**438**
[Sim+15] (H, I)	816	320 + 448[1]	430	—	**428**
[Jea+17b] (H, I)	1152	**288 + 448**	620	—	599

* Stopped algorithm after three hours runtime.

† The authors of [Bei+16a; LW16] did not only give one matrix, but instead whole classes of MDS matrices. For [Bei+16a], we chose the canonical candidate from its class. For [LW16], we chose the matrix presented as an example in the paper.

[1] Reported by [Jea+17b].

Table 5.3: Comparison of 4 × 4 and 8 × 8 MDS matrices over GL(4, $\mathbb{F}_2$) and GL(8, $\mathbb{F}_2$). Bold entries in the Literature column are matrices with up to now lowest XOR counts, bold entries in the BP column are the matrices with now lowest XOR count. The matrix constructions are Circulant (C), Hadamard (H), Involutory (I), Subfield (S), and Toeplitz (T).

Table 5.4: Matrices used in ciphers or hash functions. Note that matrices in the lower part of the table, marked with [‖], are not MDS. Additionally these matrices are commonly not a target for “XOR count”-based implementation optimisations, as they are per se very efficiently implementable. The matrix constructions are Circulant (C), Hadamard (H), Involutory (I), and Subfield (S).

Cipher	Type	Naïve	Literature	Paar1	Paar2	BP
AES [DR02][‡] (C)	$(\mathbb{F}_2[x]/\texttt{0x11b})^{4\times4}$	152	$7+96$[1]	108	108[*]	97[†]
Anubis [BRa] (H, I)	$(\mathbb{F}_2[x]/\texttt{0x11d})^{4\times4}$	184	$80+96$[2]	122	121[*]	113
Clefia M_0 [Shi+07] (H)	$(\mathbb{F}_2[x]/\texttt{0x11d})^{4\times4}$	184	—[5]	121	121[*]	106
Clefia M_1 [Shi+07] (H)	$(\mathbb{F}_2[x]/\texttt{0x11d})^{4\times4}$	208	—[5]	121	121[*]	111
Fox mu4 [JV04]	$(\mathbb{F}_2[x]/\texttt{0x11b})^{4\times4}$	219	—[5]	144	143[*]	137
Twofish [Sch+98]	$(\mathbb{F}_2[x]/\texttt{0x169})^{4\times4}$	327	—[5]	151	149[*]	129
Fox mu8 [JV04]	$(\mathbb{F}_2[x]/\texttt{0x11b})^{8\times8}$	1257	—[5]	611	—	594
Grøstl [Gau+] (C)	$(\mathbb{F}_2[x]/\texttt{0x11b})^{8\times8}$	1112	$504+448$[2]	493	—	475
Khazad [BRb] (H, I)	$(\mathbb{F}_2[x]/\texttt{0x11d})^{8\times8}$	1232	$584+448$[2]	488	—	507
Whirlpool [BRc][§] (C)	$(\mathbb{F}_2[x]/\texttt{0x11d})^{8\times8}$	840	$304+448$[2]	481	—	465
Joltik [Jea+14] (H, I)	$(\mathbb{F}_2[x]/\texttt{0x13})^{4\times4}$	72	$20+48$[2]	52	48	48
SmallScale AES [Cid+05] (C)	$(\mathbb{F}_2[x]/\texttt{0x13})^{4\times4}$	72	—[5]	54	54	47
Whirlwind M_0 [Bar+10] (H, S)	$(\mathbb{F}_2[x]/\texttt{0x13})^{8\times8}$	488	$168+224$[2]	218	218[*]	212
Whirlwind M_1 [Bar+10] (H, S)	$(\mathbb{F}_2[x]/\texttt{0x13})^{8\times8}$	536	$184+224$[2]	246	244[*]	235
Qarma128 [Ava17][‖] (C)	$(\mathbb{F}_2[x]/\texttt{0x101})^{4\times4}$	64	—[5]	48	48	48
Aria [Kwo+03][‖] (I)	$(\mathbb{F}_2)^{128\times128}$	768	480[3]	416	—	—
Midori [Ban+15][‖,¶] (C)	$(\mathbb{F}_{2^4})^{4\times4}$	32	—[5]	24	24	24
Prince $\widehat{M}_0$, $\widehat{M}_1$ [Bor+12][‖]	$(\mathbb{F}_2)^{16\times16}$	32	—[5]	24	24	24
Pride L_0–L_3 [Alb+14][‖]	$(\mathbb{F}_2)^{16\times16}$	32	—[5]	24	24	24
Qarma64 [Ava17][‖] (C)	$(\mathbb{F}_2[x]/\texttt{0x11})^{4\times4}$	32	—[5]	24	24	24
Skinny64 [Bei+16b][‖]	$(\mathbb{F}_{2^4})^{4\times4}$	16	12[4]	12	12	12

[*] Stopped algorithm after three hours runtime.
[†] For the implementation see our GitHub repository.
[‡] Also used in other primitives, e. g. its predecessor Square [Dae+97], and Mugi [Wat+02].
[§] Also used in Maelstrom [Fil+06].
[¶] Also used in other ciphers, e. g. Mantis [Bei+16b], and Fides [Bil+13].
[‖] Not an MDS matrix.

[1] Reported by [Jea+17a].
[2] Reported by [Jea+17b].
[3] Reported by [Bir+04].
[4] Reported by the designers.
[5] We are not aware of any reported results for this matrix.

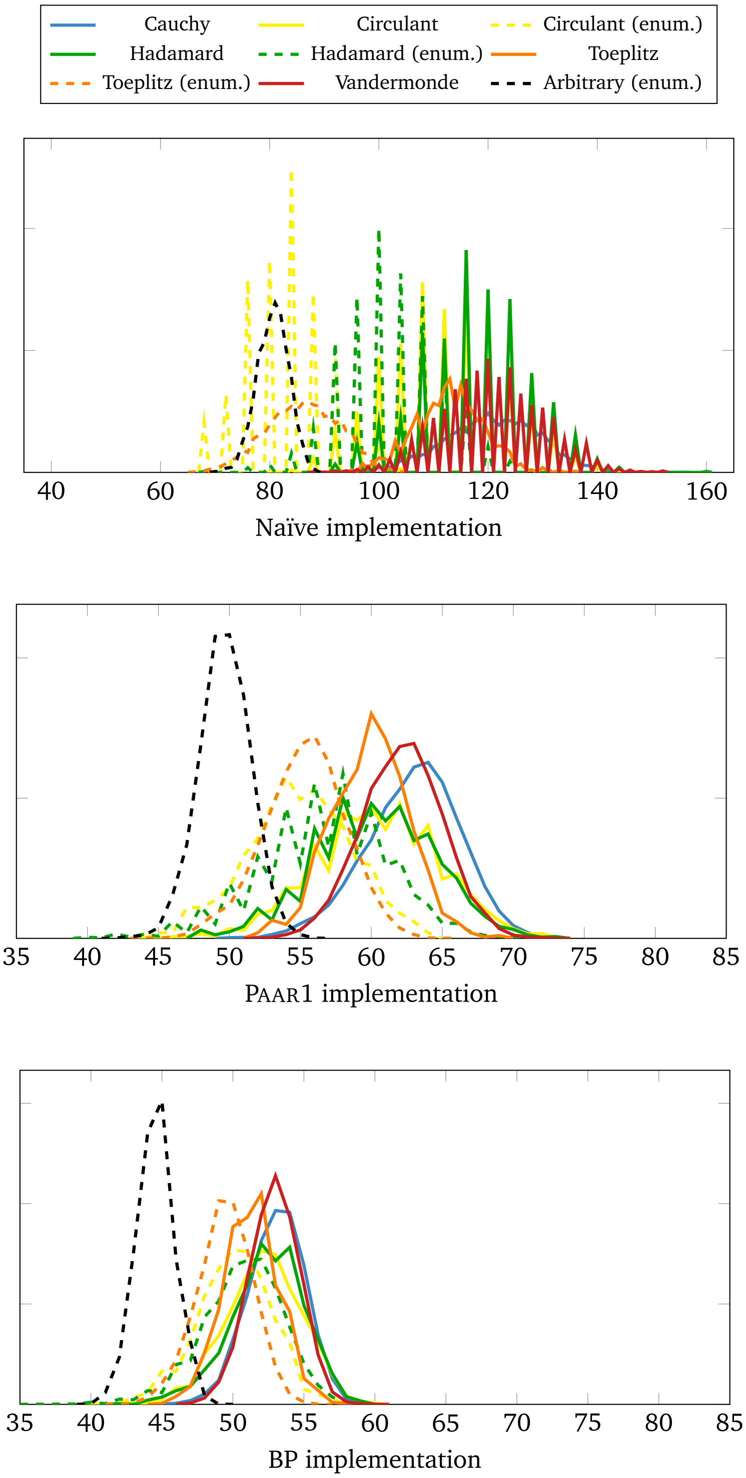


FIGURE 5.2: XOR count distributions for 4×4 MDS matrix constructions over $\mathrm{GL}(4, \mathbb{F}_2)$.

6

Propagating Truncated Differentials

"He that would perfect his work must first sharpen his tools."
—Confucius

▸ SYNOPSIS We propose an algorithm for propagating probability-one-subspace trails, and as a generalisation probability-one-truncated differentials, through an iterated function. Subsequently we discuss two applications. The first application of this algorithm, is an automated way to prove the resistance of an SPN cipher to subspace trail cryptanalysis. This is based on the article

> Gregor Leander, Cihangir Tezcan, and Friedrich Wiemer. "Searching for Subspace Trails and Truncated Differentials". In: *IACR Trans. Symm. Cryptol.* 2018.1 (2018), pp. 74–100. ISSN: 2519-173X. DOI: 10.13154/tosc.v2018.i1.74-100.

All authors contributed equally.

The second application of our propagation algorithm is about the automation of the key recovery part of a cryptanalytic attack on block ciphers, covered in Chapter 7.

After defining subspace trails and truncated differential attacks, we develop the propagation algorithm for subspace trails. Additionally to the generic algorithm from [Lea+18], we here give an improved algorithm for round functions of degree two. The following sections then describe a method to use this algorithm to prove the absence of subspace trails in an SPN construction.

[Lea+18] Leander et al., "Searching for Subspace Trails and Truncated Differentials"

6.1 INTRODUCTION

Truncated differentials have been introduced by Knudsen [Knu95] more than 20 years ago and since then been used in the analysis of many symmetric primitives, e. g. [Knu+99; BN14; Tez16; Gra17]. Informally, for truncated differentials, instead of trying to understand the exact behaviour of the output difference, one restricts to understand certain patterns that appear in the differences with an unusual high probability. Surprisingly, even the case of truncated differentials with probability one is not fully understood yet, as this work shows.

[Knu95] Knudsen, "Truncated and Higher Order Differentials"

Recently, new interest in truncated differentials has been triggered by [Gra+16] where the notion of subspace trail cryptanalysis has been defined and used to derive new interesting properties of the AES. As subspace trails are closely related to truncated differentials with probability one,

[Gra+16] Grassi et al., "Subspace Trail Cryptanalysis and its Applications to AES" those recent works again highlight that it is important to finally completely understand this topic.

A one round subspace trail can be captured as follows: Given a function F on n bit strings, a subspace trail is specified by two subspaces $U, V \subseteq \mathbb{F}_2^n$, such that any coset of U is mapped to a coset of V, that is

$$\forall a \in \mathbb{F}_2^n \quad \exists b \in \mathbb{F}_2^n \text{ such that } F(U+a) \subseteq V+b$$

and we denote this by $U \to V$.

Extending this concept to r rounds of an iterated block cipher with round function F, we consider subspace trails given by a tuple of vector spaces $(U_0, \ldots, U_r)$ such that for all i we have

$$U_i \to U_{i+1}.$$

The important question, both for the design as well as for the analysis of block ciphers, is how to identify the most powerful subspace trails, where most powerful basically corresponds to covering as many rounds as possible. In the case of truncated differentials, several heuristics have been used to solve this problem so far. For SPN ciphers, where each round function consists of a layer of parallel S-boxes followed by a linear mapping, the two most common ones are to ignore the details of the S-box and to restrict to the cases where U_0 only activates one S-box. While intuitively this approach seems to cover the best subspace trails, it seems hard to exclude the existence of better subspace trails outside those special cases. Indeed, to underline that the heuristic of activating a single S-box is not sufficient in general we provide several examples (see Section 6.4.1) where the best subspace trails actually activate more than one S-box.

For the designers of block ciphers it would be very convenient to exclude the existence of subspace trails without making any restrictions and thus to avoid having to base the security arguments of a cipher upon heuristics.

For attackers it is interesting to see if attacks could be improved by avoiding those restrictions. As an example, the subspace trail used in [Gra+17] does not make use of any specific properties of the AES S-box. Thus, one important question raised is if those results on AES could actually be improved by taking the specific structure of the AES S-box into account.

[Gra+17] Grassi et al., "A New Structural-Differential Property of 5-Round AES"

▸ Our Contribution

In this work we rigorously analyse subspace trails for SPN ciphers. As a result of our considerations, we provide efficient and generic algorithms that can be applied to any SPN cipher and compute the longest subspace trails without any heuristics or restrictions. Thus, we are able to give a solution to Problem 5 (Algorithmic Resistance) for subspace trails.

As a first step in Section 6.2, we recall that it is actually possible to efficiently compute the entire subspace trail for any number of rounds efficiently, *given the starting subspace* U_0. Thus, the task of finding the best solution actually boils down to choosing the best starting spaces U_0. However, a priori there is a huge choice of possible starting spaces and it is clearly inefficient to simply try all possible U_0. We thus have to exhibit a way to reduce the choice of U_0 to a suitable number in such a way that we are *guaranteed* to not exclude the best choices. Our consideration here show an interesting difference depending on whether the S-box used in the cipher has linear structures or not.[1] Makarim and Tezcan [MT14] already observed an influence of linear structures on truncated differentials, but restricted themselves to linear structures in the coordinate functions of S-boxes. By considering all linear structures of an S-box, we are able to fully understand its influence on subspace trails.

More precisely, if the S-box used in the cipher does not have any linear structures (see Section 6.4.2), we can prove that the approach sketched above, that is to ignore the details of the S-box and to only consider the case where U_0 activates a single S-box, always results in the strongest subspace trail. More technically, we show that in this case for any subspace trail the subspaces U_i are without loss of generality direct products of subspaces that are aligned with the S-box layer. Note that the AES S-box does not have any linear structures. In particular, this shows that an attacker cannot hope to improve the work of [Gra+17] by taking the details of the AES S-box into account.

In the case when the S-box actually does have linear structures (see Section 6.4.3), the situation is slightly more complicated and the choice for U_0 remains huge. However, we show that simply switching from the input subspace of the first S-box layer to the output subspace of the first S-box layer results in a simple and efficient workaround. Here, we can prove that it is sufficient to consider a very limited choice of at most $k2^n$ choices for the output subspace of the first S-box layer, where k is the number of parallel S-boxes and n is the input size of a single S-box. The price to pay for this switch is that it is in general unclear if any of those trails can actually be extended backwards through the first S-box layer. However, using this approach we are able to provably bound the longest subspace trail. More precisely, our bound is either tight (in the case where we can actually extend trails backwards) or off by at most one round (in the case where we cannot).

All our algorithms are efficient for any concrete instance of a block cipher we are aware of. We run the algorithms on a number of ciphers and report on the results in the respective parts of Sections 6.4.2 and 6.4.3. Note that while our results will most likely not lead to new attacks on the ciphers we have been investigating, the main point is that we can now provably exclude the existence of such attacks.

We implemented all algorithms in C and SageMath; the SageMath the source code is listed in [Lea+18, Appendix A].

▸ Related Work

Besides the subspace trail on AES [Gra+17], subspace trail cryptanalysis has been applied to Prince in [GR16]. A paper that is technically related, but focuses on invariant subspaces instead of subspace trails is [LR17b].

Grassi et al. [Gra+16] noted the strong connection between subspace trails and truncated differentials with probability one. Actually subspace trails are a special case of the latter. Truncated differentials are commonly used for impossible differential cryptanalysis, developed by Knudsen [Knu98] and Biham et al.; Biham et al. [Bih+99a; Bih+99b]. Indeed, truncated differentials with probability one, and thus subspace trails as a special case,

[1] We like to note that both situations, that is S-boxes with and without linear structures occur in actual cipher designs.

[MT14] Makarim and Tezcan "Relating Undisturbed Bits to Other Properties of Substitution Boxes"

[Lea+18] Leander et al., "Searching for Subspace Trails and Truncated Differentials"

[GR16] Grassi and Rechberger, "Practical Low Data-Complexity Subspace-Trail Cryptanalysis of Round-Reduced PRINCE"

[LR17b] Liu and Rijmen, *New Observations on Invariant Subspace Attack*

[Gra+16] Grassi et al. "Subspace Trail Cryptanalysis and its Applications to AES"

[Knu98] Knudsen *DEAL - A 128-bit Block Cipher* can be used to construct impossible differentials, e. g. by looking for two trails that miss in the middle.

Several automatic tools were proposed for finding impossible differentials. However, none of them is able to *provably* find the best impossible differential efficiently.

The majority of these do not consider S-box details [Kim+03; Luo+09; WW12; Luo+14; Sun+16; Cui+17]. Only few attempts were done to understand the influence of the S-box layer. Wang and Jin [WJ17] improve on [Sun+16] by analysing this influence in the case of the AES. An attempt to partially cover the S-box influence is the notion of undisturbed bits, developed in the context of probability one truncated differentials [Tez14]. Makarim and Tezcan [MT14] started to describe undisturbed bits with linear structures of coordinate functions of S-boxes. Exploiting these bits results in the best known impossible differentials for some block ciphers [Tez14; Tez+14; Tez16].

Derbez and Fouque [DF16] and Sasaki and Todo [ST17] tackle the influence of an S-box without restricting to a special one. The first develop a generic algorithm working on a system of equations which describes the algorithm under scrutiny. While this in principle allows to handle a very large class of block ciphers, it comes at the cost of an increased runtime. To solve this problematic long runtime, the authors revert to handling (parts of) the S-box as a black box. Moreover, the truncated differentials considered are restricted to the case where state bits may be active or passive, while more general subspaces are not handled. The second use a different approach, namely mixed integer linear programming (MILP). Due to size constraints, the MILP model is not able to handle 8-bit S-boxes and also block sizes of 256 or 512 bits seem to be out of reach with this technique. Additionally, the authors did not consider all possible starting differences, but focused on activating only single S-boxes.

[WJ17] Wang and Jin "Upper bound of the length of truncated impossible differentials for AES"

[Sun+16] Sun et al., "Provable Security Evaluation of Structures Against Impossible Differential and Zero Correlation Linear Cryptanalysis"

[Tez14] Tezcan, "Improbable differential attacks on Present using undisturbed bits"

[MT14] Makarim and Tezcan "Relating Undisturbed Bits to Other Properties of Substitution Boxes"

[DF16] Derbez and Fouque "Automatic Search of Meet-in-the-Middle and Impossible Differential Attacks"

[ST17] Sasaki and Todo "New Impossible Differential Search Tool from Design and Cryptanalysis Aspects - Revealing Structural Properties of Several Ciphers"

6.2 SUBSPACE TRAILS AND TRUNCATED DIFFERENTIALS

Here, we continue our discussion started in Section 2.3.6. Based on Definition 49, some trivial subspace trails can be observed. In order to exclude these from our further investigation, we define essential subspace trails. Furthermore we recapitulate truncated differentials and then show that subspace trails are actually a special case of them.

6.2.1 *Subspace Trails*

We can identify some trivial subspace trails:

- $U = \{0\}$, $V = \{0\}$
- pick any $U \subseteq \mathbb{F}_2^n$, $V = \mathbb{F}_2^n$

Besides these, if $U \rightrightarrows^{F} V$, then for all $U' \subseteq U$ and $V' \supseteq V$ we have $U' \rightrightarrows^{F} V'$. The intuition here is that decreasing U will never result in a bigger V and increasing V does of course also not change the possible output differences in the trail. This leads to the following definition.

Definition 125 (Essential Subspace Trail)**.** Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and $U \subseteq \mathbb{F}_2^n$, $V \subseteq \mathbb{F}_2^m$. If $U \rightrightarrows^F V$ forms a subspace trail, i. e. $F(U + a) \subseteq V + b$, and if for all subspaces U' and V' of $\mathbb{F}_2^n$ the two properties (6.1) and (6.2) hold, we call $U \rightrightarrows^F V$ an *essential subspace trail*:

$$\forall U' \supset U : U' \overset{F}{\not\rightrightarrows} V \tag{6.1}$$

$$\forall V' \subset V : U \overset{F}{\not\rightrightarrows} V' \tag{6.2}$$

The two properties above ensure that we cannot increase U, nor decrease V without destroying the subspace trail property. Essential subspace trails are clearly the most interesting ones from an attacker perspective. Another important observation is the following.

Corollary 126. *Let $U_1 \rightrightarrows^F U_2$ be a subspace trail through F and $V_1 \subseteq U_1$ a subspace contained in U_1. Then there exists a subspace $V_2 \subseteq U_2$, such that $V_1 \rightrightarrows^F V_2$ is also a subspace trail:*

$$\begin{array}{ccc} U_1 & \overset{F}{\rightrightarrows} & U_2 \\ \cup\!\shortmid & & \cup\!\shortmid \\ V_1 & \overset{F}{\rightrightarrows} & V_2 \end{array} \tag{6.3}$$

In other words, it is actually enough to consider one dimensional starting subspaces only, when trying to identify the longest possible subspace trail. That is, the effect of reducing the initial dimension of the starting subspace can only cause longer subspace trails, not shorter ones. Thus when we are using this to bound the subspace trail lengths we are potentially only overestimating the length. However, as even in this case the number of starting spaces to consider grows exponentially with the block size, this is still clearly unfeasible for most common block sizes.

We next elaborate on the relation between subspace trails and truncated differentials with probability one.

6.2.2 *Truncated Differentials*

The following lemma is the key observation that links subspace trails to truncated differentials.

Lemma 127. *Given the subspace trail $U \rightrightarrows^F V$. Then*

$$\forall u \in U : \operatorname{Im} \Delta_u(F) \subseteq V.$$

Moreover, for any subspace $U \subset \mathbb{F}_2^n$ it holds that

$$U \overset{F}{\rightrightarrows} \operatorname{Span}\left\{ \bigcup_{u \in U} \operatorname{Im} \Delta_u(F) \right\}$$

Figure 6.1 depicts the intuition of this lemma.

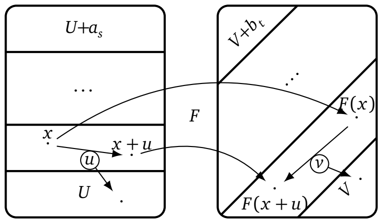


FIGURE 6.1: Graphical representation of Lemma 127.

Proof. Let $u \in U$. Because $U \rightrightarrows^F V$ is a subspace trail for F, for any $x \in \mathbb{F}_2^n$: both, x and $x + u$, are in a coset $U + x$ of U. Due to the subspace trail, they get mapped to a coset of V: $F(x), F(x + u) \in V + b$. Therefore their sum is again in V: $F(x) + F(x + u) \in V$.

Lemma 127 also confirms the intuition that subspace trail attacks and truncated differentials with probability one are closely related. Grassi et al. [Gra+16] already discussed this, but we want to state this explicitly:

[Gra+16] Grassi et al. “Subspace Trail Cryptanalysis and its Applications to AES”

Corollary 128 (Link between Subspace Trails and Truncated Differentials)**.** *Given $U \rightrightarrows^{F} V$. Then U and V determine a truncated differential with* linear *subspaces that holds with probability one: $U+0 \rightrightarrows^{F} V+0$.*

Thus, while subspace trails are included in truncated differentials (as linear subspaces are a special case of affine subspaces), the converse is not true in general. In other words, using truncated differentials we obtain a bit more information on the actual structure of the investigated function. This is depicted in the following example.

[Bog+07] Bogdanov et al., “PRESENT: An Ultra-Lightweight Block Cipher”

Example 129. We choose a single PRESENT S-box, see [Bog+07], as F and compute the subspace trail, resp. truncated differential, in terms of linear and affine subspaces for the starting difference `0x1`:

$$\{\texttt{0x0}, \texttt{0x1}\} \overset{F}{\rightrightarrows} \{\texttt{0x0}, \texttt{0x3}, \texttt{0x4}, \texttt{0x7}, \texttt{0x9}, \texttt{0xa}, \texttt{0xd}, \texttt{0xe}\}$$

$$\{\texttt{0x0}\}+\texttt{0x1} \overset{F}{\rightrightarrows} \{\texttt{0x0}, \texttt{0x4}, \texttt{0xa}, \texttt{0xe}\}+\texttt{0x9}$$

The affine subspaces we obtain are one dimension smaller than the linear subspaces.

Therefore truncated differentials are rather a generalisation of subspace trails than vice versa. For the remainder of this work, we focus on the search for subspace trails.

6.3 ALGORITHMIC PROPAGATION OF DIFFERENCES

A starting point for finding subspace trails is: Given an initial subspace, how to compute the resulting trail? One approach is based on Lemma 127. In order to compute V, we have to compute the images of the derivatives of F in direction U. To speed this up, we can exploit two facts. First, when choosing $x \in_{\mathrm{R}} \mathbb{F}_2^n$, assuming a random behaviour, it is sufficient to take slightly more than n many x to compute the subspace spanned by the image. Second, we do not need to compute the image of every element in U; instead it is enough to take a basis of U, see the following lemma.

Lemma 130. *Given $U \subseteq \mathbb{F}_2^n$ and a basis* $\mathrm{Basis}(U) = \{b_1, \ldots, b_k\}$ *of U, then*

$$\mathrm{Span}\left\{\bigcup_{u \in U} \mathrm{Im}\,\Delta_u(F)\right\} = \mathrm{Span}\left\{\bigcup_{1 \leqslant i \leqslant k} \mathrm{Im}\,\Delta_{b_i}(F)\right\}.$$

Proof. It is clear that the set on the right side is a subset of the set on the left side of the equation. Thus, we are left with showing that the left side is a subset of the right side. Moreover, as we consider the linear span on both sides, it suffices to show that any

$$v \in \bigcup_{u \in U} \mathrm{Im}\,\Delta_u(F)$$

is contained in

$$\operatorname{Span}\left\{\bigcup_{1\leqslant i\leqslant k} \operatorname{Im}\Delta_{b_i}(F)\right\}.$$

We will prove this by induction over the dimension of U. The case $\dim U = 1$ is trivial. Now assume that the statement is correct for any U' of dimension smaller than k.

We consider

$$v \in \operatorname{Im}\Delta_u(F)$$

for $u \in U$. That is, there exists an element $x \in \mathbb{F}_2^n$ such that

$$v = \Delta_u(F)(x) = F(x) + F(x+u)\,.$$

As the b_i form a basis of U, we can express u as a linear combination of the b_i, that is

$$u = \sum_{i=1}^{k} \lambda_i b_i$$

for suitable $\lambda_i \in \mathbb{F}_2$. Thus

$$v = F(x) + F(x + \sum_{i=1}^{k} \lambda_i b_i)\,.$$

By defining $x' = x + \lambda_k b_k$ we get

$$\begin{aligned} v &= F(x) + F(x + \sum_{i=1}^{k} \lambda_i b_i) \\ &= F(x' + \lambda_k b_k) + F(x' + \sum_{i=1}^{k-1} \lambda_i b_i) \\ &= F(x' + \lambda_k b_k) + F(x') + F(x') + F(x' + \sum_{i=1}^{k-1} \lambda_i b_i) \\ &= \lambda_k \Delta_{b_k}(F)(x') + \lambda' \Delta_{u'}(F)(x') \end{aligned}$$

where

$$\lambda' = \bigvee_{i=1}^{k-1} \lambda_i, \qquad u' = \sum_{i=1}^{k-1} \lambda_i b_i\,.$$

Thus

$$v \in \operatorname{Span}\left\{\operatorname{Im}\Delta_{b_k}(F) \cup \operatorname{Im}\Delta_{u'}(F)\right\},$$

and the lemma follows by induction as u' is contained in a $(k-1)$ dimensional subspace

$$U' = \operatorname{Span}\{b_1, \ldots, b_{k-1}\}\,.$$

Assembling the above observations, we get the recursive Algorithm 1 to compute the optimal subspace trail for a given starting subspace U. In Lines 6 and 7 we are sampling random elements of the subspace V. To get the full subspace we are looking for, when computing the span, we have to test enough random x. Assuming V is a random subspace of $\mathbb{F}_2^n$ and upper bounding its dimension by n, we are interested in the probability that m random vectors of length n form a matrix with full rank n. The probability

Algorithm 1 Computation of subspace trails

Precondition: A nonlinear function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, a subspace U.
Postcondition: A subspace trail $U \rightrightarrows^F \cdots \rightrightarrows^F V$.

```
function COMPUTE TRAIL($F$, $U$)
    if $\dim U = n$ then
        return $U$
    $V \leftarrow \emptyset$
    for $u_i$ basis vectors of $U$ do
        for enough $x \in_R \mathbb{F}_2^n$ do
            $V \leftarrow V \cup \Delta_{u_i}(F)(x)$
    $V \leftarrow \mathrm{Span}\{V\}$
    return $U \rightrightarrows^F$ COMPUTE TRAIL($F, V$)
```

for the i-th vector to be linearly independent of the previous $i-1$ vectors is $1-2^{i-1-m}$. Thus the probability for all m vectors to be linearly independent is $\prod_{i=0}^{n-1}(1-2^{i-m})$. This is also known as $(2^{-m};2)_n$, the 2-Pochhammer symbol or 2-shifted factorial. Computing $(2^{-m};2)_n$ shows that it is "enough" to sample e. g. $m = n + 100$ random x to compute the full subspace V with overwhelming probability.[2]

Note that in the case where F has algebraic degree two, its derivative $\Delta_{u_i}(F)$ is linear in all possible directions u_i and thus we can compute V in Algorithm 1 deterministically. In this case, similar to the linear layer case below, V has to contain the image of every derivative. Applying this optimisation yields Algorithm 2.

Algorithm 2 Computation of subspace trails for degree-two functions

Precondition: A nonlinear function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ with algebraic degree two, a subspace U.
Postcondition: A subspace trail $U \rightrightarrows^F \cdots \rightrightarrows^F V$.

```
function COMPUTE TRAIL DEGREE TWO($F$, $U$)
    if $\dim U = n$ then
        return $U$
    $V \leftarrow \mathrm{Span}\left\{ \bigcup_{u \in \mathrm{Basis}(U)} \mathrm{Im}\,\Delta_u(F) \right\}$
    return $U \rightrightarrows^F$ COMPUTE TRAIL DEGREE TWO($F, V$)
```

Unfortunately this still does not reduce the number of possible starting differences. Thus let us now take a more detailed look at the parts of an SPN. The influence of the linear layer on subspace trails is straightforward.

Proposition 131 (Subspace trails through linear layers)**.** *Let $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a bijective linear function. Then every $U \subseteq \mathbb{F}_2^n$ defines a subspace trail of the form $U \rightrightarrows^L L(U)$. U and $L(U)$ have the same dimension. Moreover, any essential subspace trail for L is of this form.*

[2] With $m = n + 100$ we obtain an error probability of 2^{-100}, and $m = n + 20$ results in roughly 2^{-20}.

Proof. Recall that for a subspace trail $U \rightrightarrows^{L} V$ it holds that

$$V \supseteq \bigcup_{u \in U} \operatorname{Im} \Delta_u(L).$$

For any bijective linear function

$$\operatorname{Im} \Delta_u(L) = \{L(x) + L(x+u)\} = \{L(x) + L(x) + L(u)\} = \{L(u)\}.$$

As L is bijective, for any pairwise different $u_1, u_2 \in U$, $L(u_1)$ and $L(u_2)$ are also different. Thus, $V = L(U)$ and $\dim U = \dim V$.

The S-box layer exhibits a more interesting behaviour. Here, we need to distinguish between S-boxes without non-trivial linear structures and ones with linear structures. The first case is covered in Section 6.4.2, the second in Section 6.4.3.

Before we study the S-box layer, we will next motivate why the problem of finding the best subspace trail is not as easy as it might seem.

6.4 PROVING RESISTANCE AGAINST SUBSPACE TRAIL ATTACKS

6.4.1 *Activating Only Single S-boxes*

As mentioned in the introduction, one common heuristic used in finding long truncated differentials, is to activate only a single S-box in the input difference. Due to the close relation of truncated differentials and subspace trails discussed above it seems natural to use the same heuristic for the latter. Intuitively it may seem that using starting spaces U that contain non-zero values only for a single S-box performs the best.

In this section we argue that this is not always the best approach. This is done by considering two examples. The first one is a toy example based on the PRESENT S-box, while the second one is the cryptographic permutation KECCAK-f.

Example 132. We choose again the PRESENT S-box, $F : \mathbb{F}_2^4 \to \mathbb{F}_2^4$, and a block size of 16 bits, so the S-box layer applies four S-boxes in parallel, $F^4 : \mathbb{F}_2^{16} \to \mathbb{F}_2^{16}$. The round function $R : \mathbb{F}_2^{16} \to \mathbb{F}_2^{16}$ is then $R(x) := L \cdot F^4(x)$, where as the linear layer L we choose $L = L' \cdot B^{-1}$, with L, L', and B as given in the margin.

$$L = \begin{pmatrix}
0&0&0&1&1&0&1&0&0&0&0&1&1&0&1&0\\
1&0&1&1&1&0&1&1&1&0&1&1&1&0&1&1\\
0&0&0&0&0&0&0&1&1&0&1&0&1&0&1&1\\
0&0&0&1&1&0&1&1&1&0&1&0&0&0&0&0\\
1&0&1&0&0&0&0&1&1&0&1&1&1&0&1&1\\
0&0&0&1&1&0&1&1&0&0&0&0&0&0&0&1\\
0&0&0&1&0&0&0&1&1&0&1&0&1&0&1&0\\
0&0&1&1&1&0&1&1&0&0&0&0&1&0&1&0\\
1&1&1&0&0&0&0&0&1&0&1&1&0&0&0&1\\
1&0&1&1&0&0&1&0&1&0&1&0&1&0&1&0\\
1&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&1&0&1&0&0&0&0&0&0&0\\
1&0&1&1&1&0&1&1&0&1&0&0&1&0&1&1\\
0&0&0&0&1&0&1&0&0&0&0&0&1&0&0&0\\
1&0&1&1&0&0&0&0&0&0&1&1&1&1&1&0\\
0&0&0&1&0&0&0&1&0&0&0&0&1&0&1&0
\end{pmatrix}$$

$$L' = \begin{pmatrix}
0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1\\
0&0&0&0&0&0&0&0&0&1&1&1&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&1&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1\\
0&0&0&0&0&0&0&0&0&1&1&0&0&1&1&1\\
1&0&0&0&0&0&0&0&0&1&0&0&1&1&0&1\\
0&1&0&0&0&0&0&0&0&0&0&1&0&0&1&0\\
0&0&1&0&0&0&0&0&0&1&1&1&1&1&0&0\\
0&0&0&1&0&0&0&0&0&0&1&1&0&1&1&0\\
0&0&0&0&1&0&0&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&1&0&0&0&0&1&1&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&1&1&1&0&1\\
0&0&0&0&0&0&0&1&0&1&0&1&0&1&1&0\\
0&0&0&0&0&0&0&0&1&0&0&1&1&0&1&0
\end{pmatrix}$$

$$B = \begin{pmatrix}
1&0&0&0&0&0&0&0&1&1&0&0&1&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&1&0&0&0&1\\
1&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1&1&0&1&1\\
0&0&1&0&0&0&0&0&1&1&0&1&0&1&0&0\\
0&0&0&1&0&0&0&0&0&1&0&1&1&1&0&0\\
0&0&1&0&0&0&0&0&0&1&0&1&1&0&0&0\\
0&0&0&0&0&0&0&0&1&1&0&1&0&0&1&0\\
0&0&0&0&1&0&0&0&1&1&0&0&0&1&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&1\\
0&0&0&0&1&0&0&0&0&0&1&1&1&0&0&1\\
0&0&0&0&0&0&0&0&1&0&1&1&1&0&1&1\\
0&0&0&0&0&0&1&0&1&0&0&1&0&0&0&1\\
0&0&0&0&0&0&0&1&0&0&1&1&0&1&0&0\\
0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&0&0&0&0&0&1&0&1&1&0&1&0&0
\end{pmatrix}$$

Let us first observe which subspace trails this round function exhibits. Almost all subspace trails starting with only one active S-box are of the form

$$\dim 1 \rightrightarrows \dim 4 \rightrightarrows \dim 16.$$

Thus reaching the full dimension after two rounds. In contrast to this, the subspace trail with a 0x1 difference on every S-box input, more precisely the starting subspace is $\{\texttt{0x0000}, \texttt{0x1111}\}$ – which is one dimensional but activates all four S-boxes in the beginning, has the following form:

$$\texttt{0x1111}: \quad \dim 1 \rightrightarrows \dim 9 \rightrightarrows \dim 11 \rightrightarrows \dim 16, \tag{6.4}$$

effectively lasting for a full additional round. This is of course due to the special structure of the linear layer. The left part of the chosen basis B

consists of exactly the vectors spanning the space of dimension nine after the first round. In combination with the given L', all these vectors are then mapped to only activating the rightmost three S-boxes (with a `0x1` difference on the third S-box), leaving the last S-box passive.

Still, there are four subspace trails, with only one active S-box, that also reach three rounds:

$$\begin{array}{ll} \texttt{0x1000}: & \dim 1 \rightrightarrows \dim 3 \rightrightarrows \dim 14 \rightrightarrows \dim 16 \\ \texttt{0x0001}: & \dim 1 \rightrightarrows \dim 3 \rightrightarrows \dim 15 \rightrightarrows \dim 16 \\ \texttt{0x0f00}: & \dim 1 \rightrightarrows \dim 3 \rightrightarrows \dim 14 \rightrightarrows \dim 16 \\ \texttt{0x000f}: & \dim 1 \rightrightarrows \dim 3 \rightrightarrows \dim 15 \rightrightarrows \dim 16 \end{array}$$

But note that all these subspace trails are worse, as the last dimension, before reaching the full dimension, is higher than eleven. In this example it is thus not the best choice, to activate only one S-box. Also note that the resulting linear layer L looks sufficiently random, s. t. its hidden structure might be hard to find. Of course it is very unlikely that such an effect occurs, when the design of the linear layer follows e. g. the wide trail strategy.

Example 133. A real-world example of the above observation is the SHA-3 hash function, more precisely KECCAK-f [Ber+11; SHA3]. Starting with any subspace trail of dimension one that has only one active S-box gives us a three round trail with final dimension 1139 in the best case. But instead starting with an other one dimensional subspace trail with *two* active S-boxes results in a different three round trail that has dimension 949 – a huge reduction compared to the initial trail. The exact dimensions are the following:

[Ber+11] Bertoni et al., *The Keccak reference*

[SHA3], *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*

$$\begin{array}{ll} 1 \times \texttt{0x4}: & \dim 1 \rightrightarrows \dim 3 \rightrightarrows \dim 65 \rightrightarrows \dim 1139 \rightrightarrows \dim 1600 \\ 2 \times \texttt{0x4}: & \dim 1 \rightrightarrows \dim 5 \rightrightarrows \dim 49 \rightrightarrows \dim \quad 949 \rightrightarrows \dim 1600 \end{array}$$

The big advantage of looking only at possible input spaces with one active S-box is the huge reduction in starting points we need to analyse. But as we have just seen, we cannot rely on these special inputs, when we want to prove the absence of subspace trails. While we can argue with Corollary 126 that it is enough to look at trails starting with a one dimensional subspace, we are still facing the problem to cope with exponentially many possible subspace trails. In the next sections, we develop arguments and algorithms to again reduce this complexity.

6.4.2 *S-box layers without Linear Structures*

The distinction of S-boxes based on their linear structures is based on the following lemma that explains the link. Recall that $\mathrm{L}_u(F)$ is defined as the set of all α such that for a fixed $c \in \mathbb{F}_2$

$$\alpha \cdot \Delta_u(F)(x) = c \quad \forall x \in \mathbb{F}_2^n .$$

Let us start by considering a single S-box $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ only.

Lemma 134. *Given an essential subspace trail $U \rightrightarrows^F V$. Then $V^\perp = \bigcap_{u \in U} \mathrm{L}_u(F)$.*

Proof. According to Lemma 127 it holds that

$$\forall x \in \mathbb{F}_2^n \; \forall u \in U : \Delta_u(F)(x) \in V \, .$$

Now, for an arbitrary element $\alpha \in V^{\perp}$ this implies that

$$\forall x \in \mathbb{F}_2^n \; \forall u \in U : \alpha \cdot \Delta_u(F)(x) = 0 \, .$$

Thus

$$V^{\perp} = \bigcap_{u \in U} \mathrm{L}_u(F)$$

and, as the trail is essential, the statement follows.

In particular we have Corollary 135 as a direct consequence.

Corollary 135. *If F has no non-trivial linear structure, then there are only two essential subspace trails: $\{0\} \rightrightarrows \{0\}$ and $\mathbb{F}_2^n \rightrightarrows \mathbb{F}_2^n$.*

The most important observation is that, in the case of F without linear structures, a similar observation holds for the entire S-box layer consisting of k parallel S-boxes. Namely, in this case any essential subspace trail is *the direct product of subspace trails of the single S-box*. This implies with the Corollary 135 that any essential subspace trail through an S-box layer can simply be characterised by active and passive S-boxes. The following proposition covers exactly this.

Proposition 136 (Subspace trails through S-box layers)**.** *Let F^k be an S-box layer, and $U \rightrightarrows^{F^k} V$ an essential subspace trail. If F has no non-trivial linear structures, then*

$$U = V = U_1 \times \cdots \times U_k$$

where for $1 \leqslant i \leqslant k$ it holds that $U_i \in \{\{0\}, \mathbb{F}_2^n\}$.

Proof. Let $U \rightrightarrows^{F^k} V$ be essential. Then $\forall \alpha = (\alpha_1, \ldots, \alpha_k) \in V^{\perp}$ the following holds

$$\alpha \cdot \Delta_u\big(F^k\big)(x) = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_k \end{pmatrix} \cdot \begin{pmatrix} \Delta_{u_1}(F)(x_1) \\ \vdots \\ \Delta_{u_k}(F)(x_k) \end{pmatrix} = \sum_{i=1}^{k} \alpha_i \cdot \Delta_{u_i}(F)(x_i) = 0$$

for all $x_i \in \mathbb{F}_2^n$. Thus every term $\alpha_i \cdot \Delta_{u_i}(F)(x_i)$ is constant. As F has no non-trivial linear structures, this implies that either α_i or u_i is zero. But then, in both cases, if one of the two is zero, the other can take any value and still has to be in the corresponding subspace, because the trail is by assumption essential.

It is important to note that not only Corollary 135 does not hold for F with linear structures, but more fundamentally (and maybe counterintuitively) the statement that any essential subspace trail is a direct product of subspace trails of the single S-box *does not hold in the case of F with linear structures*. A particular case of this is given by the subspace trail in Example 132 (see Eq. (6.4)). We come back to this case in Section 6.4.3.

Let us give a specific notation for the particular structure of the above subspaces. If $U = U_1 \times \cdots \times U_k$ with $U_i \in \big\{\{0\}, \mathbb{F}_2^n\big\}$, we write $U = \big\{\{0\}, \mathbb{F}_2^n\big\}^u$

if and only if $U_i = \mathbb{F}_2^n \Leftrightarrow u_i = 1$ where u_i is the i-th bit of the binary depiction of u.

Now a direct consequence of the proposition is the following corollary.

Corollary 137. *If F has no non-trivial linear structures, there are only 2^k possible $U \in \left\{\{0\}, \mathbb{F}_2^n\right\}^k$, such that $\exists V \subseteq \left(\mathbb{F}_2^n\right)^k : U \rightrightarrows^{F^k} V$ is essential.*

In other words, and we would like to stress this, an S-box without non-trivial linear structures behaves like any generic S-box. Particularly the only important property of such an S-box layer is if the S-box is "activated" or not. As, e. g. the AES S-box does not have any linear structures (and neither does its inverse has any), this means that *we cannot improve* the subspace trail attacks given in [Gra+17] by taking details of the S-box into account.

[Gra+17] Grassi et al., "A New Structural-Differential Property of 5-Round AES"

With the above we can construct Algorithm 3 to compute a list of all one round subspace trails. This list is enough to construct all possible subspace trails. To iterate the rounds we just have to look up the next subspace in it. After reaching a subspace with full dimension, the actual trail cannot be iterated further.

Algorithm 3 No Non-Trivial Linear Structures

Precondition: A linear layer matrix $M : \mathbb{F}_2^{n \cdot k \times n \cdot k}$.
Postcondition: A list with all one round subspace trails.

```
function One Round Subspace Trails($M$)
    empty list $L$
    for possible initial subspaces represented by $u \in \{0,1\}^k$ do
        $U = \left\{\{0\}, \mathbb{F}_2^n\right\}^u$
        $U' = \{M \cdot u_i \mid u_i$ basis vectors of $U\}$
        $V = (V_1, \dots, V_k) = \left\{\{0\}, \mathbb{F}_2^n\right\}^v$, s. t.
            $V_i = \begin{cases} \{0\} & \text{if } U_i' = \{0\} \\ \mathbb{F}_2^n & \text{else} \end{cases}$    with respective
            $v_i = \begin{cases} 0 & \text{if } U_i' = \{0\} \\ 1 & \text{else} \end{cases}$
        append $U \rightrightarrows V$ to $L$
    return $L$
```

Results

Besides the normal round function, we also have to take the inverse into account, as an S-box and its inverse do not necessarily have the same linear structures. It may even be the case that an S-box has no non-trivial linear structures, while its inverse has some. Examples for such S-boxes are the ones from Safer [Mas94], SC2000 [Shi+02], and Fides [Bil+13]. Moreover, the inverse of the linear layer might be weaker than the linear layer itself.

[Per17] Perrin "Cryptanalysis, Reverse-Engineering and Design of Symmetric Cryptographic Algorithms"

Perrin [Per17] collected a list of S-boxes used or discussed in the literature. From this list of over 200 S-boxes, we filtered those designs that are generic SPNs and applied Algorithm 3 to them, if the corresponding S-box has no non-trivial linear structures. The resulting lengths of the longest subspace

	Algorithm 3			
Cipher	r_e	d	r_d	d
AES [DR02]	2	32	2	32
Anubis [BRa]	2	104	—[†]	—[†]
Klein [Gon+11]	3	60	2	32
Kuznyechik [GOST15]	1	8	1	8
Prince [Bor+12]	2	16	2	16
Qarma [Ava17][*]	2	36	2	36

[*] Qarma comes with three S-boxes, two of these do not exhibit linear structures. The given results are thus for Qarma instantiated with either σ_1, or σ_2.

[†] Anubis uses involutional operations, so we only give the encryption trails.

Table 6.1: Lengths of the longest subspace trails in various SPN ciphers with S-boxes that do not have any non-trivial linear structures. r_e denotes the number of encryption rounds covered, r_d the number of decryption rounds, and d the final dimension.

trails are listed in Table 6.1. To the best of our knowledge, all these trails are not better than what is already known in the literature.

Note that this algorithm works for any generic S-box that has no non-trivial linear structures. Regarding its runtime, there are 2^k possible initial subspaces. For each we have to compute the corresponding output subspace, which boils down to a matrix multiplication with every basis vector of the input subspace. Altogether this results in an overall runtime that is exponential in the number of S-boxes and linear in the dimension of the linear layer. A non-optimised C implementation takes seconds to finish the computations for the tested ciphers on a single CPU core of a standard laptop.

From a designers perspective this maintains the ability to independently choose the linear and S-box layer, with respect to subspace trails. Nevertheless, we might not want to use such a decoupled design approach, e. g. in the design process of lightweight cryptography, we often aim for vigorous optimised designs. During such an optimisation process, we might decide to trade strong security properties for more efficient parts by carefully matching the components of the whole design. A recent proposal, which does exactly this, is [Ban+17].

[Ban+17] Banik et al., “GIFT: A Small Present - Towards Reaching the Limit of Lightweight Encryption”

Thus we also have to cover the other case, involving S-boxes with non-trivial linear structures.

6.4.3 *S-box layers with Linear Structures*

As mentioned above, the core observation in the case of an S-box layer with an S-box without linear structures is that any essential subspace trail is the direct product of subspace trails for the single S-boxes. Unfortunately, this does not hold when moving to S-boxes that actually do have linear structures. An example of this has been presented in Section 6.4.2.

However, without a significant reduction of the possible starting subspaces that have to be considered, we do not have the possibility to guarantee the non-existence of long subspace trails. To overcome this problem, we move from considering the input subspace to the first S-box layer to the output subspace of the first S-box layer. The key point is that even though we cannot restrict the input subspaces in a meaningful way, we can actually

$$\begin{tikzcd}
U_1 \arrow[d, "S"'] \arrow[dr] & \\
U_1' \arrow[r, phantom, "\supseteq"] \arrow[d, "L"'] & W_1 = \{W\} \arrow[d, "L"'] \\
U_2 \arrow[d, "S"'] & W_2 \arrow[d, "S"'] \\
\vdots \arrow[d, "L"'] & \vdots \arrow[d, "L"'] \\
U_k & W_\ell
\end{tikzcd}$$

Figure 6.2: How Corollary 126 is used to upper bound the length of subspace trails.

do so for the output subspaces. Namely, we will show below that the output space of the first S-box layer always contains a subspace that (again) activates only one S-box. Figure 6.2 depicts this approach.

More precisely, as it is too expensive to check the trail length of every possible U_1, we look at the output spaces U_1' after the first S-box layer. Applying Corollary 126 results still in a subspace trail starting from $W_1 \subseteq U_1'$, and we additionally restrict ourselves to one dimensional W_1. Eventually, we use the length ℓ of the subspace trail $W_1 \rightrightarrows W_\ell$ to provably bound the length k of the longest subspace trail. Note that it stays uncertain at this point if any of those trails can actually be extended backwards through the first S-box layer, thus $k = \ell$ if the trail cannot be backward extended, or $k = \ell + 1$ else. However, using this approach we are able to provably bound the longest subspace trail. Moreover our bound is either tight or off by at most one round.

The remaining problems are: What is the problem with backward extension? Why does the length of $W_1 \rightrightarrows W_\ell$ actually gives a meaningful bound on the longest subspace trail? And third, we have to find a (small) set $\mathbb{W}$ of all possible W_1, such that *every possible* U_1 contains at least one element from it after the first S-box layer.

▸ Backward extension

Admittedly it is easy, given any W_1, to find an U_1, such that U_1 is a starting point for a subspace trail that contains W_1 after the S-box layer. But the real problem is that we want to backward extend W_1 in such a way that both trails continue with the same dimensions. This can either happen if $U_1 \rightrightarrows W_1$ is essential and $\dim U_1 = \dim W_1 = 1$, but then U_1 would be a probability one differential characteristic for the S-box layer, or if $W_1 \rightrightarrows W_2$ is not essential. In the latter case, it is not clear how to extend W_1 backwards, as we are missing the information what else U_1' contains and is missing in W_1 to make the trail essential.

▸ Bounding the length

Assume the longest subspace trail starting from an element in $\mathbb{W}$ covers ℓ rounds. From Figure 6.2 it is clear that $\ell \geqslant k$, and we can either backward extend W_1 or we cannot. If $\ell = k$ and backward extension is possible, our bound ℓ is off by one round compared to the actual longest subspace trail. On the other side, if we cannot backward extend the trail, our bound is tight. If $\ell > k$ (this is possible, because W_1 is a subset of U_1' and thus might cover more rounds), we effectively found a longer trail, we can directly use this as the longest subspace trail, and our bound is tight.

▸ The set $\mathbb{W}$

For $\mathbb{W}$, we look at all possible ways to activate one S-box after the first S-box layer. As the choices for activating one S-box are very limited and typically the number of S-boxes is small compared to the overall block size, $\mathbb{W}$ is also small compared to all possible starting points. Thus we need to show that $\mathbb{W}$ is an appropriate set in the sense of our above requirement – that is, for every possible U_1 there is one element W, such that $W \in U_1'$ after the first S-box layer. Before stating the main proposition, we need the following lemma.

Lemma 138. *Let F be an S-box with differential uniformity $\delta_F < 2^n$, i. e. for all $\alpha, \beta \in \mathbb{F}_2^n$, the cardinality of $\left\{x \in \mathbb{F}_2^n \;\middle|\; F(x) + F(x+\alpha) = \beta\right\}$ is smaller than 2^n. Then*

$$\forall u \in \mathbb{F}_2^n \setminus \{0\} : \mathrm{L}_u(F) \neq \mathbb{F}_2^n.$$

Proof. Assume $\delta_F < 2^n$ and there is an u, such that $\mathrm{L}_u(F) = \mathbb{F}_2^n$. Let $\{e_i\}$ be the canonical basis for $\mathrm{L}_u(F)$. For every e_i the following holds:

$$e_i \cdot \Delta_u(F)(x) = c_{e_i} \qquad \forall x \in \mathbb{F}_2^n, c_{e_i} \in \mathbb{F}_2$$

But then the differential characteristic $u = (u_1, \ldots, u_n)$ to $v = (c_{e_1}, \ldots, c_{e_n})$ holds with probability one and thus $\delta_F = 2^n$, which is a contradiction.

We construct the set $\mathbb{W}$ of subspaces as follows:

$$\mathbb{W} := \left\{W_{i,\alpha} := \{0\}^{i-1} \times \{0, \alpha\} \times \{0\}^{k-i} \;\middle|\; \alpha \in \mathbb{F}_2^n, 1 \leqslant i \leqslant k\right\}.$$

The following proposition shows that any output subspace after the S-box layer actually has to contain at least one element $W \in \mathbb{W}$.

Proposition 139. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an S-box with differential uniformity $\delta_F < 2^n$, and $U \rightrightarrows^{F^k} V$ a non-trivial subspace trail. Then there exists a $W \in \mathbb{W}$ such that*

$$W \subseteq V$$

Proof. As U is non-trivial, it contains at least one non-zero element $u = (u_1, \ldots, u_k)$. For an arbitrary element $\beta \in V^\perp$ and all $x \in \mathbb{F}_2^n$ we get

$$\beta \cdot \Delta_u(F)(x) = 0$$

which implies that $\beta \in L_u^0(F^k)$ and thus

$$V^\perp \subseteq L_u^0(F^k)$$

Furthermore,

$$\mathrm{L}_u^0(F^k) \subseteq \mathrm{L}_{u_1}(F) \times \mathrm{L}_{u_2}(F) \times \cdots \times \mathrm{L}_{u_k}(F).$$

As u is non-zero, at least one of its components u_i is non-zero. According to Lemma 138, $\mathrm{L}_{u_i}(F) \neq \mathbb{F}_2^n$, which implies that there exists an $\alpha \in \mathbb{F}_2^n$ such that

$$\mathrm{L}_{u_i} \subseteq \{0, \alpha\}^\perp$$

Putting things together we conclude that

$$V^\perp \subseteq L_u^0(F^k) \subseteq \left(\mathbb{F}_2^n\right)^{i-1} \times \{0, \alpha\}^\perp \times \left(\mathbb{F}_2^n\right)^{k-i}.$$

By looking at the orthogonal complement of both sides we finally get

$$\left(\left(\mathbb{F}_2^n\right)^{i-1} \times \{0, \alpha\}^\perp \times \left(\mathbb{F}_2^n\right)^{k-i}\right)^\perp = \{0\}^{i-1} \times \{0, \alpha\} \times \{0\}^{k-i} \subseteq V,$$

which concludes the proof.

Algorithm 4 Generic Subspace Trail Search

Precondition: A linear layer matrix $M : \mathbb{F}_2^{n\cdot k \times n\cdot k}$, an S-box $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$.
Postcondition: An upper bound on the length of any subspace trail.

```
function GENERIC SUBSPACE TRAIL LENGTH($M$, $F$)
    empty list $L$
    for possible initial subspaces represented by $W_{i,\alpha} \in \mathbb{W}$ do
        $L$.append(COMPUTE TRAIL($F^k \circ M$, ($\{0\}$, $W_{i,\alpha}$)))
    return max $\{\text{len}(t) \mid t \in L\}$
```

TABLE 6.2: Lengths of the longest subspace trails in various SPN ciphers with 1-linear structures. r_e denotes the number of rounds covered for encryption, r_d the number of rounds covered for decryption, and d the final dimension. Algorithms 3 and 4 are compared, where the algorithm taking the influence of linear structures into account (Algorithm 4) finds better results, that is longer trails and/or lower dimensions.

	Algorithm 4				Algorithm 3			
Cipher	r_e	d	r_d	d	r_e	d	r_d	d
ASCON [Dob+16]	3	298	1	125	3	310	1	155
CLYDE [Bel+19]	2	44	2	41	2	68	2	64
GIFT [Ban+17]	3	60	3	60	2	16	2	16
KECCAK [Ber+11]	2	546	1	169	2	1290	1	270
PRESENT [Bog+07]	3	43	3	63	2	16	2	16
PRIDE [Alb+14]	2	31	2	34	2	56	1	40
QARMA [Ava17]*	2	36	2	36	2	36	2	36
SERPENT [Bih+98]	2	88	2	62	2	100	2	68
SHADOW [Bel+19]	4	384			—†	—†	—†	—†
SKINNY64 [Bei+16b]	5	48	5	48	4	48	4	48
SKINNY128 [Bei+16b]	5	96	5	96	5	96	5	96

* The given results are for QARMA instantiated with σ_0.
† Computational infeasible

We can now build Algorithm 4, which simply applies the algorithm in Section 6.3 to every element in $\mathbb{W}$ to complete the corresponding trail. Because $|\mathbb{W}|$ is small and Algorithm 1 is efficient, this will allow us to compute the longest subspace trail that starts after the first layer of S-boxes, and its trail bounds the longest possible subspace trail.

▸ RESULTS

We again filtered the list of S-boxes by Perrin [Per17] for SPN constructions and S-boxes that exhibit 1-linear structures. The runtime of Algorithm 4 is similar to the one of the previous algorithm. Here, we have to compute trails for $|\mathbb{W}| = k \cdot 2^n$ many initial subspaces. Computing trails for a given U_0 is dominated by Gaussian eliminations for the basis reductions of the corresponding subspaces. Thus, the runtime mainly depends on the block size, length of trails and number and size of the S-box. Although our reference C implementation has room for optimisations, it takes only a few seconds to find the trails of 64-bit block ciphers and a few minutes to find the trails of 128-bit block ciphers using a single CPU core on a standard laptop.

[Per17] Perrin "Cryptanalysis, Reverse-Engineering and Design of Symmetric Cryptographic Algorithms"

In Table 6.2, we upper bound the length of subspace trails for several

SPN ciphers. To the best of our knowledge, no subspace trails for any of these ciphers is known in the literature at the time of writing. We give the resulting trail lengths of Algorithm 4 with the corresponding dimensions of the final subspace. Note that for a subspace trail of length r the actual upper bound is $r+1$. But as already mentioned above, we might be able to backward extend the trail for one more round.

One example, where exactly this happens, is Skinny128. The best impossible differential known is given in [Bei+16b, Section 4.3] and exploits a six round differential in the encryption direction that misses another five round differential in decryption direction. Algorithm 4 finds a five round subspace trail, thus in this case the known differential reaches the $5+1$ bound.

[Bei+16b] Beierle et al., "The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS"

As a comparison to Algorithm 3, we also give the resulting longest subspace trails for the case of an S-box without linear structures in the second part of Table 6.2. Here again Skinny64 is a nice example, because it highlights the influence of the linear structures in its S-box. The bound of four rounds by Algorithm 3 is too low compared to the actual best trail.

This concludes our security argument against subspace trail attacks.

6.5 OPEN PROBLEMS

Before discussing a second application of our ComputeTrail algorithm, let us discuss some open problems related to subspace trails.

There are basically two directions to generalise our results. First, it would be interesting to not only focus on SPN ciphers but also cover Feistel structures. A technically related problem to the generalisation is the following: For a subspace trail $U \rightrightarrows^{F^k} V$ the joint space of inputs and outputs $U \times V \subseteq \mathbb{F}_2^{2n}$ is (obviously) a direct product of subspaces. A question that appears when studying Feistel structures, is if there exists a meaningful generalisation where one considers more general subspaces of $\mathbb{F}_2^{2n}$ for the joint input and output spaces. An answer to this question could not only be useful for the case of Feistel ciphers, but be of independent interest as well. However, although our results for SPN ciphers do not directly apply to Feistel ciphers, our algorithms can be used to obtain probability one subspace trails for them. We observe that the longest subspace trails our algorithms can find are compatible with the longest probability one truncated differentials that are used in the literature for obtaining impossible differentials via the miss-in-the-middle technique. Moreover, since we take into account the S-box properties and we focus on the dimensions instead of non-indeterminate bit differences, the dimensions we find are most of the time better than the ones used in the literature. These reductions in the dimensions can provide more freedom to the attacker and may lead to better attacks. We provide the longest subspace trails that our search algorithms find in Table 6.3.

To obtain our results, we search for the longest subspace trail starting with a single active S-box only. For ciphers like Simon that do not use an S-box, we consider differences in one word only. Although we obtain a 13-round subspace trail for Skipjack, we cannot use it to extend the 24-round impossible differential of [Bih+99a; Bih+05] to 25 or 26 rounds, because Skipjack does not use identical round functions for all rounds. The

Table 6.3: Lengths of the longest truncated differentials used in the best known impossible differentials in various Feistel ciphers. r denotes the number of rounds covered by the differential, and d the dimension at the miss-in-the-middle-point. Our results are obtained via subspace trails.

	Literature			Our Results	
Cipher	r	d	Source	r	d
Camellia	4	128[†]	[BL12]	4	105
Clefia	5	128[†]	[Tsu+08]	5	112
LBlock	7	—[‡]	[WZ11]	7	58
Piccolo	4	64[†]	[WW12]	3	48
Simon32/64	6	29	[DF16]	6	30[*]
Simon64/128	8	—[‡]	[Bou+14]	8	62
Simon128/256	12	—[‡]	[Bou+14]	12	126
Skipjack	12	48	[Bih+99a; Bih+05]	13	48

[*] Moving to affine subspaces reduces this dimension to 29. Thus, this is an another example to show the differences between subspace trails and truncated differentials.
[†] The contradiction for the miss-in-the-middle is caused by exploiting special properties of the round function.
[‡] The truncated differential is not explicitly given in the literature, so we do not know its exact dimension.

limitations of our algorithms should also be considered when constructing impossible differentials using probability one subspace trails:

- A subspace trail with full dimension may still be useful for constructing impossible differentials via the miss-in-the-middle technique. Even if it reaches full dimension, it may still contain exploitable information like, e. g. an S-box output difference being nonzero.
- Unlike SPN, for Feistel ciphers it may be possible to obtain $(r_e + r_d + 1)$-round impossible differentials (instead of $r_e + r_d$) by using properties of the round function.
- This direct application of our search algorithms to Feistel ciphers of course do not prove the resistance against subspace trail attacks.

The second way to generalise our results is to consider truncated differential trails with probability one instead of subspace trails. As mentioned in Section 6.2, the only difference is that one moves from subspaces to affine subspaces. Indeed, in the case of S-boxes without linear structures, our results are directly applicable to truncated differential trails as well. However, in the case of S-boxes with linear structures, the situation is quite different. Here, the main task is to find a suitable reduction in the possible offsets of the affine spaces used, which seems non-trivial.

Finally, we like to mention two points related to invariant subspaces (see e. g. [Lea+15]). First, they behave very differently, compared to subspace trails. It is actually not trivial to understand the possible invariant subspaces of an S-box layer, even in the case of an S-box without linear structures (see [LR17b] for interesting partial results along those lines). In particular, it is not the case that an invariant subspace for an S-box layer is always the direct product of invariant subspaces of the single S-boxes. An easy counterexample is the following: Consider two parallel application of an

[Lea+15] Leander et al., “A Generic Approach to Invariant Subspace Attacks: Cryptanalysis of Robin, iSCREAM and Zorro”

[LR17b] Liu and Rijmen, *New Observations on Invariant Subspace Attack*

S-box. If the two inputs are identical, so are the outputs. That is, the space

$$U = \{x, x\}$$

is an invariant subspace for any S-box and is clearly not a direct product. Hence, an efficient classification of all invariant subspaces for an S-box layer is a non-trivial but interesting open problem as well.

And second, a generalisation of subspace trails, to fit the notion of invariant subspaces, might be of interest. This is based on the preprint

> Lorenzo Grassi, Gregor Leander, Christian Rechberger, Cihangir Tezcan, and Friedrich Wiemer. *Weak-Key Subspace Trails and Applications to AES*. 2019. IACR PREPRINT: 2019/852.

The underlying idea becomes clearer, when comparing subspace trail and invariant subspace attacks. Two obvious but important differences can be observed. First, subspace trails are more general as they allow different spaces in the domain and co-domain of the round function F. But second, subspace trails are also more restrictive, as not only one coset of the subspace has to be mapped to one coset of (a potentially different) subspace, but rather all cosets have to be mapped to cosets. The later fact is the main reason that subspace trails work with arbitrary round keys. One possible generalisation of subspace trails, to capture this observation, is the following.

The main idea for so called *weak-key subspace trails* is to stick to the property of invariant subspace attacks where only few (or even just one) cosets of a subspace are mapped to other cosets of a subspace. Here, invariant subspace are defined as

Definition 140 (Invariant subspace trail). Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be the round function of a key-alternating cipher, denote by $F_k(x) = F(x) + k$ the round including the addition of the round key, and let $K \subseteq \left(\mathbb{F}_2^n\right)^r$ be a set of keys, where for $k \in K$ we denote by k_i the i-th round key.

For a subspace $U \subset \mathbb{F}_2^n$, its cosets $U{+}a_i$ form an *invariant subspace trail* (of length r) through F, denoted $U{+}a_1 \mapsto \cdots \mapsto U{+}a_r$, if for all $k \in K$

$$\forall 1 \leqslant i < r : \quad F_{k_i}(U{+}a_i) = F(U{+}a_i) + k_i = U{+}a_{i+1} \ .$$

The keys in K are called *weak keys*.

However, borrowing from subspace trails, we allow those subspaces to be different for each round. As this will again restrict the choice of round keys that will keep this property invariant to a class of weak keys we call this combination *weak-key subspace trails*.[3] The formal definition is the following.

Definition 141 (Weak-key subspace trails). Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be the round function of a key-alternating cipher, denote by $F_k(x) = F(x) + k$ the round including the addition of the round key, and let $K \subseteq \left(\mathbb{F}_2^n\right)^r$ be a set of keys, where for $k \in K$ we denote by k_i the i-th round key.

Further let $(U_0, U_1, \ldots, U_r)$ denote a set of $r + 1$ subspaces, $U_i \subseteq \mathbb{F}_2^n$, with $\dim U_i \leqslant \dim U_{i+1}$. We call $(U_1, U_2, \ldots, U_{r+1})$ a *weak-key subspace trail*

[3] In fact one could argue that this is what subspace trails should have been named in the first place, as a generalisation of invariant subspaces.

(WKST) of length r through F if for each $1 \leqslant i < r$ there exists a non-empty set $A_i \subseteq U_i^c$ for which the following property holds:

$$\forall a_i \in A_i : \quad \exists! a_{i+1} \in A_{i+1} : \quad F_{k_i}(U_i + a_i) = F(U_i + a_i) + k_i \subseteq U_{i+1} + a_{i+1} \,.$$

The keys in K are called *weak keys*.

Usually, the set $A_i \subseteq U_i^c$ reduces to a single element a_i, that is $A_i = \{a_i\}$. Moreover, we can easily see that Definition 141 is a generalisation of both Definitions 49 and 140:

- if K is equal to the whole set of keys and if $A_i = U_i^c$, then it corresponds to subspace trails;
- if $U_i = U_{i+1}$ for all i, then it corresponds to invariant subspace trails.

In comparison to invariant subspace attacks, WKSTs have the potential of being better applicable to block ciphers with a non trivial key schedule. At the same time, with respect to subspace trails it is not necessary for WKSTs to hold for all possible keys. However, finding an example WKST that improves on previous results seems to be a hard problem. We thus do not follow this path further.

6.6 FURTHER WORK

By now, the results from this chapter are used in two ways. For the NIST lightweight candidate SATURNIN, Canteaut et al. [Can+19d] proved resistance against subspace trails by exploiting the structure of the trails we exhibited in Section 6.4.2.

[Can+19d] Canteaut et al. *Saturnin: a suite of lightweight symmetric algorithms for post-quantum security*

A recent application of the COMPUTETRAIL algorithm is due to Boura et al. [Bou+19]. Here the authors provide a new framework to prove the properties described by Grassi et al. [Gra+16; Gra+17] and apply them to other ciphers. For this, they need to find subspace trails of a specific structure, which they do by using Algorithm 1.

[Bou+19] Boura et al. “A General Proof Framework for Recent AES Distinguishers”

7

Towards Automated Key Recovery Attacks

"Algorithms are the computational content of proofs"
—Robert Harper

▸ SYNOPSIS We discuss an application of the COMPUTE TRAIL algorithm. This is based on unpublished work together with Patrick Derbez, Phil Hebborn, Virginie Lallemand, and Gregor Leander; all authors contributed equally.

The main goal is to automate the key recovery part of a cryptanalytic attack. That is, given a (differential) distinguisher over some rounds for a block cipher, the developed method should turn this into a key recovery attack.

7.1 KEY RECOVERY TECHNIQUES

Let us first recall the general approach of turning a distinguisher into a key recovery and the common ways of doing so. In the following, we only discuss the appending of rounds to a distinguisher; note that prepending rounds works basically in the same way, see also our initial discussion in Section 2.3.1, but we leave it out here for simplicity.

▸ COUNTING KEYS

Biham and Shamir proposed the *counting* technique for key recoveries in their seminal paper on differential cryptanalysis, see [BS91, End of Section 3]. Given a differential distinguisher on r rounds of our cipher under scrutiny, the counting technique appends s rounds on the ciphertext side. Assuming the distinguisher requires the input difference α, output difference β and has probability p, an attacker asks for $c \cdot p^{-1}$ encryptions of message pairs m_i and $m_i' = m_i + \alpha$. Denote these encryptions after $r+s$ rounds by c_i and c_i'. Running the below Algorithm 5, the adversary learns the most probable key $k = \textsc{Counting Key Recovery}(F, \alpha \rightrightarrows \beta, K, S)$, i. e. the candidate in K which most probable was used by the encryption oracle. The fundamental idea for this approach is that for a correct key guess the number of right pairs, i. e. the number of plaintext/ciphertext pairs following the distinguisher, is proportional to the probability p of the distinguisher. Note however that wrong pairs also exists for correct key guesses, as we are exploiting non-deterministic distinguishers.

[BS91] Biham and Shamir, "Differential Cryptanalysis of DES-like Cryptosystems"

Example key recovery attacks following this strategy are [Abe+15; LR17a]. Several extensions can improve this basic counting approach. The main idea for these extensions is to filter out wrong plaintext/ciphertext pairs

Algorithm 5 Basic Counting Key Recovery

```
Precondition:
        the round function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$,
        a distinguisher $\alpha \rightrightarrows^{F^r_k} \beta$,
        a set $K$ of involved key bits for the last $s$ rounds,
        a set of plaintext/ciphertext pairs $S = \{(m_i, m_i'), (c_i, c_i')\}$.
Postcondition: A key candidate $k \in K$

function COUNTING KEY RECOVERY($F$, $\alpha \rightrightarrows^{F^r_k} \beta$, $K$, $S$)
    $\forall k' \in K : \mathrm{cnt}_{k'} \leftarrow 0$    ▷ Set up a key counter for each candidate key.
    for all $k' \in K$ do
        for all $(m_i, m_i'), (c_i, c_i') \in S$ do
            $\beta' \leftarrow F^{-s}(c_i, k') + F^{-s}(c_i', k')$
            if $\beta = \beta'$ then
                $\mathrm{cnt}_{k'} \leftarrow \mathrm{cnt}_{k'} + 1$    ▷ Increase corresponding key counter.
    return key candidate $k$ such that $\forall k' \in K : \mathrm{cnt}_k \geqslant \mathrm{cnt}_{k'}$
```

suggesting wrong key candidates. Before reviewing those, we first discuss an alternative approach, due to Albrecht and Cid [AC09], applied in [Alb+10; Wan+11] and later generalised by Dinur [Din14].

Note that the efficiency of the counting key recovery, i. e. how many rounds s we can additionally cover, crucially relies on two facts. First, the set of key candidates K must be much smaller than 2^n, and second the applied filters must be strong enough, i. e. not suggest to many wrong keys. For counting based key recoveries, these filters are typically linear conditions on the plain- or ciphertext, as we see below. While such linear conditions have the advantage that we can handle these well in terms of precise mathematical models and understand their behaviour well, their main disadvantage is that the resulting key recovery may be weaker than with non-linear filters. On the other side, the main advantage of the enumeration technique is exactly that it handles non-linear filters.

► Enumerating Keys

The basic idea in [AC09] for the enumeration framework was to combine statistical attacks with algebraic based techniques, i. e. solving non-linear systems of equations with tools like Gröbner bases. Dinur [Din14] generalised this framework to allow for different techniques then algebraic ones. For this, the cipher is split into two parts: the one covered by the distinguisher and a second part, "sub-cipher", covered by the key recovery. The attack then enumerates, hence the name, all possible keys for this sub-cipher, until an event occurs that determines the correctness of the current key candidate. The initial proposal by Albrecht and Cid used the already mentioned algebraic techniques to solve this, Dinur applies a guess-and-determine technique and the attack on Zorro by Bar-On et al. [Bar+15] can also be viewed in this framework.

However, Dinur [Din14, p. 4] states that the enumeration technique often becomes too expensive when covering too many rounds s. Let us thus

[AC09] Albrecht and Cid, "Algebraic Techniques in Differential Cryptanalysis"

[Din14] Dinur "Improved Differential Cryptanalysis of Round-Reduced Speck"

[Bar+15] Bar-On et al. "Cryptanalysis of SP Networks with Partial Non-Linear Layers" look at a natural improvement to the counting technique.

► Early Abort

Lu et al. [Lu+08] proposed the *early abort* technique which is based on the simple idea of finding conditions which can be checked during partial en- or decryption of plaintext/ciphertext pairs, based on a subset of the guessed key-bits. These conditions are then used to discard pairs that cannot follow our distinguisher as early as possible, thus reducing the number of used wrong pairs. Typical conditions determine bit-values of intermediate differences or similar – thus, such conditions are *linear*.

[Lu+08] Lu et al. "Improving the Efficiency of Impossible Differential Cryptanalysis of Reduced Camellia and MISTY1"

Besides the reduction of wrong pairs, using this early checks also comes with the advantage of splitting the key material into smaller chunks that need to be guessed. A natural assumption is that this splitting allows for a (possible great) reduction in runtime.

In all what follows, we concentrate on key recovery attacks which are based on a differential cryptanalysis distinguisher and only use a basic counting technique with linear filtering. For other types of distinguishers and advanced key recovery strategies, other techniques may be needed. Eventually, our goal is to provide a solution for Problem 6 (Automate Key Recovery).

7.2 KEY RECOVERY BY EXAMPLE

As stated in the introduction, our goal is to produce an algorithm that, given a differential distinguisher and a fixed number of rounds, returns the corresponding best key recovery step and its complexities.

This section introduces our aim and the necessary notations by illustrating them on a toy example. We consider a cipher with a four branch Feistel Network structure, see Figure 7.1, and look for a key recovery that covers the last two rounds that separate the ciphertext from the end of a fixed differential. Thus, the state of the cipher consists of four words of n-bit each, round keys are $k_i = (k_{i,1}, k_{i,2})$ with $k_{i,j} \in \mathbb{F}_2^n$ and the round function uses the S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ to compute

$$F_{k_{i,1},k_{i,2}} : (x_1, x_2, x_3, x_4) \mapsto (x_4, x_1 + S(x_2 + k_{i,1}), x_2, x_3 + S(x_4 + k_{i,2}))\ .$$

To be more specific about the round function, we instantiate S with the 4-bit PRESENT S-box, see Table 7.1 In our example, we suppose that the differential only has one active word (second word from left, in red in Figure 7.1). In particular, our distinguisher ends in, and our starting subspace is thus,

$$U_0 = \{0\} \times \mathbb{F}_2^4 \times \{0\} \times \{0\}\ .$$

TABLE 7.1: The 4-bit S-box used in PRESENT.

x	$S(x)$
0	c
1	5
2	6
3	b
4	9
5	0
6	a
7	d
8	3
9	e
a	f
b	8
c	4
d	7
e	1
f	2

► Step 1: Forward Propagation

The first step an adversary can do, is to append rounds to the distinguisher. For this, we have to trace how the output differences of the distinguisher evolves through these additional rounds. In our example two cases occur: 1. the active word is shifted to a new position, then this new position is active in the next round, or 2. the active word is fed into an S-box. We could either

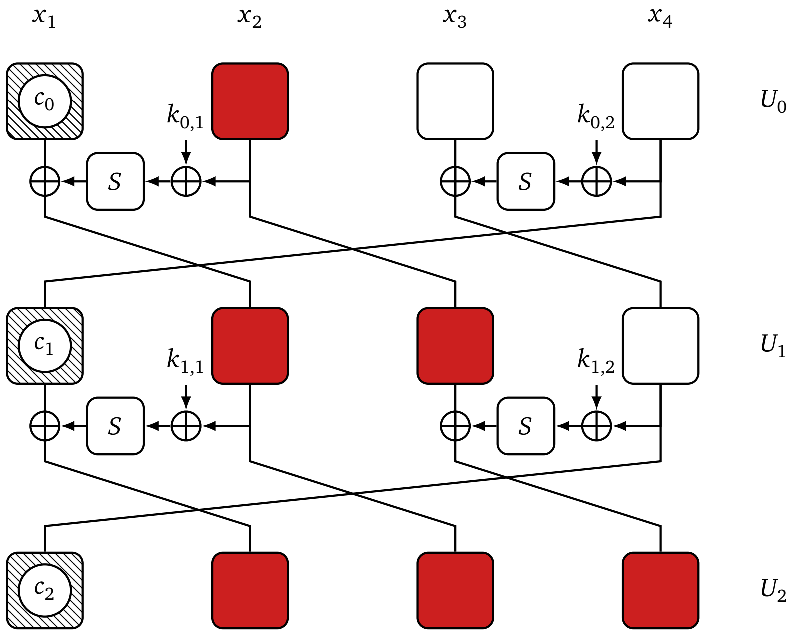


FIGURE 7.1: Example of a 2-round key recovery associated to differential distinguisher. U_0 is the output difference of the distinguisher, the start of our key recovery. The key recovery captures two further rounds of encryption. The red state blocks mark active words in the distinguisher, the hatched blocks words on which we have conditions to check during the partial decryption.

not consider any details of the S-box here, then we just assume an active S-box translates to an active output word, or compute the actual influence of the S-box. Using this two propagation criteria, we can propagate the initial active word through the two rounds to three active words in the actual output of the cipher. We denote these possible intermediate differences U_0, U_1 and U_2.

For our example, the subspaces are

$$\begin{aligned} U_0 &= \{0\} \times \mathbb{F}_2^4 \times \{0\} \times \{0\} \text{ (as before)}, \\ U_1 &= \{0\} \times \mathbb{F}_2^4 \times \ \mathbb{F}_2^4 \ \times \{0\} \text{ and} \\ U_2 &= \{0\} \times \mathbb{F}_2^4 \times \ \mathbb{F}_2^4 \ \times \ \mathbb{F}_2^4 \ . \end{aligned}$$

As we capture all possible difference-transitions with this propagation, we know for sure that a right ciphertext pair has to fulfill these differences. A first observation we can make is that every right pair's difference has to be passive in the left most word. This gives us the first conditions, depicted c_2 in Figure 7.1, we can use for filtering wrong pairs. Next we look at how to get further such conditions.

▸ Step 2: Backward Propagation and Determination of Conditions

As we now know with which ciphertext pairs we have to work, we can simply reverse the approach and propagate the possible differences backwards.

Starting with a difference α in U_2 we are thus interested in all possible differences that α could originate from. Note that, in order to introduce some structure, we are actually working with subspaces spanned by the possible intermediate differences. While computing the span of the differences introduce impossible differences, i. e. differences that can not occur for a right pair, we use the following conditions, to filter out such impossible differentials and corresponding wrong pairs.

Reversing the approach gives us now subspaces V_1 and V_0, corresponding to U_1 and U_0, such that we have the following properties 1. propagating a difference in U_i forward, gives a difference in U_{i+1}, 2. propagating a difference in U_i backward, gives a difference in V_{i-1}, and 3. U_i is contained in V_i. The goal of this backward propagation is to determine conditions with which we can partially check and discard wrong pairs early where possible. Thus, during the key recovery phase we will partially decrypt a ciphertext pair and check if the differences follow the possible intermediate ones, i. e. lay in the U_i's. If they do, we continue with partial decryption. If they do not lay in U_i, they will instead be in V_i. Hence, the conditions we have to check during partial decryption is if we are in V_i but not in U_i and if so, discard the pair. From this information we then deduce the conditions c_i.

In our case, we get

$$\begin{aligned} V_0 &= \mathbb{F}_2^4 \times \mathbb{F}_2^4 \times \{0\} \times \{0\} \text{ and} \\ V_1 &= \mathbb{F}_2^4 \times \mathbb{F}_2^4 \times \ \mathbb{F}_2^4 \ \times \{0\} \,, \end{aligned}$$

the corresponding conditions are

$$c_0 = c_1 = c_2 = \mathbb{F}_2^4 \times \{0\} \times \{0\} \times \{0\} \,.$$

We now have a basic key recovery strategy, consisting of a distinguisher (predefined), possible right intermediate differences (U_i) and possible wrong intermediate differences (c_i deduced from U_i and V_i). The next step is to compute these information algorithmically.

7.3 ALGORITHMS FOR KEY RECOVERY

Following the above example, we discuss algorithmic ways to solve each of the needed tasks, i. e. forward and backward propagation, condition determination, and building the key recovery strategy.

▸ FORWARD AND BACKWARD PROPAGATION

Propagating the difference subspaces can be done with the COMPUTE TRAIL Algorithm 1. The forward propagation is based on computing the trail starting in U_0 over iterations of the round function F:

$$\textsc{Compute Trail}(F, U_0) = U_0 \overset{F}{\rightrightarrows} \cdots \overset{F}{\rightrightarrows} U_s \,.$$

For the backward propagation and the computation of the V_i subspaces, we simply use the same approach with the inverse round function F^{-1}, but stop the propagation after one iteration:

$$\textsc{Compute Trail}(F^{-1}, U_i) = U_i \overset{F^{-1}}{\rightrightarrows} V_{i-1} \quad \forall 1 \leqslant i \leqslant s \,.$$

We only compute one propagation here, because we check the intermediate differences in each round.

▸ SWITCHING TO AFFINE SUBSPACES

Similar to the case of subspace trails, we can actually get some more information on the involved subspace when we turn from linear subspaces to affine subspaces. While for the security argument against subspace trails[1] the propagation of affine subspaces is not much helpful, there are no such

[1] Recall that the problem there is to find a suitable reduction in starting points.

Algorithm 6 Computation of truncated differentials

Precondition: A nonlinear function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, an affine subspace $U{+}a$.
Postcondition: A truncated differential trail $U{+}a \rightrightarrows^F \cdots \rightrightarrows^F V{+}b$.

```
 1 function COMPUTE AFFINE TRAIL($F$, $U{+}a$)
 2     if $\dim(U) = n$ then
 3         return $U{+}a$
 4     $V \leftarrow \emptyset$
 5     $b \leftarrow \Delta_a(F)(0)$
 6     for $u_i \in \mathrm{Basis}(U) \cup \{0\}$ do
 7         for enough $x \in_R \mathbb{F}_2^n$ do
 8             $V \leftarrow V \cup \Delta_{u_i+a}(F)(x) + b$
 9     $V \leftarrow \mathrm{Span}\{V\}$
10     return $U{+}a \rightrightarrows^F$ COMPUTE AFFINE TRAIL($F, V{+}b$)
```

problems for this application. A modified COMPUTE TRAIL algorithm which handles affine subspaces is the following.

Compared to Algorithm 1, we now have to also propagate the displacement vector a through the round function and consider the zero vector in Line 6 of Algorithm 6. Apart from this minor changes, the runtime analysis stays the same.

▸ DETERMINING CONDITIONS

Next, we have to determine the conditions, the ciphertext pairs have to follow in order to be valid for the exploited distinguisher, based on the U_i's and V_i's computed in the previous step.

The conditions are of the following kind. With probability one, the difference moves from U_{i+1} to V_i, while we actually want the difference to be in $U_i \subseteq V_i$.

We thus want to identify a minimal set of conditions $\{\alpha\}$ such that

$$x \in U_i \Leftrightarrow \alpha \cdot x = 0\,.$$

This is obviously fulfilled for all $\alpha \in {U_i}^\perp$. But, as from $U_i \subseteq V_i$ it follows that ${V_i}^\perp \subseteq {U_i}^\perp$, see Lemma 9, and we already know that x is going to be in V_i, parts of these conditions are already fulfilled. More precisely, write

$${U_i}^\perp = {V_i}^\perp \oplus W\,.$$

Then $\alpha \in {U_i}^\perp$ can be written as $\alpha = \beta + \gamma$ with $\beta \in {V_i}^\perp$ and $\gamma \in W$. Further

$$\alpha \in {U_i}^\perp \quad \Leftrightarrow \quad \alpha \cdot x = 0 \quad \forall x \in U \quad \Leftrightarrow \quad \beta \cdot x + \gamma \cdot x = 0\,,$$

with $\beta \in {V_i}^\perp$, and $\gamma \in W$. From $x \in V_i$ we can now deduce $\beta \cdot x = 0$ and thus the only condition that is left to check, is

$$\gamma \cdot x = 0\,.$$

It is thus enough to check all $\gamma \in W$.

Returning to our example, the resulting conditions C_i are computed as this direct sum ${U_i}^\perp = {V_i}^\perp \oplus C_i$:

$${U_1}^\perp = {V_1}^\perp \oplus C_1, \qquad {U_0}^\perp = {V_0}^\perp \oplus C_0\,.$$

- SWITCHING TO AFFINE CONDITIONS

When working with affine subspaces $U_{i+1}+a_{i+1}$, the resulting backward propagated subspaces are V_i+b_i. In other words, for every input difference in $U_{i+1}+a_{i+1}$ we then know that it will be propagated through the round to an $x \in V_i+b_i$, where we want to check, if $x \in U_i+a_i$. Note that first, as U_i+a_i is contained in V_i+b_i, we can change the translation vector b_i to be a_i; and then second $x + a_i \in V$. Again, the obvious conditions are then of the form $\alpha \in U_i^\perp$ and

$$x \in U_i+a_i \Leftrightarrow \alpha \cdot (x + a_i) = 0$$

Following a similar argument as above, we have

$$\alpha \in {U_i}^\perp \quad \Leftrightarrow \quad \alpha \cdot (x+a_i) = 0 \quad \forall x \in U \quad \Leftrightarrow \quad \beta \cdot (x+a_i) + \gamma \cdot (x+a_i) = 0\,,$$

with $U_i^\perp = V_i^\perp + C_i$, $\beta \in V_i^\perp$, $\gamma \in C_i$ as above, and thus

$$\gamma \cdot (x + a_i) = 0$$

is the actual condition we need to check.

7.3.1 *Influencing Conditions*

So far, we have computed the intermediate differences and conditions we want to check when validating a ciphertext pair. However, until now, we have not covered the role of the round keys in this process. In order to partially decrypt a ciphertext, we of course have to guess (parts of) the involved round keys. The efficiency of a key recovery strategy is greatly influenced by this key guessing part. We thus now study the influence of the round keys.

Definition 142 (Key bits influencing a condition). Given a round function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, subspaces U, C, and K (input differences, conditions, and keys). We say *the linear combination $\delta_k \in K$ of key bits influences the condition $\alpha \in C$*, if and only if for all x and k

$$\exists \delta_x \in U \ : \ \alpha \cdot \Delta_{(\delta_x,0)}\big(\Delta_{(0,\delta_k)}(F)\big)(x,k) \text{ is not constant.}$$

Conversely, the linear combination *does not influence* the condition, if and only if

$$\forall \delta_x \in U \ : \ \alpha \cdot \Delta_{(\delta_x,0)}\big(\Delta_{(0,\delta_k)}(F)\big)(x,k) \text{ is constant.}$$

We define the set of independences, i. e. the set of all linear combinations that do not influence a given condition α, as

$$I(\alpha) := \{\delta_k \mid \delta_k \text{ does not influence the condition } \alpha\}$$

Lemma 143. *$I(\alpha)$ is a subspace.*

Proof. To show that $I(\alpha)$ is a subset of $\mathbb{F}_2^n$ we need to prove that it contains 0 and that it is closed under summation. The first point is easy to prove since for all δ_x and $\alpha \in \mathbb{F}_2^n$:

$$\alpha \cdot \Delta_{(\delta_x,0)}\big(\Delta_{(0,0)}(F)\big)(x,k) = 0.$$

Second, assume that δ_{k_1} and δ_{k_2} are in $I(\alpha)$. From the definition, it implies that

$$\forall \delta_x \in U \ : \ \alpha \cdot (F(x+\delta_x, k+\delta_{k_1}) + F(x+\delta_x, k) + F(x, k+\delta_{k_1}) + F(x,k))$$

is constant and

$$\forall \delta_x \in U \ : \ \alpha \cdot (F(x+\delta_x, k+\delta_{k_2}) + F(x+\delta_x, k) + F(x, k+\delta_{k_2}) + F(x,k))$$

is constant.

Since K is a subspace and contains both δ_{k_1} and δ_{k_2}, we can introduce $k' \in K$ defined by $k' = k + \delta_{k_1}$. By simple manipulations of the two previous relations we obtain that

$$\forall \delta_x \in U \ : \ \alpha \cdot \begin{pmatrix} F(x+\delta_x, k'+\delta_{k_1}+\delta_{k_2}) + F(x+\delta_x, k') \\ \qquad + F(x, k'+\delta_{k_1}+\delta_{k_2}) + F(x,k') \end{pmatrix}$$

is constant, which concludes the proof.

Now, for every condition, we need to find the key bits which influence it. The naïve way to compute this dependencies is the following Algorithm 7.

Algorithm 7 Naïve Dependencies Check

Precondition:

A round function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$,

a subspace U of input differences,

a subspace C of conditions, and

a set K of key differences.

Postcondition: The set of independences $I(\alpha)$ for any $\alpha \in C$.

```
function CHECK DEPENDENCIES($F$, $U$, $C$, $K$)
    for all $\alpha \in C$ do
        for all $\delta_k \in K$ do
            for enough $x \in_R \mathbb{F}_2^n$, $k \in_R \mathbb{F}_2^n$, and $\delta_x \in_R U$ do
                $\beta \leftarrow F(x,k)+F(x+\delta_x,k)+F(x,k+\delta_k)+F(x+\delta_x,k+\delta_k)$
                if $\alpha \cdot \beta$ is not constant for all $x$ and $k$ then
                    $\delta_k$ influences the condition $\alpha$
    return all $\delta_k$ which do or do not influence any condition
```

Next, when we want to check if condition α holds, we have to guess only keys *up to elements in* $I(\alpha)$. This is because if two keys k and k' differ by some elements in $I(\alpha)$ they either are both good or both bad, because the condition is independent of the difference. So we have to make key guesses actually in

$$\mathbb{F}_2^n / I(\alpha)$$

that is the quotient space. In practice, we have to pick some representative for this space and we can do this simply by extending a basis for $I(\alpha)$ to a basis of $\mathbb{F}_2^n$ and our key guesses are then linear combinations of the extended part of the basis. In other words, we pick some (arbitrary) space $I(\alpha)^c$ such that

$$\mathbb{F}_2^n = I(\alpha) \oplus I(\alpha)^c$$

and do key guesses in $I(\alpha)^c$. Note that it might not be possible to take the dual of $I(\alpha)$ as $I(\alpha)^c$ but if possible this is probably the nicest choice.

Let us conclude with an example comparing linear and affine subspace propagations.

Example 144 (Linear vs. affine subspaces). We again use the generalised Feistel structure depicted in Figure 7.1, but with a different distinguisher end point U_0. Here, we assume a standard differential distinguisher, thus ending in one possible output difference $(\begin{smallmatrix}0&8&0&0\end{smallmatrix})$. In the following, we give the bases of the respective subspaces.

Step 1 Using linear subspaces, we end up with the following extension over three rounds.[2]

$$\begin{array}{ccccc}
U_0 & \rightrightarrows & U_1 & \rightrightarrows & U_2 \\
\left\{\begin{smallmatrix}0&8&0&0\end{smallmatrix}\right\} & \rightrightarrows & \left\{\begin{smallmatrix}0&8&0&0\\0&4&0&0\\0&2&0&0\\0&1&8&0\end{smallmatrix}\right\} & \rightrightarrows & \left\{\begin{smallmatrix}0&8&0&0\\0&4&0&0\\0&2&0&0\\0&1&0&0\\0&0&8&0\\0&0&4&0\\0&0&2&0\\0&0&1&8\end{smallmatrix}\right\}
\end{array}$$

Comparing to affine subspaces, we save one dimension in each round.

$$\begin{array}{ccccc}
U_0{+}a_0 & \rightrightarrows & U_1{+}a_1 & \rightrightarrows & U_2{+}a_2 \\
\left\{\begin{smallmatrix}0&0&0&0\end{smallmatrix}\right\}{+}\left(\begin{smallmatrix}0&8&0&0\end{smallmatrix}\right) & \rightrightarrows & \left\{\begin{smallmatrix}0&8&0&0\\0&4&0&0\\0&2&0&0\end{smallmatrix}\right\}{+}\left(\begin{smallmatrix}0&1&8&0\end{smallmatrix}\right) & \rightrightarrows & \left\{\begin{smallmatrix}0&8&0&0\\0&4&0&0\\0&2&0&0\\0&1&0&0\\0&0&8&0\\0&0&4&0\\0&0&2&0\end{smallmatrix}\right\}{+}\left(\begin{smallmatrix}0&0&1&8\end{smallmatrix}\right)
\end{array}$$

Step 2 Next we compute the backward propagated V_i, resp. $V_i{+}b_i$.

$$\begin{array}{ccccccc}
V_0 & \leftleftarrows & U_1 & \quad & V_1 & \leftleftarrows & U_2 \\
\left\{\begin{smallmatrix}8&0&0&0\\4&0&0&0\\2&0&0&0\\1&0&0&0\\0&8&0&0\end{smallmatrix}\right\} & \leftleftarrows & \left\{\begin{smallmatrix}0&8&0&0\\0&4&0&0\\0&2&0&0\\0&1&8&0\end{smallmatrix}\right\} & & \left\{\begin{smallmatrix}8&0&0&0\\4&0&0&0\\2&0&0&0\\1&0&0&0\\0&8&0&0\\0&4&0&0\\0&2&0&0\\0&1&8&0\end{smallmatrix}\right\} & \leftleftarrows & \left\{\begin{smallmatrix}0&8&0&0\\0&4&0&0\\0&2&0&0\\0&1&0&0\\0&0&8&0\\0&0&4&0\\0&0&2&0\\0&0&1&8\end{smallmatrix}\right\}
\end{array}$$

Again, we save one dimension in the affine case.

$$\begin{array}{ccc}
V_0{+}b_0 & \leftleftarrows & U_1{+}a_1 \\
\left\{\begin{smallmatrix}8&0&0&0\\4&0&0&0\\2&0&0&0\\1&0&0&0\end{smallmatrix}\right\}{+}\left(\begin{smallmatrix}0&8&0&0\end{smallmatrix}\right) & \leftleftarrows & \left\{\begin{smallmatrix}0&8&0&0\\0&4&0&0\\0&2&0&0\end{smallmatrix}\right\}{+}\left(\begin{smallmatrix}0&1&8&0\end{smallmatrix}\right) \\
V_1{+}b_1 & \leftleftarrows & U_2{+}a_2 \\
\left\{\begin{smallmatrix}8&0&0&0\\4&0&0&0\\2&0&0&0\\1&0&0&0\\0&8&0&0\\0&4&0&0\\0&2&0&0\end{smallmatrix}\right\}{+}\left(\begin{smallmatrix}0&1&8&0\end{smallmatrix}\right) & \leftleftarrows & \left\{\begin{smallmatrix}0&8&0&0\\0&4&0&0\\0&2&0&0\\0&1&0&0\\0&0&8&0\\0&0&4&0\\0&0&2&0\end{smallmatrix}\right\}{+}\left(\begin{smallmatrix}0&0&1&8\end{smallmatrix}\right)
\end{array}$$

Step 3 Computing the conditions gives the same C_i's for the linear subspace case as for the first example:

$$C_0 = C_1 = \left\{\begin{smallmatrix}8&0&0&0\\4&0&0&0\\2&0&0&0\\1&0&0&0\end{smallmatrix}\right\}.$$

Recall that for affine subspaces, the conditions are of the form (C_i, a_i) as we have to check $\gamma \cdot (x + a_i) = 0$ for every $\gamma \in C_i$. Here, a_i is the

[2] Stopping the extension after three rounds is somewhat arbitrary to not lengthen the example too much.

translation vector of the corresponding affine subspace $U_i{+}a_i$. The conditions are thus

$$C_0,\ a_0 = \left\{\begin{smallmatrix}8&0&0&0\\4&0&0&0\\2&0&0&0\\1&0&0&0\end{smallmatrix}\right\},\ (\texttt{0 8 0 0})$$

and

$$C_1,\ a_1 = \left\{\begin{smallmatrix}8&0&0&0\\4&0&0&0\\2&0&0&0\\1&0&0&0\end{smallmatrix}\right\},\ (\texttt{0 1 8 0})\,.$$

Step 4 For the sake of brevity, we only compute the independences for the first round here, i. e. we concentrate on $F_{k_{0,1},k_{0,2}}$ and thus independences of $k_0 = (k_{0,1}, k_{0,2}) \in \mathbb{F}_2^{4+4}$. For each of the 15 elements $\alpha_i = \mathtt{i} \in C_0$ (we ignore the trivial condition $\alpha_0 = 0$), we compute $I(\alpha_i)$ as given in Table 7.2. Two obvious observations are 1. every condition α_i is independent of all bits in $k_{0,2}$, which is directly clear from Figure 7.1, as the second round key $k_{0,2}$ only influences the right most nibble x_4, and 2. the conditions for the affine subspace case are the same, as $\alpha_i \cdot (\texttt{0 8 0 0}) = 0$ for all α_i and thus the affine part does not influence the conditions in this case.

A more interesting observation is that all conditions are independent of the leftmost bit of $k_{0,1}$, i. e. independent of $8 \cdot k_{0,1}$. During our key recovery attack, we thus do not have to guess this key bit.

Table 7.2: Exemplary $I(\alpha)$

α_i	$I(\alpha_i)$
$\alpha_{\texttt{1000}}$	$\mathrm{Span}\left\{\begin{smallmatrix}8\\2\end{smallmatrix}\right\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{2000}}$	$\mathrm{Span}\{\texttt{8}\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{3000}}$	$\mathrm{Span}\left\{\begin{smallmatrix}8\\4\end{smallmatrix}\right\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{4000}}$	$\mathrm{Span}\left\{\begin{smallmatrix}8\\6\end{smallmatrix}\right\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{5000}}$	$\mathrm{Span}\{\texttt{8}\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{6000}}$	$\mathrm{Span}\left\{\begin{smallmatrix}8\\5\\2\end{smallmatrix}\right\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{7000}}$	$\mathrm{Span}\left\{\begin{smallmatrix}8\\2\end{smallmatrix}\right\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{8000}}$	$\mathrm{Span}\left\{\begin{smallmatrix}8\\4\\1\end{smallmatrix}\right\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{9000}}$	$\mathrm{Span}\{\texttt{8}\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{a000}}$	$\mathrm{Span}\left\{\begin{smallmatrix}8\\6\end{smallmatrix}\right\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{b000}}$	$\mathrm{Span}\left\{\begin{smallmatrix}8\\4\end{smallmatrix}\right\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{c000}}$	$\mathrm{Span}\{\texttt{8}\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{d000}}$	$\mathrm{Span}\{\texttt{8}\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{e000}}$	$\mathrm{Span}\left\{\begin{smallmatrix}8\\5\\3\end{smallmatrix}\right\} \times \mathbb{F}_2^4$
$\alpha_{\texttt{f000}}$	$\mathrm{Span}\{\texttt{8}\} \times \mathbb{F}_2^4$

7.4 FUTURE WORK

While the sketched approach looks promising to yield an usable algorithmic way to detect key recovery strategies, there are still many open problems to solve. The most import one is how the up to here found information can be turned into the most effective key recovery. It might well be the case that some $I(\alpha)$ intersect (e. g. for $\alpha_{\texttt{6000}}$ and $\alpha_{\texttt{e000}}$ in Table 7.2); such synergies should be exploited. Further, some conditions might be independent of many more key bits, than others. This implies that for such conditions only very few key bits have to be guessed, thus these conditions should be checked early. Hence, developing a search strategy for this condition ordering is an important next problem.

Apart from this very basic unsolved problem, others will arise from applying the above algorithms to practical block ciphers. The big block sizes of such instances render exhaustive checks, e. g. over all possible round keys, impossible. For these applications we most probable have to revert to heuristics.

Finally, any automated key recovery should be tested against known key recovery attacks on ciphers. Only this practical verification will in the end approve this whole approach.

Part IV

EPILOGUE

8

Conclusion

"But at the laste, every thing hath ende"
—Geoffrey Chaucer

Since the development of the DES and AES, our understanding of secure designs for encryption schemes has greatly evolved. In particular in the area of symmetric cryptography, we are today, after more than 40 years of research, able to design very efficient ciphers, which we firmly believe to be secure – with the AES being the prime example withstanding 20 years of cryptanalysis. Our progress pushed efficiency bounds further and further, especially within the trend of lightweight cryptography.

However the time may has come where we should shift our focus to improving security arguments for new designs – because the improvement since the development of bounds for differential and linear cryptanalysis seems marginal. We see this thesis, specifically the first part on security arguments, as a step in this direction. With our block cipher instances BISON and WISENT we are for the first time able to give precise bounds on the *differential* instead of only on differential trails. This initial result may lead to further investigation of alternative constructions for block ciphers. An interesting question in this direction is if a construction can be found which exhibits similar good properties with respect to linear cryptanalysis. A second worthwhile direction is the study of unbalanced Feistel networks which seem to be related to the WSN construction.

Apart from our results on differential cryptanalysis, our study of the ACT revealed a connection between differential-linear cryptanalysis and previously studied properties of Boolean functions. In our opinion the most interesting observation, from a cryptanalytic perspective, is that the decryption function might be weaker than the encryption against differential-linear attacks. This result implies future analysis has to be extended in this direction. From a more theoretical point of view, it is interesting that vectorial Boolean functions exhibit a lower bound for the absolute indicator, while for Boolean functions it seems to be a hard problem finding such a lower bound. Overall, our results on this new connection contribute to a further understanding of differential-linear cryptanalysis.

In the second part of the thesis, we concentrated on automated tools for the design and analysis of block ciphers. Our main result here was the conceptual simple algorithm for propagating subspaces through an iterative round function. Despite the underlying simple idea, this algorithm turns out to be useful not only for one application. For its original purpose, we use

Compute Trail to algorithmically bound the longest subspace trail through an SPN cipher and thus construct an algorithmic security argument against this recent type of attack.

However, besides the study of single attacks, a more principle task is to extend a distinguishing attack into a key recovery. Especially when such an extension is possible over some rounds, it might make the difference between a cipher with a thin security margin and a broken one. Thus, while being a very important part of cryptanalysis, finding key recovery strategies remains a highly manual, and thus error prone, task. As discussed in the last chapter, our subspace trail algorithm may be used in an automatisation approach for exactly this problem – albeit working out the exact techniques for such an automated key recovery remains to be done.

Apart from these possibilities for automated tools discussed in this thesis, a different application are cryptanalysis techniques based on MILPs. We only briefly mentioned MILPs for bounding the number of active S-boxes. However, they have by now a broad spectrum of use cases, e. g. for finding differential or linear trails, for finding division properties or similar. All these applications have the same basic process that needs the cipher under scrutiny and the analysis technique to be modeled as an instance of the specific programming style, i. e. as a MILP. The needed building blocks for these models are known for every typical part used in ciphers, still the cryptanalyst has to assemble the models manually. Again this is a tedious and error prone task which could easily be automated. The development of such a MILP compiler (or similarly a SAT compiler for constrained programming models) quite likely requires techniques from programming languages and compiler theory. It seems to be an interesting problem to work on.

Finding the best representation of a cipher for these models (both for MILPs and SAT) is another problem which yet remains unsolved. This occurs especially when modeling the nonlinear S-boxes, for which different approaches exist: broadly speaking one could model the S-box in full detail, or try to pre-optimise the model on a varying level. Similar to the XOR count optimisations it is then unclear, how much pre-optimisation helps in the end and what level of optimisation restricts the solver too much for its own optimisation strategies.

Bibliography

[Abd+12] Mohamed Ahmed Abdelraheem, Martin Ågren, Peter Beelen, and Gregor Leander. "On the Distribution of Linear Biases: Three Instructive Examples". In: *CRYPTO 2012*. Ed. by Reihaneh Safavi-Naini and Ran Canetti. Vol. 7417. LNCS. Springer, Heidelberg, Aug. 2012, pp. 50–67. DOI: 10.1007/978-3-642-32009-5_4 (cit. on p. 31).

[Abe+15] Farzaneh Abed, Eik List, Stefan Lucks, and Jakob Wenzel. "Differential Cryptanalysis of Round-Reduced Simon and Speck". In: *FSE 2014*. Ed. by Carlos Cid and Christian Rechberger. Vol. 8540. LNCS. Springer, Heidelberg, Mar. 2015, pp. 525–545. DOI: 10.1007/978-3-662-46706-0_27 (cit. on p. 143).

[Ada97] Carlisle M. Adams. "Constructing Symmetric Ciphers Using the CAST Design Procedure". In: *Designs, Codes and Cryptography* 12.3 (1997), pp. 283–316. DOI: 10.1023/A:1008229029587 (cit. on p. 51).

[AES] *Advanced Encryption Standard (AES)*. National Institute of Standards and Technology (NIST), FIPS PUB 197, Nov. 2001 (cit. on p. 22).

[Alb+10] Martin R. Albrecht, Carlos Cid, Thomas Dullien, Jean-Charles Faugère, and Ludovic Perret. "Algebraic Precomputations in Differential and Integral Cryptanalysis". In: *Inscrypt*. Vol. 6584. LNCS. Springer, 2010, pp. 387–403. DOI: 10.1007/978-3-642-21518-6_27 (cit. on p. 144).

[Alb+14] Martin R. Albrecht, Benedikt Driessen, Elif Bilge Kavun, Gregor Leander, Christof Paar, and Tolga Yalçin. "Block Ciphers - Focus on the Linear Layer (feat. PRIDE)". In: *CRYPTO 2014, Part I*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. LNCS. Springer, Heidelberg, Aug. 2014, pp. 57–76. DOI: 10.1007/978-3-662-44371-2_4 (cit. on pp. 106, 108, 120, 138).

[AC09] Martin Albrecht and Carlos Cid. "Algebraic Techniques in Differential Cryptanalysis". In: *FSE 2009*. Ed. by Orr Dunkelman. Vol. 5665. LNCS. Springer, Heidelberg, Feb. 2009, pp. 193–208. DOI: 10.1007/978-3-642-03317-9_12 (cit. on p. 144).

[Alf+18] Gianira Alfarano, Christof Beierle, Takanori Isobe, Stefan Kölbl, and Gregor Leander. "ShiftRows Alternatives for AES-like Ciphers and Optimal Cell Permutations for Midori and Skinny". In: *IACR Trans. Symm. Cryptol.* 2018.2 (2018), pp. 20–47. ISSN: 2519-173X. DOI: 10.13154/tosc.v2018.i2.20-47 (cit. on p. 41).

[And+13] Elena Andreeva, Andrey Bogdanov, Yevgeniy Dodis, Bart Mennink, and John P. Steinberger. "On the Indifferentiability of Key-Alternating Ciphers". In: *CRYPTO 2013, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. LNCS. Springer, Heidelberg, Aug. 2013, pp. 531–550. DOI: 10.1007/978-3-642-40041-4_29 (cit. on p. 26).

[And+14] Elena Andreeva, Begül Bilgin, Andrey Bogdanov, Atul Luykx, Florian Mendel, Bart Mennink, Nicky Mouha, Qingju Wang, and Kan Yasuda. *PRIMATEs v1.02*. Submission to the CAESAR competition. 2014 (cit. on p. 113).

[Aum] Aumasson et al. *Cryptography in industrial embedded systems: our experience of the needs and constraints*. Available at https://blog.teserakt.io/2019/06/03/cryptography-in-industrial-embedded-systems-our-experience-of-the-needs-and-constraints/, visited 9th June 2019. (Cit. on p. 4).

[Ava17] Roberto Avanzi. "The QARMA Block Cipher Family". In: *IACR Trans. Symm. Cryptol.* 2017.1 (2017), pp. 4–44. ISSN: 2519-173X. DOI: 10.13154/tosc.v2017.i1.4-44 (cit. on pp. 41, 106, 120, 135, 138).

[BL12] Dongxia Bai and Leibo Li. "New Impossible Differential Attacks on Camellia". In: *ISPEC*. Vol. 7232. LNCS. Springer, 2012, pp. 80–96. DOI: 10.1007/978-3-642-29101-2_6 (cit. on p. 140).

[Ban+15] Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, Kyoji Shibutani, Harunaga Hiwatari, Toru Akishita, and Francesco Regazzoni. "Midori: A Block Cipher for Low Energy". In: *ASIACRYPT 2015, Part II*. Ed. by Tetsu Iwata and Jung Hee Cheon. Vol. 9453. LNCS. Springer, Heidelberg, 2015, pp. 411–436. DOI: 10.1007/978-3-662-48800-3_17 (cit. on pp. 106, 120).

[Ban+16a] Subhadeep Banik, Andrey Bogdanov, and Francesco Regazzoni. "Atomic-AES: A Compact Implementation of the AES Encryption/Decryption Core". In: *INDOCRYPT 2016*. Ed. by Orr Dunkelman and Somitra Kumar Sanadhya. Vol. 10095. LNCS. Springer, Heidelberg, Dec. 2016, pp. 173–190. DOI: 10.1007/978-3-319-49890-4_10 (cit. on p. 111).

[Ban+16b] Subhadeep Banik, Andrey Bogdanov, and Francesco Regazzoni. *Atomic-AES v2.0*. Cryptology ePrint Archive, Report 2016/1005. http://eprint.iacr.org/2016/1005. 2016 (cit. on p. 111).

[Ban+17] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. "GIFT: A Small Present - Towards Reaching the Limit of Lightweight Encryption". In: *CHES 2017*. Ed. by Wieland Fischer and Naofumi Homma. Vol. 10529. LNCS. Springer, Heidelberg, Sept. 2017, pp. 321–345. DOI: 10.1007/978-3-319-66787-4_16 (cit. on pp. 135, 138).

[Ban+19] Subhadeep Banik, Yuki Funabiki, and Takanori Isobe. "More results on Shortest Linear Programs". In: *IWSEC 2019*. LNCS. Springer, Aug. 2019. IACR PREPRINT: 2019/856 (cit. on p. 118).

[Bar+15] Achiya Bar-On, Itai Dinur, Orr Dunkelman, Virginie Lallemand, Nathan Keller, and Boaz Tsaban. "Cryptanalysis of SP Networks with Partial Non-Linear Layers". In: *EUROCRYPT 2015, Part I*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. LNCS. Springer, Heidelberg, Apr. 2015, pp. 315–342. DOI: 10.1007/978-3-662-46800-5_13 (cit. on p. 144).

[Bar+18] Achiya Bar-On, Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. "Improved Key Recovery Attacks on Reduced-Round AES with Practical Data and Memory Complexities". In: *CRYPTO 2018, Part II*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10992. LNCS. Springer, Heidelberg, Aug. 2018, pp. 185–212. DOI: 10.1007/978-3-319-96881-0_7 (cit. on p. 9).

[BO+19] Achiya Bar-On, Orr Dunkelman, Nathan Keller, and Ariel Weizmann. *DLCT: A New Tool for Differential-Linear Cryptanalysis*. To appear at Eurocrypt 2019; Preprint available as Cryptology ePrint Archive, Report 2019/256. https://ia.cr/2019/256. 2019 (cit. on p. 56).

[Bar+19] Achiya Bar-On, Orr Dunkelman, Nathan Keller, and Ariel Weizman. "DLCT: A New Tool for Differential-Linear Cryptanalysis". In: *EUROCRYPT 2019, Part I*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11476. LNCS. Springer, Heidelberg, May 2019, pp. 313–342. DOI: 10.1007/978-3-030-17653-2_11 (cit. on pp. 6, 7, 43, 57).

[Bar+10] Paulo S. L. M. Barreto, Ventzislav Nikov, Svetla Nikova, Vincent Rijmen, and Elmar Tischhauser. "Whirlwind: a new cryptographic hash function". In: *Designs, Codes and Cryptography* 56.2–3 (2010), pp. 141–162. DOI: 10.1007/s10623-010-9391-y (cit. on pp. 105, 108, 120).

[BRa] Paulo Barreto and Vincent Rijmen. *The ANUBIS Block Cipher*. First Open NESSIE Workshop (cit. on pp. 120, 135).

[BRb] Paulo Barreto and Vincent Rijmen. *The Khazad legacy-level Block Cipher*. First Open NESSIE Workshop (cit. on p. 120).

[BRc] Paulo Barreto and Vincent Rijmen. *The* WHIRLPOOL *Hashing Function*. First Open NESSIE Workshop (cit. on p. 120).

[Bei+16a] Christof Beierle, Thorsten Kranz, and Gregor Leander. "Lightweight Multiplication in GF(2^n) with Applications to MDS Matrices". In: *CRYPTO 2016, Part I*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9814. LNCS. Springer, Heidelberg, Aug. 2016, pp. 625–653. DOI: 10.1007/978-3-662-53018-4_23 (cit. on pp. 8, 104, 109, 110, 119).

[Bei+16b] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. "The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS". In: *CRYPTO 2016, Part II*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9815. LNCS. Springer, Heidelberg, Aug. 2016, pp. 123–153. DOI: 10.1007/978-3-662-53008-5_5 (cit. on pp. 41, 120, 138, 139).

[Bei+17] Christof Beierle, Anne Canteaut, Gregor Leander, and Yann Rotella. "Proving Resistance Against Invariant Attacks: How to Choose the Round Constants". In: *CRYPTO 2017, Part II*. Ed. by Jonathan Katz and Hovav Shacham. Vol. 10402. LNCS. Springer, Heidelberg, Aug. 2017, pp. 647–678. DOI: 10.1007/978-3-319-63715-0_22 (cit. on pp. 47, 87, 92, 97).

[Bei+19] Christof Beierle, Gregor Leander, Amir Moradi, and Shahram Rasoolzadeh. "CRAFT: Lightweight Tweakable Block Cipher with Efficient Protection Against DFA Attacks". In: *IACR Trans. Symm. Cryptol.* 2019.1 (2019), pp. 5–45. ISSN: 2519-173X. DOI: 10.13154/tosc.v2019.i1.5-45 (cit. on p. 41).

[Bel+94] Mihir Bellare, Joe Kilian, and Phillip Rogaway. "The Security of Cipher Block Chaining". In: *CRYPTO'94*. Ed. by Yvo Desmedt. Vol. 839. LNCS. Springer, Heidelberg, Aug. 1994, pp. 341–358. DOI: 10.1007/3-540-48658-5_32 (cit. on pp. 22, 25).

[Bel+97] Mihir Bellare, Anand Desai, Eric Jokipii, and Phillip Rogaway. "A Concrete Security Treatment of Symmetric Encryption". In: *38th FOCS*. IEEE Computer Society Press, Oct. 1997, pp. 394–403. DOI: 10.1109/SFCS.1997.646128 (cit. on pp. 17, 25).

[Bel+19] Davide Bellizia, Francesco Berti, Olivier Bronchain, Gaëtan Cassiers, Sébastien Duval, Chun Guo, Gregor Leander, Gaëtan Leurent, Itamar Levi, Charles Momin, Olivier Pereira, Thomas Peters, François-Xavier Standaert, and Friedrich Wiemer. *Spook: Sponge-Based Leakage-Resilient Authenticated Encryption with a Masked Tweakable Block Cipher*. Submitted to NIST LWC. 2019 (cit. on pp. 10, 138).

[BF98] T. D. Bending and Dmitry Fon-Der-Flaass. “Crooked Functions, Bent Functions, and Distance Regular Graphs”. In: *Electr. J. Comb.* 5 (1998) (cit. on p. 66).

[Ber+06] Thierry P. Berger, Anne Canteaut, Pascale Charpin, and Yann Laigle-Chapuy. “On Almost Perfect Nonlinear Functions Over $\mathbf{F}_2^n$”. In: *IEEE Trans. Information Theory* 52.9 (2006), pp. 4160–4170 (cit. on pp. 57, 61, 64).

[Ber09] Daniel J. Bernstein. “Optimizing linear maps modulo 2”. In: *Workshop Record of SPEED-CC – Software Performance Enhancement for Encryption and Decryption and Cryptographic Compilers*. 2009, pp. 3–18 (cit. on p. 112).

[BC14] Daniel J. Bernstein and Tung Chou. “Faster Binary-Field Multiplication and Faster Binary-Field MACs”. In: *SAC 2014*. Ed. by Antoine Joux and Amr M. Youssef. Vol. 8781. LNCS. Springer, Heidelberg, Aug. 2014, pp. 92–111. DOI: 10.1007/978-3-319-13051-4_6 (cit. on p. 112).

[Ber+11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. *The Keccak reference*. Available at http://keccak.noekeon.org/. Jan. 2011 (cit. on pp. 132, 138).

[BS91] Eli Biham and Adi Shamir. “Differential Cryptanalysis of DES-like Cryptosystems”. In: *CRYPTO’90*. Ed. by Alfred J. Menezes and Scott A. Vanstone. Vol. 537. LNCS. Springer, Heidelberg, Aug. 1991, pp. 2–21. DOI: 10.1007/3-540-38424-3_1 (cit. on pp. 29, 143).

[Bih+98] Eli Biham, Ross J. Anderson, and Lars R. Knudsen. “Serpent: A New Block Cipher Proposal”. In: *FSE’98*. Ed. by Serge Vaudenay. Vol. 1372. LNCS. Springer, Heidelberg, Mar. 1998, pp. 222–238. DOI: 10.1007/3-540-69710-1_15 (cit. on p. 138).

[Bih+99a] Eli Biham, Alex Biryukov, and Adi Shamir. “Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials”. In: *EUROCRYPT’99*. Ed. by Jacques Stern. Vol. 1592. LNCS. Springer, Heidelberg, May 1999, pp. 12–23. DOI: 10.1007/3-540-48910-X_2 (cit. on pp. 125, 139, 140).

[Bih+99b] Eli Biham, Alex Biryukov, and Adi Shamir. “Miss in the Middle Attacks on IDEA and Khufu”. In: *FSE’99*. Ed. by Lars R. Knudsen. Vol. 1636. LNCS. Springer, Heidelberg, Mar. 1999, pp. 124–138. DOI: 10.1007/3-540-48519-8_10 (cit. on p. 125).

[Bih+02] Eli Biham, Orr Dunkelman, and Nathan Keller. “Enhancing Differential-Linear Cryptanalysis”. In: *ASIACRYPT 2002*. Ed. by Yuliang Zheng. Vol. 2501. LNCS. Springer, Heidelberg, Dec. 2002, pp. 254–266. DOI: 10.1007/3-540-36178-2_16 (cit. on p. 42).

[Bih+05] Eli Biham, Alex Biryukov, and Adi Shamir. “Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials”. In: *Journal of Cryptology* 18.4 (Sept. 2005), pp. 291–311. DOI: 10.1007/s00145-005-0129-3 (cit. on pp. 139, 140).

[Bil+13] Begül Bilgin, Andrey Bogdanov, Miroslav Knežević, Florian Mendel, and Qingju Wang. “Fides: Lightweight Authenticated Cipher with Side-Channel Resistance for Constrained Hardware”. In: *CHES 2013*. Ed. by Guido Bertoni and Jean-Sébastien Coron. Vol. 8086. LNCS. Springer, Heidelberg, Aug. 2013, pp. 142–158. DOI: 10.1007/978-3-642-40349-1_9 (cit. on pp. 41, 60, 120, 134).

[BK09] Alex Biryukov and Dmitry Khovratovich. “Related-Key Cryptanalysis of the Full AES-192 and AES-256”. In: *ASIACRYPT 2009*. Ed. by Mitsuru Matsui. Vol. 5912. LNCS. Springer, Heidelberg, Dec. 2009, pp. 1–18. DOI: 10.1007/978-3-642-10366-7_1 (cit. on p. 22).

[BP15] Alex Biryukov and Léo Perrin. “On Reverse-Engineering S-Boxes with Hidden Design Criteria or Structure”. In: *CRYPTO 2015, Part I*. Ed. by Rosario Gennaro and Matthew J. B. Robshaw. Vol. 9215. LNCS. Springer, Heidelberg, Aug. 2015, pp. 116–140. DOI: 10.1007/978-3-662-47989-6_6 (cit. on p. 94).

[Bir+04] Alex Biryukov, Christophe De Cannieére, Joseph Lano, Siddika Berna Ors, and Bart Preneel. *Security and Performance Analysis of ARIA*. Jan. 2004. URL: https://www.esat.kuleuven.be/cosic/publications/article-500.pdf (cit. on p. 120).

[Bir+09] Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolic. “Distinguisher and Related-Key Attack on the Full AES-256”. In: *CRYPTO 2009*. Ed. by Shai Halevi. Vol. 5677. LNCS. Springer, Heidelberg, Aug. 2009, pp. 231–249. DOI: 10.1007/978-3-642-03356-8_14 (cit. on p. 22).

[Bir+15] Alex Biryukov, Patrick Derbez, and Léo Perrin. “Differential Analysis and Meet-in-the-Middle Attack Against Round-Reduced TWINE”. In: *FSE 2015*. Ed. by Gregor Leander. Vol. 9054. LNCS. Springer, Heidelberg, Mar. 2015, pp. 3–27. DOI: 10.1007/978-3-662-48116-5_1 (cit. on p. 44).

[Bla+02] John Black, Phillip Rogaway, and Thomas Shrimpton. “Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV”. In: *CRYPTO 2002*. Ed. by Moti Yung. Vol. 2442. LNCS. Springer, Heidelberg, Aug. 2002, pp. 320–335. DOI: 10.1007/3-540-45708-9_21 (cit. on p. 22).

[BN13] Céline Blondeau and Kaisa Nyberg. “New Links between Differential and Linear Cryptanalysis”. In: *EUROCRYPT 2013*. Ed. by Thomas Johansson and Phong Q. Nguyen. Vol. 7881. LNCS. Springer, Heidelberg, May 2013, pp. 388–404. DOI: 10.1007/978-3-642-38348-9_24 (cit. on pp. 55, 57).

[BN14] Céline Blondeau and Kaisa Nyberg. "Links between Truncated Differential and Multidimensional Linear Properties of Block Ciphers and Underlying Attack Complexities". In: *EUROCRYPT 2014*. Ed. by Phong Q. Nguyen and Elisabeth Oswald. Vol. 8441. LNCS. Springer, Heidelberg, May 2014, pp. 165–182. DOI: 10.1007/978-3-642-55220-5_10 (cit. on p. 123).

[Blo+15] Céline Blondeau, Gregor Leander, and Kaisa Nyberg. "Differential-Linear Cryptanalysis Revisited". In: *FSE 2014*. Ed. by Carlos Cid and Christian Rechberger. Vol. 8540. LNCS. Springer, Heidelberg, Mar. 2015, pp. 411–430. DOI: 10.1007/978-3-662-46706-0_21 (cit. on pp. 43, 44).

[Blo+17] Céline Blondeau, Gregor Leander, and Kaisa Nyberg. "Differential-Linear Cryptanalysis Revisited". In: *Journal of Cryptology* 30.3 (July 2017), pp. 859–888. DOI: 10.1007/s00145-016-9237-5 (cit. on p. 43).

[Bog+07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. "PRESENT: An Ultra-Lightweight Block Cipher". In: *CHES 2007*. Ed. by Pascal Paillier and Ingrid Verbauwhede. Vol. 4727. LNCS. Springer, Heidelberg, Sept. 2007, pp. 450–466. DOI: 10.1007/978-3-540-74735-2_31 (cit. on pp. 103, 106, 128, 138).

[Bog+12] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, François-Xavier Standaert, John P. Steinberger, and Elmar Tischhauser. "Key-Alternating Ciphers in a Provable Setting: Encryption Using a Small Number of Public Permutations - (Extended Abstract)". In: *EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Vol. 7237. LNCS. Springer, Heidelberg, Apr. 2012, pp. 45–62. DOI: 10.1007/978-3-642-29011-4_5 (cit. on p. 26).

[Bor+12] Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knežević, Lars R. Knudsen, Gregor Leander, Ventzislav Nikov, Christof Paar, Christian Rechberger, Peter Rombouts, Søren S. Thomsen, and Tolga Yalçin. "PRINCE - A Low-Latency Block Cipher for Pervasive Computing Applications - Extended Abstract". In: *ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Vol. 7658. LNCS. Springer, Heidelberg, Dec. 2012, pp. 208–225. DOI: 10.1007/978-3-642-34961-4_14 (cit. on pp. 106, 120, 135).

[Bor+97] Johan Borst, Lars R. Knudsen, and Vincent Rijmen. "Two Attacks on Reduced IDEA". In: *EUROCRYPT'97*. Ed. by Walter Fumy. Vol. 1233. LNCS. Springer, Heidelberg, May 1997, pp. 1–13. DOI: 10.1007/3-540-69053-0_1 (cit. on p. 44).

[Bou+11] Christina Boura, Anne Canteaut, and Christophe De Cannière. "Higher-order differential properties of Keccak and Luffa". In: *FSE 2011*. Ed. by Antoine Joux. Vol. 6733. LNCS. Springer, Heidelberg, Feb. 2011 (cit. on pp. 73, 91).

[Bou+14] Christina Boura, María Naya-Plasencia, and Valentin Suder. "Scrutinizing and Improving Impossible Differential Attacks: Applications to CLEFIA, Camellia, LBlock and Simon". In: *ASIACRYPT 2014, Part I*. Ed. by Palash Sarkar and Tetsu Iwata. Vol. 8873. LNCS. Springer, Heidelberg, Dec. 2014, pp. 179–199. DOI: 10.1007/978-3-662-45611-8_10 (cit. on p. 140).

[Bou+19] Christina Boura, Anne Canteaut, and Daniel Coggia. "A General Proof Framework for Recent AES Distinguishers". In: *IACR Trans. Symm. Cryptol.* 2019.1 (2019), pp. 170–191. ISSN: 2519-173X. DOI: 10.13154/tosc.v2019.i1.170-191 (cit. on p. 142).

[BP10] Joan Boyar and René Peralta. "A New Combinational Logic Minimization Technique with Applications to Cryptology". In: *SEA 2010*. Vol. 6049. LNCS. 2010, pp. 178–189. DOI: 10.1007/978-3-642-13193-6_16 (cit. on pp. 104, 109, 112, 113).

[Boy+08] Joan Boyar, Philip Matthews, and René Peralta. "On the Shortest Linear Straight-Line Program for Computing Linear Forms". In: *MFCS 2008*. Vol. 5162. LNCS. 2008, pp. 168–179. DOI: 10.1007/978-3-540-85238-4_13 (cit. on pp. 104, 109, 111–113).

[Boy+13] Joan Boyar, Philip Matthews, and René Peralta. "Logic Minimization Techniques with Applications to Cryptology". In: *Journal of Cryptology* 26.2 (Apr. 2013), pp. 280–312. DOI: 10.1007/s00145-012-9124-7 (cit. on pp. 104, 109, 111–113, 116).

[Boy+17] Joan Boyar, Magnus Gausdal Find, and René Peralta. *Low-Depth, Low-Size Circuits for Cryptographic Applications*. BFA 2017. 2017 (cit. on pp. 104, 109, 112, 113, 117).

[CC03] Anne Canteaut and Pascale Charpin. "Decomposing bent functions". In: *IEEE Trans. Information Theory* 49.8 (2003), pp. 2004–2019 (cit. on p. 66).

[CR15] Anne Canteaut and Joëlle Roué. "On the Behaviors of Affine Equivalent Sboxes Regarding Differential and Linear Attacks". In: *EUROCRYPT 2015, Part I*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. LNCS. Springer, Heidelberg, Apr. 2015, pp. 45–74. DOI: 10.1007/978-3-662-46800-5_3 (cit. on pp. 32, 39, 73).

[CV02] Anne Canteaut and Marion Videau. "Degree of Composition of Highly Nonlinear Functions and Applications to Higher Order Differential Cryptanalysis". In: *EUROCRYPT 2002*. Ed. by Lars R. Knudsen. Vol. 2332. LNCS. Springer, Heidelberg, 2002, pp. 518–533. DOI: 10.1007/3-540-46035-7_34 (cit. on p. 51).

[CV05] Anne Canteaut and Marion Videau. “Symmetric Boolean functions”. In: *IEEE Trans. Information Theory* 51.8 (2005), pp. 2791–2811. DOI: 10.1109/TIT.2005.851743 (cit. on p. 47).

[Can+08] Anne Canteaut, Pascale Charpin, and Gohar M. Kyureghyan. “A new class of monomial bent functions”. In: *Finite Fields and Their Applications* 14.1 (2008), pp. 221–241. ISSN: 1071-5797. DOI: 10.1016/j.ffa.2007.02.004 (cit. on p. 69).

[Can+18] Anne Canteaut, Virginie Lallemand, Gregor Leander, Patrick Neumann, and Friedrich Wiemer. *BISON – Instantiating the Whitened Swap-Or-Not Construction (Full Version)*. 2018. IACR PREPRINT: 2018/1011 (cit. on pp. 79, 86, 94, 95, 98).

[Can+19a] Anne Canteaut, Virginie Lallemand, Gregor Leander, Patrick Neumann, and Friedrich Wiemer. “BISON – Instantiating the Whitened Swap-Or-Not Construction”. In: *EUROCRYPT 2019, Part III*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11478. LNCS. Springer, Heidelberg, May 2019, pp. 585–616. DOI: 10.1007/978-3-030-17659-4_20. IACR PREPRINT: 2018/1011 (cit. on pp. 6–9, 71, 73).

[Can+19b] Anne Canteaut, Lukas Kölsch, and Friedrich Wiemer. *Observations on the DLCT and Absolute Indicators*. 2019. IACR PREPRINT: 2019/848 (cit. on pp. 6, 10, 55).

[Can+19c] Anne Canteaut, Lukas Kölsch, Chunlei Li, Chao Li, Kangquan Li, Longjiang Qu, and Friedrich Wiemer. *On the Differential-Linear Connectivity Table of Vectorial Boolean Functions*. Submitted to IEEE Transactions on Information Theory. 2019 (cit. on pp. 10, 70).

[Can+19d] Anne Canteaut, Sébastien Duval, Gaëtan Leurent, María Naya-Plasencia, Léo Perrin, Thomas Pornin, and André Schrottenloher. *Saturnin: a suite of lightweight symmetric algorithms for post-quantum security*. Submitted to NIST LWC. 2019 (cit. on p. 142).

[Car08] Claude Carlet. “Recursive Lower Bounds on the Nonlinearity Profile of Boolean Functions and Their Applications”. In: *IEEE Transactions on Information Theory* 54.3 (Mar. 2008), pp. 1262–1272. ISSN: 0018-9448. DOI: 10.1109/TIT.2007.915704 (cit. on pp. 68, 69).

[Car10a] Claude Carlet. “Boolean Functions for Cryptography and Error-Correcting Codes”. In: *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*. Ed. by Yves Crama and Peter L. Hammer. Cambridge University Press, 2010, pp. 257–397. DOI: 10.1017/CBO9780511780448.011 (cit. on pp. 45, 50, 51, 56, 63).

[Car10b] Claude Carlet. “Vectorial Boolean Functions for Cryptography”. In: *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*. Ed. by Yves Crama and Peter L. Hammer. Cambridge University Press, 2010, pp. 398–469. DOI: 10.1017/CBO9780511780448.011 (cit. on p. 45).

[Car+98] Claude Carlet, Pascale Charpin, and Victor A. Zinoviev. “Codes, Bent Functions and Permutations Suitable For DES-like Cryptosystems”. In: *Designs, Codes and Cryptography* 15.2 (1998), pp. 125–156. DOI: 10.1023/A:1008344232130 (cit. on p. 67).

[CV95] Florent Chabaud and Serge Vaudenay. “Links Between Differential and Linear Cryptanalysis”. In: *EUROCRYPT’94*. Ed. by Alfredo De Santis. Vol. 950. LNCS. Springer, Heidelberg, May 1995, pp. 356–365. DOI: 10.1007/BFb0053450 (cit. on pp. 50, 55, 57).

[Cha+07] Pascale Charpin, Tor Helleseth, and Victor A. Zinoviev. “Propagation characteristics of $x \mapsto x^{-1}$ and Kloosterman sums”. In: *Finite Fields and Their Applications* 13.2 (2007), pp. 366–381. DOI: 10.1016/j.ffa.2005.08.007 (cit. on pp. 63, 65).

[Cho+12] Jiali Choy, Huihui Yap, Khoongming Khoo, Jian Guo, Thomas Peyrin, Axel Poschmann, and Chik How Tan. “SPN-Hash: Improving the Provable Resistance against Differential Collision Attacks”. In: *AFRICACRYPT 12*. Ed. by Aikaterini Mitrokotsa and Serge Vaudenay. Vol. 7374. LNCS. Springer, Heidelberg, July 2012, pp. 270–286 (cit. on p. 108).

[Cid+05] Carlos Cid, Sean Murphy, and Matthew J. B. Robshaw. “Small Scale Variants of the AES”. In: *FSE 2005*. Ed. by Henri Gilbert and Helena Handschuh. Vol. 3557. LNCS. Springer, Heidelberg, Feb. 2005, pp. 145–162. DOI: 10.1007/11502760_10 (cit. on p. 120).

[Cui+17] Ting Cui, Chenhui Jin, Bin Zhang, Zhuo Chen, and Guoshuang Zhang. “Searching all truncated impossible differentials in SPN”. In: *IET Information Security* 11.2 (2017), pp. 89–96. DOI: 10.1049/iet-ifs.2015.0052 (cit. on p. 126).

[Dae93] Joan Daemen. “Limitations of the Even-Mansour Construction (Rump Session)”. In: *ASIACRYPT’91*. Ed. by Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto. Vol. 739. LNCS. Springer, Heidelberg, Nov. 1993, pp. 495–498. DOI: 10.1007/3-540-57332-1_46 (cit. on p. 26).

[Dae95] Joan Daemen. “Cipher and Hash Function Design Strategies based on linear and differential cryptanalysis”. PhD thesis. Katholieke Universiteit Leuven, Mar. 1995 (cit. on pp. 32, 36, 40, 41, 106).

[DR98] Joan Daemen and Vincent Rijmen. “The Block Cipher Rijndael”. In: *CARDIS'98*. Vol. 1820. LNCS. Springer, 1998, pp. 277–284. DOI: 10.1007/10721064_26 (cit. on p. 22).

[DR01] Joan Daemen and Vincent Rijmen. “The Wide Trail Design Strategy”. In: *8th IMA International Conference on Cryptography and Coding*. Ed. by Bahram Honary. Vol. 2260. LNCS. Springer, Heidelberg, Dec. 2001, pp. 222–238 (cit. on p. 24).

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002. ISBN: 3-540-42580-2. DOI: 10.1007/978-3-662-04722-4 (cit. on pp. 24, 31, 32, 36, 41, 106, 120, 135).

[DR06] Joan Daemen and Vincent Rijmen. “Understanding Two-Round Differentials in AES”. In: *SCN 06*. Ed. by Roberto De Prisco and Moti Yung. Vol. 4116. LNCS. Springer, Heidelberg, Sept. 2006, pp. 78–94. DOI: 10.1007/11832072_6 (cit. on p. 73).

[Dae+93] Joan Daemen, René Govaerts, and Joos Vandewalle. “Block ciphers based on Modular Arithmetic”. In: *State and Progress in the Research of Cryptography*. Ed. by W. Wolfowicz. Fondazione Ugo Bordoni, 1993, pp. 80–89 (cit. on p. 24).

[Dae+95] Joan Daemen, René Govaerts, and Joos Vandewalle. “Correlation Matrices”. In: *FSE'94*. Ed. by Bart Preneel. Vol. 1008. LNCS. Springer, Heidelberg, Dec. 1995, pp. 275–285. DOI: 10.1007/3-540-60590-8_21 (cit. on p. 36).

[Dae+97] Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. “The Block Cipher Square”. In: *FSE'97*. Ed. by Eli Biham. Vol. 1267. LNCS. Springer, Heidelberg, Jan. 1997, pp. 149–165. DOI: 10.1007/BFb0052343 (cit. on pp. 106, 120).

[DF16] Patrick Derbez and Pierre-Alain Fouque. “Automatic Search of Meet-in-the-Middle and Impossible Differential Attacks”. In: *CRYPTO 2016, Part II*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9815. LNCS. Springer, Heidelberg, Aug. 2016, pp. 157–184. DOI: 10.1007/978-3-662-53008-5_6 (cit. on pp. 126, 140).

[Der+13] Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. “Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting”. In: *EUROCRYPT 2013*. Ed. by Thomas Johansson and Phong Q. Nguyen. Vol. 7881. LNCS. Springer, Heidelberg, May 2013, pp. 371–387. DOI: 10.1007/978-3-642-38348-9_23 (cit. on p. 22).

[Dil72] John F Dillon. “A survey of bent functions”. In: *The NSA technical journal* 191 (1972), p. 215 (cit. on p. 51).

[Dil99] John F. Dillon. “Multiplicative Difference Sets via Additive Characters”. In: *Designs, Codes and Cryptography* 17.1 (Sept. 1999), pp. 225–235. ISSN: 1573-7586. DOI: 10.1023/A:1026435428030 (cit. on p. 69).

[Din+16] Daniel Dinu, Léo Perrin, Aleksei Udovenko, Vesselin Velichkov, Johann Großschädl, and Alex Biryukov. “Design Strategies for ARX with Provable Bounds: Sparx and LAX”. In: *ASIACRYPT 2016, Part I*. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Vol. 10031. LNCS. Springer, Heidelberg, Dec. 2016, pp. 484–513. DOI: 10.1007/978-3-662-53887-6_18 (cit. on p. 40).

[Din14] Itai Dinur. “Improved Differential Cryptanalysis of Round-Reduced Speck”. In: *SAC 2014*. Ed. by Antoine Joux and Amr M. Youssef. Vol. 8781. LNCS. Springer, Heidelberg, Aug. 2014, pp. 147–164. DOI: 10.1007/978-3-319-13051-4_9 (cit. on p. 144).

[Dob95] Hans Dobbertin. “Construction of Bent Functions and Balanced Boolean Functions with High Nonlinearity”. In: *FSE'94*. Ed. by Bart Preneel. Vol. 1008. LNCS. Springer, Heidelberg, Dec. 1995, pp. 61–74. DOI: 10.1007/3-540-60590-8_5 (cit. on p. 65).

[Dob+15] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. “Cryptanalysis of Ascon”. In: *CT-RSA 2015*. Ed. by Kaisa Nyberg. Vol. 9048. LNCS. Springer, Heidelberg, Apr. 2015, pp. 371–387. DOI: 10.1007/978-3-319-16715-2_20 (cit. on p. 43).

[Dob+16] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. *Ascon v1. 2*. Submission to the CAESAR Competition. Available at http://ascon.iaik.tugraz.at/files/asconv12.pdf. 2016 (cit. on p. 138).

[Dun+10] Orr Dunkelman, Nathan Keller, and Adi Shamir. “Improved Single-Key Attacks on 8-Round AES-192 and AES-256”. In: *ASIACRYPT 2010*. Ed. by Masayuki Abe. Vol. 6477. LNCS. Springer, Heidelberg, Dec. 2010, pp. 158–176. DOI: 10.1007/978-3-642-17373-8_10 (cit. on p. 22).

[DL18] Sébastien Duval and Gaëtan Leurent. “MDS Matrices with Lightweight Circuits”. In: *IACR Trans. Symm. Cryptol.* 2018.2 (2018), pp. 48–78. ISSN: 2519-173X. DOI: 10.13154/tosc.v2018.i2.48-78 (cit. on pp. 117, 118).

[EM97] Shimon Even and Yishay Mansour. “A Construction of a Cipher from a Single Pseudorandom Permutation”. In: *Journal of Cryptology* 10.3 (June 1997), pp. 151–162. DOI: 10.1007/s001459900025 (cit. on p. 26).

[Eve88] Jan-Hendrik Evertse. “Linear Structures in Blockciphers”. In: *EUROCRYPT’87*. Ed. by David Chaum and Wyn L. Price. Vol. 304. LNCS. Springer, Heidelberg, Apr. 1988, pp. 249–266. DOI: 10.1007/3-540-39118-5_23 (cit. on p. 47).

[Fei71] Horst Feistel. *Block Cipher Cryptographic System*. US Patent 3,798,359. June 1971 (cit. on p. 24).

[Fer+01] Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Michael Stay, David Wagner, and Doug Whiting. “Improved Cryptanalysis of Rijndael”. In: *FSE 2000*. Ed. by Bruce Schneier. Vol. 1978. LNCS. Springer, Heidelberg, Apr. 2001, pp. 213–230. DOI: 10.1007/3-540-44706-7_15 (cit. on p. 22).

[Fil+06] Décio Luiz Gazzoni Filho, Paulo S L M Barreto, and Vincent Rijmen. *The* MAELSTROM-0 *Hash Function*. 2006 (cit. on p. 120).

[FSK10] Carsten Fuhs and Peter Schneider-Kamp. “Optimizing the AES S-Box using SAT”. In: *IWIL@LPAR*. Ed. by Geoff Sutcliffe, Stephan Schulz, and Eugenia Ternovska. Vol. 2. EPiC Series in Computing. EasyChair, 2010, pp. 64–70 (cit. on pp. 104, 109, 111).

[FS10] Carsten Fuhs and Peter Schneider-Kamp. “Synthesizing Shortest Linear Straight-Line Programs over GF(2) Using SAT”. In: *SAT*. Vol. 6175. LNCS. Springer, 2010, pp. 71–84. DOI: 10.1007/978-3-642-14186-7_8 (cit. on pp. 104, 109, 111).

[GOST15] *(GOST R 34.12-2015) Information Technology – Cryptographic Data Security – Block Ciphers*. Federal Agency on Technical Regulation and Metrology (GOST), 2015 (cit. on p. 135).

[Gan+06] Sugata Gangopadhyay, Pradipkumar H. Keskar, and Subhamoy Maitra. “Patterson-Wiedemann construction revisited”. In: *Discrete Mathematics* 306.14 (2006), pp. 1540–1556. DOI: 10.1016/j.disc.2005.06.033 (cit. on p. 61).

[Gau+] Praveen Gauravaram, Lars R. Knudsen, Krystian Matusiewicz, Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. *Grøstl – a SHA-3 candidate*. Submitted to SHA-3 (cit. on p. 120).

[GM00] Henri Gilbert and Marine Minier. “A Collision Attack on 7 Rounds of Rijndael”. In: *AES Candidate Conference*. Vol. 230. 2000, p. 241 (cit. on p. 22).

[GK04] Guang Gong and Khoongming Khoo. “Additive Autocorrelation of Resilient Boolean Functions”. In: *SAC 2003*. Ed. by Mitsuru Matsui and Robert J. Zuccherato. Vol. 3006. LNCS. Springer, Heidelberg, Aug. 2004, pp. 275–290. DOI: 10.1007/978-3-540-24654-1_20 (cit. on p. 69).

[Gon+11] Zheng Gong, Svetla Nikova, and Yee Wei Law. “KLEIN: A new family of lightweight block ciphers”. In: *RFIDSec’11*. Ed. by Ari Juels and Christof Paar. Vol. 7055. LNCS. Springer, Heidelberg, 2011, pp. 1–18 (cit. on p. 135).

[Gra17] Lorenzo Grassi. *Structural Truncated Differential Attacks on round-reduced AES*. Cryptology ePrint Archive, Report 2017/832. http://eprint.iacr.org/2017/832. 2017 (cit. on p. 123).

[GR16] Lorenzo Grassi and Christian Rechberger. “Practical Low Data-Complexity Subspace-Trail Cryptanalysis of Round-Reduced PRINCE”. In: *INDOCRYPT 2016*. Ed. by Orr Dunkelman and Somitra Kumar Sanadhya. Vol. 10095. LNCS. Springer, Heidelberg, Dec. 2016, pp. 322–342. DOI: 10.1007/978-3-319-49890-4_18 (cit. on p. 125).

[Gra+16] Lorenzo Grassi, Christian Rechberger, and Sondre Rønjom. “Subspace Trail Cryptanalysis and its Applications to AES”. In: *IACR Trans. Symm. Cryptol.* 2016.2 (2016). http://tosc.iacr.org/index.php/ToSC/article/view/571, pp. 192–225. ISSN: 2519-173X. DOI: 10.13154/tosc.v2016.i2.192-225 (cit. on pp. 9, 22, 44, 86, 123, 125, 128, 142).

[Gra+17] Lorenzo Grassi, Christian Rechberger, and Sondre Rønjom. “A New Structural-Differential Property of 5-Round AES”. In: *EUROCRYPT 2017, Part II*. Ed. by Jean-Sébastien Coron and Jesper Buus Nielsen. Vol. 10211. LNCS. Springer, Heidelberg, 2017, pp. 289–317. DOI: 10.1007/978-3-319-56614-6_10 (cit. on pp. 9, 22, 124, 125, 134, 142).

[Gra+19] Lorenzo Grassi, Gregor Leander, Christian Rechberger, Cihangir Tezcan, and Friedrich Wiemer. *Weak-Key Subspace Trails and Applications to AES*. 2019. IACR PREPRINT: 2019/852 (cit. on p. 141).

[GL15] Chun Guo and Dongdai Lin. “On the Indifferentiability of Key-Alternating Feistel Ciphers with No Key Derivation”. In: *TCC 2015, Part I*. Ed. by Yevgeniy Dodis and Jesper Buus Nielsen. Vol. 9014. LNCS. Springer, Heidelberg, Mar. 2015, pp. 110–133. DOI: 10.1007/978-3-662-46494-6_6 (cit. on p. 26).

[Guo+11a] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. “The LED Block Cipher”. In: *CHES 2011*. Ed. by Bart Preneel and Tsuyoshi Takagi. Vol. 6917. LNCS. Springer, Heidelberg, 2011, pp. 326–341. DOI: 10.1007/978-3-642-23951-9_22 (cit. on pp. 103, 113).

[Guo+11b] Jian Guo, Thomas Peyrin, and Axel Poschmann. “The PHOTON Family of Lightweight Hash Functions”. In: *CRYPTO 2011*. Ed. by Phillip Rogaway. Vol. 6841. LNCS. Springer, Heidelberg, Aug. 2011, pp. 222–239. DOI: 10.1007/978-3-642-22792-9_13 (cit. on pp. 103, 113).

[GR13] Kishan Chand Gupta and Indranil Ghosh Ray. “On Constructions of Involutory MDS Matrices”. In: *AFRICACRYPT 13*. Ed. by Amr Youssef, Abderrahmane Nitaj, and Aboul Ella Hassanien. Vol. 7918. LNCS. Springer, Heidelberg, June 2013, pp. 43–60. DOI: 10.1007/978-3-642-38553-7_3 (cit. on pp. 106, 107).

[Har+95] Carlo Harpes, Gerhard G. Kramer, and James L. Massey. “A Generalization of Linear Cryptanalysis and the Applicability of Matsui’s Piling-Up Lemma”. In: *EUROCRYPT’95*. Ed. by Louis C. Guillou and Jean-Jacques Quisquater. Vol. 921. LNCS. Springer, Heidelberg, May 1995, pp. 24–38. DOI: 10.1007/3-540-49264-X_3 (cit. on p. 28).

[Hel+06] Martin Hell, Thomas Johansson, Alexander Maximov, and Willi Meier. “A Stream Cipher Proposal: Grain-128”. In: *ISIT*. IEEE, 2006, pp. 1614–1618. DOI: 10.1109/ISIT.2006.261549 (cit. on p. 51).

[HT16] Viet Tung Hoang and Stefano Tessaro. “Key-Alternating Ciphers and Key-Length Extension: Exact Bounds and Multi-user Security”. In: *CRYPTO 2016, Part I*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9814. LNCS. Springer, Heidelberg, Aug. 2016, pp. 3–32. DOI: 10.1007/978-3-662-53018-4_1 (cit. on p. 26).

[Hoa+12] Viet Tung Hoang, Ben Morris, and Phillip Rogaway. “An Enciphering Scheme Based on a Card Shuffle”. In: *CRYPTO 2012*. Ed. by Reihaneh Safavi-Naini and Ran Canetti. Vol. 7417. LNCS. Springer, Heidelberg, Aug. 2012, pp. 1–13. DOI: 10.1007/978-3-642-32009-5_1 (cit. on pp. 27, 71, 72).

[Hon+01] Seokhie Hong, Sangjin Lee, Jongin Lim, Jaechul Sung, Dong Hyeon Cheon, and Inho Cho. “Provable Security against Differential and Linear Cryptanalysis for the SPN Structure”. In: *FSE 2000*. Ed. by Bruce Schneier. Vol. 1978. LNCS. Springer, Heidelberg, Apr. 2001, pp. 273–283. DOI: 10.1007/3-540-44706-7_19 (cit. on pp. 32, 39).

[JK97] Thomas Jakobsen and Lars R. Knudsen. “The Interpolation Attack on Block Ciphers”. In: *FSE’97*. Ed. by Eli Biham. Vol. 1267. LNCS. Springer, Heidelberg, Jan. 1997, pp. 28–40. DOI: 10.1007/BFb0052332 (cit. on p. 73).

[Jea+14] Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. *Joltik*. Submission to the CAESAR competition. 2014 (cit. on p. 120).

[Jea+17a] Jérémy Jean, Amir Moradi, Thomas Peyrin, and Pascal Sasdrich. “Bit-Sliding: A Generic Technique for Bit-Serial Implementations of SPN-based Primitives - Applications to AES, PRESENT and SKINNY”. In: *CHES 2017*. Ed. by Wieland Fischer and Naofumi Homma. Vol. 10529. LNCS. Springer, Heidelberg, Sept. 2017, pp. 687–707. DOI: 10.1007/978-3-319-66787-4_33 (cit. on pp. 104, 111, 113, 120).

[Jea+17b] Jérémy Jean, Thomas Peyrin, Siang Meng Sim, and Jade Tourteaux. “Optimizing Implementations of Lightweight Building Blocks”. In: *IACR Trans. Symm. Cryptol.* 2017.4 (2017), pp. 130–168. ISSN: 2519-173X. DOI: 10.13154/tosc.v2017.i4.130-168 (cit. on pp. 104, 108–110, 113, 114, 116, 119, 120).

[JA09] Jorge Nakahara Jr. and Élcio Abrahão. “A New Involutory MDS Matrix for the AES”. In: *I. J. Network Security* 9.2 (2009), pp. 109–116. URL: http://ijns.femto.com.tw/contents/ijns-v9-n2/ijns-2009-v9-n2-p109-116.pdf (cit. on p. 107).

[JV04] Pascal Junod and Serge Vaudenay. “FOX: A New Family of Block Ciphers”. In: *SAC 2004*. Ed. by Helena Handschuh and Anwar Hasan. Vol. 3357. LNCS. Springer, Heidelberg, Aug. 2004, pp. 114–129. DOI: 10.1007/978-3-540-30564-4_8 (cit. on p. 120).

[KL08] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Cryptography and Network Security. CRC, 2008. ISBN: 1-58488-551-3 (cit. on pp. 17, 20).

[Kav16] Selçuk Kavut. “Correction to the paper: Patterson-Wiedemann construction revisited”. In: *Discrete Applied Mathematics* 202 (2016), pp. 185–187. ISSN: 0166-218X. DOI: 10.1016/j.dam.2015.07.044 (cit. on p. 61).

[Kav+07] Selçuk Kavut, Subhamoy Maitra, and Melek D. Yücel. “Search for Boolean Functions With Excellent Profiles in the Rotation Symmetric Class”. In: *IEEE Trans. Information Theory* 53.5 (2007), pp. 1743–1751. DOI: 10.1109/TIT.2007.894696 (cit. on pp. 61, 99).

[Kav+19] Selçuk Kavut, Subhamoy Maitra, and Deng Tang. “Construction and search of balanced Boolean functions on even number of variables towards excellent autocorrelation profile”. In: *Designs, Codes and Cryptogrography* 87.2–3 (2019), pp. 261–276. DOI: 10.1007/s10623-018-0522-1 (cit. on p. 61).

[KS07] Liam Keliher and Jiayuan Sui. “Exact maximum expected differential and linear probability for two-round Advanced Encryption Standard”. In: *IET Information Security* 1.2 (2007), pp. 53–57. DOI: 10.1049/iet-ifs:20060161 (cit. on p. 73).

[Ker83] Auguste Kerckhoffs. “La cryptographie militaire”. In: *Journal des sciences militaires* 9 (Jan. 1883), pp. 161–191 (cit. on p. 16).

[Kho+14] Khoongming Khoo, Thomas Peyrin, Axel York Poschmann, and Huihui Yap. “FOAM: Searching for Hardware-Optimal SPN Structures and Components with a Fair Comparison”. In: *CHES 2014*. Ed. by Lejla Batina and Matthew Robshaw. Vol. 8731. LNCS. Springer, Heidelberg, Sept. 2014, pp. 433–450. DOI: 10.1007/978-3-662-44709-3_24 (cit. on pp. 103, 108–110).

[Kim+03] Jongsung Kim, Seokhie Hong, Jaechul Sung, Changhoon Lee, and Sangjin Lee. “Impossible Differential Cryptanalysis for Block Cipher Structures”. In: *INDOCRYPT 2003*. Ed. by Thomas Johansson and Subhamoy Maitra. Vol. 2904. LNCS. Springer, Heidelberg, Dec. 2003, pp. 82–96 (cit. on p. 126).

[Knu95] Lars R. Knudsen. “Truncated and Higher Order Differentials”. In: *FSE’94*. Ed. by Bart Preneel. Vol. 1008. LNCS. Springer, Heidelberg, Dec. 1995, pp. 196–211. DOI: 10.1007/3-540-60590-8_16 (cit. on pp. 44, 123).

[KB96] Lars R. Knudsen and Thomas A. Berson. “Truncated Differentials of SAFER”. In: *FSE’96*. Ed. by Dieter Gollmann. Vol. 1039. LNCS. Springer, Heidelberg, Feb. 1996, pp. 15–26. DOI: 10.1007/3-540-60865-6_38 (cit. on p. 44).

[Knu+99] Lars R. Knudsen, Matthew J. B. Robshaw, and David Wagner. “Truncated Differentials and Skipjack”. In: *CRYPTO’99*. Ed. by Michael J. Wiener. Vol. 1666. LNCS. Springer, Heidelberg, Aug. 1999, pp. 165–180. DOI: 10.1007/3-540-48405-1_11 (cit. on pp. 44, 123).

[Knu98] Lars Knudsen. *DEAL - A 128-bit Block Cipher*. AES Submission. 1998 (cit. on p. 125).

[Köl+16] Stefan Kölbl, Martin M. Lauridsen, Florian Mendel, and Christian Rechberger. “Haraka v2 - Efficient Short-Input Hashing for Post-Quantum Applications”. In: *IACR Trans. Symm. Cryptol.* 2016.2 (2016). http://tosc.iacr.org/index.php/ToSC/article/view/563, pp. 1–29. ISSN: 2519-173X. DOI: 10.13154/tosc.v2016.i2.1-29 (cit. on p. 41).

[Köl19] Lukas Kölsch. “XOR-Counts and Lightweight Multiplication with Fixed Elements in Binary Finite Fields”. In: *EUROCRYPT 2019, Part I*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11476. LNCS. Springer, Heidelberg, May 2019, pp. 285–312. DOI: 10.1007/978-3-030-17653-2_10 (cit. on p. 8).

[Kra+17a] Thorsten Kranz, Gregor Leander, and Friedrich Wiemer. “Linear Cryptanalysis: Key Schedules and Tweakable Block Ciphers”. In: *IACR Trans. Symm. Cryptol.* 2017.1 (2017), pp. 474–505. ISSN: 2519-173X. DOI: 10.13154/tosc.v2017.i1.474-505. IACR PREPRINT: 2017/154 (cit. on pp. 9, 31, 34).

[Kra+17b] Thorsten Kranz, Gregor Leander, Ko Stoffelen, and Friedrich Wiemer. “Shorter Linear Straight-Line Programs for MDS Matrices”. In: *IACR Trans. Symm. Cryptol.* 2017.4 (2017), pp. 188–211. ISSN: 2519-173X. DOI: 10.13154/tosc.v2017.i4.188-211. IACR PREPRINT: 2017/1151 (cit. on pp. 6, 8, 9, 103, 114, 116).

[Kwo+03] Daesung Kwon, Jaesung Kim, Sangwoo Park, Soo Hak Sung, Yaekwon Sohn, Jung Hwan Song, Yongjin Yeom, E-Joong Yoon, Sangjin Lee, Jaewon Lee, Seongtaek Chee, Daewan Han, and Jin Hong. “New Block Cipher: ARIA”. In: *ICISC*. Vol. 2971. LNCS. Springer, 2003, pp. 432–445. DOI: 10.1007/978-3-540-24691-6_32 (cit. on p. 120).

[LF04] Jérôme Lacan and Jérôme Fimes. “Systematic MDS erasure codes based on Vandermonde matrices”. In: *IEEE Communications Letters* 8.9 (2004), pp. 570–572. DOI: 10.1109/LCOMM.2004.833807 (cit. on p. 106).

[Lai95] Xuejia Lai. “Additive and Linear Structures of Cryptographic Functions”. In: *FSE’94*. Ed. by Bart Preneel. Vol. 1008. LNCS. Springer, Heidelberg, Dec. 1995, pp. 75–85. DOI: 10.1007/3-540-60590-8_6 (cit. on p. 47).

[Lai+91] Xuejia Lai, James L. Massey, and Sean Murphy. “Markov Ciphers and Differential Cryptanalysis”. In: *EUROCRYPT’91*. Ed. by Donald W. Davies. Vol. 547. LNCS. Springer, Heidelberg, Apr. 1991, pp. 17–38. DOI: 10.1007/3-540-46416-6_2 (cit. on p. 31).

[LN15] Virginie Lallemand and María Naya-Plasencia. “Cryptanalysis of KLEIN”. In: *FSE 2014*. Ed. by Carlos Cid and Christian Rechberger. Vol. 8540. LNCS. Springer, Heidelberg, Mar. 2015, pp. 451–470. DOI: 10.1007/978-3-662-46706-0_23 (cit. on p. 44).

[LR17a] Virginie Lallemand and Shahram Rasoolzadeh. “Differential Cryptanalysis of 18-Round PRIDE”. In: *INDOCRYPT 2017*. Ed. by Arpita Patra and Nigel P. Smart. Vol. 10698. LNCS. Springer, Heidelberg, Dec. 2017, pp. 126–146 (cit. on pp. 28, 143).

[LS15] Rodolphe Lampe and Yannick Seurin. “Security Analysis of Key-Alternating Feistel Ciphers”. In: *FSE 2014*. Ed. by Carlos Cid and Christian Rechberger. Vol. 8540. LNCS. Springer, Heidelberg, Mar. 2015, pp. 243–264. DOI: 10.1007/978-3-662-46706-0_13 (cit. on p. 26).

[Lan95] Susan K. Langford. “Differential-Linear Cryptanalysis and Threshold Signatures”. PhD thesis. Stanford University, 1995 (cit. on p. 42).

[LH94] Susan K. Langford and Martin E. Hellman. “Differential-Linear Cryptanalysis”. In: *CRYPTO’94*. Ed. by Yvo Desmedt. Vol. 839. LNCS. Springer, Heidelberg, Aug. 1994, pp. 17–25. DOI: 10.1007/3-540-48658-5_3 (cit. on p. 42).

[Lea+11] Gregor Leander, Mohamed Ahmed Abdelraheem, Hoda AlKhzaimi, and Erik Zenner. “A Cryptanalysis of PRINTcipher: The Invariant Subspace Attack”. In: *CRYPTO 2011*. Ed. by Phillip Rogaway. Vol. 6841. LNCS. Springer, Heidelberg, Aug. 2011, pp. 206–221. DOI: 10.1007/978-3-642-22792-9_12 (cit. on p. 86).

[Lea+15] Gregor Leander, Brice Minaud, and Sondre Rønjom. “A Generic Approach to Invariant Subspace Attacks: Cryptanalysis of Robin, iSCREAM and Zorro”. In: *EUROCRYPT 2015, Part I*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. LNCS. Springer, Heidelberg, Apr. 2015, pp. 254–283. DOI: 10.1007/978-3-662-46800-5_11 (cit. on p. 140).

[Lea+18] Gregor Leander, Cihangir Tezcan, and Friedrich Wiemer. “Searching for Subspace Trails and Truncated Differentials”. In: *IACR Trans. Symm. Cryptol.* 2018.1 (2018), pp. 74–100. ISSN: 2519-173X. DOI: 10.13154/tosc.v2018.i1.74-100 (cit. on pp. 7, 9, 10, 123, 125).

[LW17] Chaoyun Li and Qingju Wang. “Design of Lightweight Linear Diffusion Layers from Near-MDS Matrices”. In: *IACR Trans. Symm. Cryptol.* 2017.1 (2017), pp. 129–155. ISSN: 2519-173X. DOI: 10.13154/tosc.v2017.i1.129-155 (cit. on pp. 104, 109).

[Li+19a] Kangquan Li, Chunlei Li, Chao Li, and Longjiang Qu. *On the Differential Linear Connectivity Table of Vectorial Boolean Functions*. 2019. arXiv: 1907.05986 [cs.IT] (cit. on pp. 10, 70).

[Li+19b] Shun Li, Siwei Sun, Chaoyun Li, Zihao Wei, and Lei Hu. “Constructing Low-latency Involutory MDS Matrices with Lightweight Circuits”. In: *IACR Trans. Symm. Cryptol.* 2019.1 (2019), pp. 84–117. ISSN: 2519-173X. DOI: 10.13154/tosc.v2019.i1.84-117 (cit. on p. 118).

[LW19] Yanbin Li and Meiqin Wang. “Cryptanalysis of MORUS”. In: *Designs, Codes and Cryptography* 87 (5 May 2019), pp. 1035–1058. ISSN: 1573-7586. DOI: 10.1007/s10623-018-0501-6 (cit. on p. 41).

[LW16] Yongqiang Li and Mingsheng Wang. “On the Construction of Lightweight Circulant Involutory MDS Matrices”. In: *FSE 2016*. Ed. by Thomas Peyrin. Vol. 9783. LNCS. Springer, Heidelberg, Mar. 2016, pp. 121–139. DOI: 10.1007/978-3-662-52993-5_7 (cit. on pp. 104, 107, 109, 110, 113, 116, 119).

[LS16] Meicheng Liu and Siang Meng Sim. “Lightweight MDS Generalized Circulant Matrices”. In: *FSE 2016*. Ed. by Thomas Peyrin. Vol. 9783. LNCS. Springer, Heidelberg, Mar. 2016, pp. 101–120. DOI: 10.1007/978-3-662-52993-5_6 (cit. on pp. 104, 109, 110, 116, 119).

[LR17b] Yunwen Liu and Vincent Rijmen. *New Observations on Invariant Subspace Attack*. Cryptology ePrint Archive, Report 2017/278. http://eprint.iacr.org/2017/278. 2017 (cit. on pp. 125, 140).

[Lu+08] Jiqiang Lu, Jongsung Kim, Nathan Keller, and Orr Dunkelman. “Improving the Efficiency of Impossible Differential Cryptanalysis of Reduced Camellia and MISTY1”. In: *CT-RSA 2008*. Ed. by Tal Malkin. Vol. 4964. LNCS. Springer, Heidelberg, Apr. 2008, pp. 370–386. DOI: 10.1007/978-3-540-79263-5_24 (cit. on p. 145).

[Luo+09] Yiyuan Luo, Zhongming Wu, Xuejia Lai, and Guang Gong. *A Unified Method for Finding Impossible Differentials of Block Cipher Structures*. Cryptology ePrint Archive, Report 2009/627. http://eprint.iacr.org/2009/627. 2009 (cit. on p. 126).

[Luo+14] Yiyuan Luo, Xuejia Lai, Zhongming Wu, and Guang Gong. “A unified method for finding impossible differentials of block cipher structures”. In: *Inf. Sci.* 263 (2014), pp. 211–220. DOI: 10.1016/j.ins.2013.08.051 (cit. on p. 126).

[MS77] Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of Error-Correcting Codes*. North-Holland Publishing Company, 1977 (cit. on p. 105).

[MS02] Subhamoy Maitra and Palash Sarkar. “Modifications of Patterson-Wiedemann functions for cryptographic applications”. In: *IEEE Trans. Information Theory* 48.1 (2002), pp. 278–284. DOI: 10.1109/18.971756 (cit. on p. 61).

[MT14] Rusydi H. Makarim and Cihangir Tezcan. “Relating Undisturbed Bits to Other Properties of Substitution Boxes”. In: *LightSec 2014*. Ed. by Thomas Eisenbarth and Erdinç Öztürk. Vol. 8898. LNCS. Springer, 2014, pp. 109–125. DOI: 10.1007/978-3-319-16363-5_7 (cit. on pp. 60, 125, 126).

[#24873] Rusydi H. Makarim and Friedrich Wiemer. *Implement algebraic_degree() for Boolean functions*. #24872. 2018 (cit. on p. 11).

[#21252] Rusydi H. Makarim and Friedrich Wiemer. *Computing nonlinear invariants in mq.SBox*. #21252. 2019 (cit. on p. 10).

[Mas94] James L. Massey. “SAFER K-64: A Byte-Oriented Block-Ciphering Algorithm”. In: *FSE’93*. Ed. by Ross J. Anderson. Vol. 809. LNCS. Springer, Heidelberg, Dec. 1994, pp. 1–17. DOI: 10.1007/3-540-58108-1_1 (cit. on pp. 60, 134).

[Mat94] Mitsuru Matsui. “Linear Cryptanalysis Method for DES Cipher”. In: *EUROCRYPT’93*. Ed. by Tor Helleseth. Vol. 765. LNCS. Springer, Heidelberg, May 1994, pp. 386–397. DOI: 10.1007/3-540-48285-7_33 (cit. on pp. 34, 35).

[Mat95] Mitsuru Matsui. “On Correlation Between the Order of S-boxes and the Strength of DES”. In: *EUROCRYPT’94*. Ed. by Alfredo De Santis. Vol. 950. LNCS. Springer, Heidelberg, May 1995, pp. 366–375. DOI: 10.1007/BFb0053451 (cit. on pp. 36, 40).

[McE72] Robert J. McEliece. “Weight congruences for p-ary cyclic codes”. In: *Discrete Mathematics* 3.1–3 (1972), pp. 177–192. DOI: 10.1016/0012-365X(72)90032-5 (cit. on p. 63).

[MS90] Willi Meier and Othmar Staffelbach. “Nonlinearity Criteria for Cryptographic Functions”. In: *EUROCRYPT’89*. Ed. by Jean-Jacques Quisquater and Joos Vandewalle. Vol. 434. LNCS. Springer, Heidelberg, Apr. 1990, pp. 549–562. DOI: 10.1007/3-540-46885-4_53 (cit. on p. 51).

[MM82] Carl H. Meyer and Stephen M. Matyas. *Cryptography: A New Dimension In Computer Data Security. A Guide for the Design and Implementation of Secure Systems*. John Wiley & Sons, 1982. ISBN: 0-471-04892-5 (cit. on p. 23).

[MY17] Sarah Miracle and Scott Yilek. “Cycle Slicer: An Algorithm for Building Permutations on Special Domains”. In: *ASIACRYPT 2017, Part III*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Vol. 10626. LNCS. Springer, Heidelberg, Dec. 2017, pp. 392–416. DOI: 10.1007/978-3-319-70700-6_14 (cit. on p. 26).

[Mou+11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. “Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming”. In: *Inscrypt’11*. Vol. 7537. LNCS. Springer, 2011, pp. 57–76. DOI: 10.1007/978-3-642-34704-7_5 (cit. on pp. 40–42).

[Nyb91] Kaisa Nyberg. “Perfect Nonlinear S-Boxes”. In: *EUROCRYPT’91*. Ed. by Donald W. Davies. Vol. 547. LNCS. Springer, Heidelberg, Apr. 1991, pp. 378–386. DOI: 10.1007/3-540-46416-6_32 (cit. on p. 50).

[Nyb94] Kaisa Nyberg. “Differentially Uniform Mappings for Cryptography”. In: *EUROCRYPT’93*. Ed. by Tor Helleseth. Vol. 765. LNCS. Springer, Heidelberg, May 1994, pp. 55–64. DOI: 10.1007/3-540-48285-7_6 (cit. on p. 36).

[Nyb95] Kaisa Nyberg. “S-boxes and Round Functions with Controllable Linearity and Differential Uniformity”. In: *FSE’94*. Ed. by Bart Preneel. Vol. 1008. LNCS. Springer, Heidelberg, Dec. 1995, pp. 111–130. DOI: 10.1007/3-540-60590-8_9 (cit. on p. 61).

[Nyb12] Kaisa Nyberg. ““Provable” Security against Differential and Linear Cryptanalysis (Invited Talk)”. In: *FSE 2012*. Ed. by Anne Canteaut. Vol. 7549. LNCS. Springer, Heidelberg, Mar. 2012, pp. 1–8. DOI: 10.1007/978-3-642-34047-5_1 (cit. on p. 73).

[NK95] Kaisa Nyberg and Lars R. Knudsen. “Provable Security Against a Differential Attack”. In: *Journal of Cryptology* 8.1 (Dec. 1995), pp. 27–37. DOI: 10.1007/BF00204800 (cit. on p. 73).

[Paa97] Christof Paar. “Optimized Arithmetic for Reed-Solomon Encoders”. In: *ISIT*. IEEE, 1997. DOI: 10.1109/ISIT.1997.613165 (cit. on pp. 104, 109, 111, 112, 116).

[Par+03] Sangwoo Park, Soo Hak Sung, Sangjin Lee, and Jongin Lim. “Improving the Upper Bound on the Maximum Differential and the Maximum Linear Hull Probability for SPN Structures and AES”. In: *FSE 2003*. Ed. by Thomas Johansson. Vol. 2887. LNCS. Springer, Heidelberg, Feb. 2003, pp. 247–260. DOI: 10.1007/978-3-540-39887-5_19 (cit. on pp. 32, 39, 73).

[Per17] Léo Paul Perrin. “Cryptanalysis, Reverse-Engineering and Design of Symmetric Cryptographic Algorithms”. Available at http://orbilu.uni.lu/bitstream/10993/31195/1/thesis.pdf. PhD thesis. University of Luxembourg, 2017 (cit. on pp. 40, 134, 138).

[Pos09] Axel York Poschmann. “Lightweight Cryptography: Cryptographic Engineering for a Pervasive World”. PhD thesis. Ruhr-Universität Bochum, 2009. IACR PREPRINT: 2009/516 (cit. on p. 118).

[Rey+18] Arash Reyhani-Masoleh, Mostafa Taha, and Doaa Ashmawy. “Smashing the Implementation Records of AES S-box”. In: *IACR TCHES* 2018.2 (2018). https://tches.iacr.org/index.php/TCHES/article/view/884, pp. 298–336. ISSN: 2569-2925. DOI: 10.13154/tches.v2018.i2.298-336 (cit. on p. 118).

[Rie] Konrad Rieck. *Top-100 Crypto Papers*. Available at https://www.sec.cs.tu-bs.de/~konrieck/topnotch/crypto_top100.html, visited 9th June 2019. (Cit. on p. 4).

[Rij+96] Vincent Rijmen, Joan Daemen, Bart Preneel, Anton Bossalaers, and Erik De Win. “The Cipher SHARK”. In: *FSE’96*. Ed. by Dieter Gollmann. Vol. 1039. LNCS. Springer, Heidelberg, Feb. 1996, pp. 99–111. DOI: 10.1007/3-540-60865-6_47 (cit. on p. 41).

[Røn+17] Sondre Rønjom, Navid Ghaedi Bardeh, and Tor Helleseth. “Yoyo Tricks with AES”. In: *ASIACRYPT 2017, Part I*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Vol. 10624. LNCS. Springer, Heidelberg, Dec. 2017, pp. 217–243. DOI: 10.1007/978-3-319-70694-8_8 (cit. on p. 22).

[Rot76] Oscar S Rothaus. “On ‘bent’ functions”. In: *Journal of Combinatorial Theory, Series A* 20.3 (1976), pp. 300–305 (cit. on p. 51).

[SHA3] *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. National Institute of Standards and Technology (NIST), FIPS PUB 202, Aug. 2015. DOI: 10.6028/NIST.FIPS.202 (cit. on p. 132).

[Saj+12] Mahdi Sajadieh, Mohammad Dakhilalian, Hamid Mala, and Behnaz Omoomi. “On construction of involutory MDS matrices from Vandermonde Matrices in GF(2^q)”. In: *Designs, Codes and Cryptography* 64.3 (Sept. 2012), pp. 287–308. ISSN: 1573-7586. DOI: 10.1007/s10623-011-9578-x (cit. on p. 106).

[SS16a] Sumanta Sarkar and Siang Meng Sim. “A Deeper Understanding of the XOR Count Distribution in the Context of Lightweight Cryptography”. In: *AFRICACRYPT 16*. Ed. by David Pointcheval, Abderrahmane Nitaj, and Tajjeeddine Rachidi. Vol. 9646. LNCS. Springer, Heidelberg, Apr. 2016, pp. 167–182. DOI: 10.1007/978-3-319-31517-1_9 (cit. on pp. 104, 109, 110).

[SS16b] Sumanta Sarkar and Habeeb Syed. “Lightweight Diffusion Layer: Importance of Toeplitz Matrices”. In: *IACR Trans. Symm. Cryptol.* 2016.1 (2016). http://tosc.iacr.org/index.php/ToSC/article/view/537, pp. 95–113. ISSN: 2519-173X. DOI: 10.13154/tosc.v2016.i1.95-113 (cit. on pp. 104, 105, 108, 109, 113, 116, 117, 119).

[SS17] Sumanta Sarkar and Habeeb Syed. “Analysis of Toeplitz MDS Matrices”. In: *ACISP 17, Part II*. Ed. by Josef Pieprzyk and Suriadi Suriadi. Vol. 10343. LNCS. Springer, Heidelberg, July 2017, pp. 3–18 (cit. on pp. 104, 109, 119).

[ST17] Yu Sasaki and Yosuke Todo. “New Impossible Differential Search Tool from Design and Cryptanalysis Aspects - Revealing Structural Properties of Several Ciphers”. In: *EUROCRYPT 2017, Part III*. Ed. by Jean-Sébastien Coron and Jesper Buus Nielsen. Vol. 10212. LNCS. Springer, Heidelberg, 2017, pp. 185–215. DOI: 10.1007/978-3-319-56617-7_7 (cit. on p. 126).

[Sat+01] Akashi Satoh, Sumio Morioka, Kohji Takano, and Seiji Munetoh. “A Compact Rijndael Hardware Architecture with S-Box Optimization”. In: *ASIACRYPT 2001*. Ed. by Colin Boyd. Vol. 2248. LNCS. Springer, Heidelberg, Dec. 2001, pp. 239–254. DOI: 10.1007/3-540-45682-1_15 (cit. on p. 111).

[Sch+98] Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson. *Twofish: A 128-Bit Block Cipher*. 1998 (cit. on p. 120).

[Seg55] Beniamino Segre. “Curve razionali normali ek-archi negli spazi finiti”. In: *Annali di Matematica Pura ed Applicata* 39.1 (Dec. 1955), pp. 357–379. ISSN: 1618-1891. DOI: 10.1007/BF02410779 (cit. on p. 105).

[Sha45] Claude E. Shannon. *A Mathematical Theory of Cryptography*. Available at the IACR’s webpage. 1945 (cit. on pp. 13, 23).

[Sha49] Claude E. Shannon. “Communication theory of secrecy systems”. In: *Bell Systems Technical Journal* 28.4 (1949), pp. 656–715 (cit. on pp. 18, 23).

[Shi+02] Takeshi Shimoyama, Hitoshi Yanami, Kazuhiro Yokoyama, Masahiko Takenaka, Kouichi Itoh, Jun Yajima, Naoya Torii, and Hidema Tanaka. “The Block Cipher SC2000”. In: *FSE 2001*. Ed. by Mitsuru Matsui. Vol. 2355. LNCS. Springer, Heidelberg, Apr. 2002, pp. 312–327. DOI: 10.1007/3-540-45473-X_26 (cit. on pp. 60, 134).

[Shi+07] Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. “The 128-Bit Blockcipher CLEFIA (Extended Abstract)”. In: *FSE 2007*. Ed. by Alex Biryukov. Vol. 4593. LNCS. Springer, Heidelberg, Mar. 2007, pp. 181–195. DOI: 10.1007/978-3-540-74619-5_12 (cit. on p. 120).

[Sid71] V. M. Sidelnikov. “On the mutual correlation of sequences”. In: *Soviet Mathematics Doklad* 12 (1971), pp. 197–201 (cit. on p. 50).

[Sim+15] Siang Meng Sim, Khoongming Khoo, Frédérique E. Oggier, and Thomas Peyrin. “Lightweight MDS Involution Matrices”. In: *FSE 2015*. Ed. by Gregor Leander. Vol. 9054. LNCS. Springer, Heidelberg, Mar. 2015, pp. 471–493. DOI: 10.1007/978-3-662-48116-5_23 (cit. on pp. 104, 108–110, 116, 119).

[Sto16] Ko Stoffelen. “Optimizing S-Box Implementations for Several Criteria Using SAT Solvers”. In: *FSE 2016*. Ed. by Thomas Peyrin. Vol. 9783. LNCS. Springer, Heidelberg, Mar. 2016, pp. 140–160. DOI: 10.1007/978-3-662-52993-5_8 (cit. on p. 111).

[Sun+16] Bing Sun, Meicheng Liu, Jian Guo, Vincent Rijmen, and Ruilin Li. “Provable Security Evaluation of Structures Against Impossible Differential and Zero Correlation Linear Cryptanalysis”. In: *EUROCRYPT 2016, Part I*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Vol. 9665. LNCS. Springer, Heidelberg, May 2016, pp. 196–213. DOI: 10.1007/978-3-662-49890-3_8 (cit. on p. 126).

[SW09] Guanghong Sun and Chuankun Wu. “The lower bounds on the second order nonlinearity of three classes of Boolean functions with high nonlinearity”. In: *Information Sciences* 179.3 (2009), pp. 267–278. ISSN: 0020-0255. DOI: 10.1016/j.ins.2008.10.002 (cit. on p. 69).

[TM18] Deng Tang and Subhamoy Maitra. “Construction of n-Variable ($n \equiv 2 mod 4$) Balanced Boolean Functions With Maximum Absolute Value in Autocorrelation Spectra $< 2^{n/2}$”. In: *IEEE Trans. Information Theory* 64.1 (2018), pp. 393–402. DOI: 10.1109/TIT.2017.2769092 (cit. on pp. 61, 99).

[Tes15] Stefano Tessaro. “Optimally Secure Block Ciphers from Ideal Primitives”. In: *ASIACRYPT 2015, Part II*. Ed. by Tetsu Iwata and Jung Hee Cheon. Vol. 9453. LNCS. Springer, Heidelberg, 2015, pp. 437–462. DOI: 10.1007/978-3-662-48800-3_18 (cit. on pp. 7, 26, 71, 72, 74, 78, 79, 83).

[Tez14] Cihangir Tezcan. “Improbable differential attacks on Present using undisturbed bits”. In: *J. Computational Applied Mathematics* 259 (2014), pp. 503–511 (cit. on p. 126).

[Tez16] Cihangir Tezcan. “Truncated, Impossible, and Improbable Differential Analysis of ASCON”. In: *ICISSP 2016*. Ed. by Olivier Camp, Steven Furnell, and Paolo Mori. SciTePress, 2016, pp. 325–332. DOI: 10.5220/0005689903250332 (cit. on pp. 123, 126).

[Tez+14] Cihangir Tezcan, Halil Kemal Taskin, and Murat Demircioglu. “Improbable Differential Attacks on Serpent using Undisturbed Bits”. In: *SIN*. Ed. by Ron Poet and Muttukrishnan Rajarajan. ACM, 2014, p. 145. DOI: 10.1145/2659651.2659660 (cit. on p. 126).

[Tod+16] Yosuke Todo, Gregor Leander, and Yu Sasaki. “Nonlinear Invariant Attack - Practical Attack on Full SCREAM, iSCREAM, and Midori64”. In: *ASIACRYPT 2016, Part II*. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Vol. 10032. LNCS. Springer, Heidelberg, Dec. 2016, pp. 3–33. DOI: 10.1007/978-3-662-53890-6_1 (cit. on p. 86).

[Tok15] Natalia Tokareva. *Bent Functions. Results and Applications to Cryptography*. Elsevier, 2015. ISBN: 978-0-12-802318-1 (cit. on p. 51).

[Tsu+08] Yukiyasu Tsunoo, Etsuko Tsujihara, Maki Shigeri, Teruo Saito, Tomoyasu Suzaki, and Hiroyasu Kubo. “Impossible Differential Cryptanalysis of CLEFIA”. In: *FSE 2008*. Ed. by Kaisa Nyberg. Vol. 5086. LNCS. Springer, Heidelberg, Feb. 2008, pp. 398–411. DOI: 10.1007/978-3-540-71039-4_25 (cit. on p. 140).

[Vau98] Serge Vaudenay. “Provable Security for Block Ciphers by Decorrelation”. In: *STACS’98*. Vol. 1373. LNCS. Springer, 1998, pp. 249–275. DOI: 10.1007/BFb0028566 (cit. on p. 73).

[Vau12] Serge Vaudenay. *The End of Encryption based on Card Shuffling*. CRYPTO 2012 Rump Session. Available at crypto.2012.rump.cr.yp.to. 2012 (cit. on p. 72).

[Vis+17] Andrea Visconti, Chiara Valentina Schiavo, and René Peralta. *Improved upper bounds for the expected circuit complexity of dense systems of linear equations over GF(2)*. Cryptology ePrint Archive, Report 2017/194. http://eprint.iacr.org/2017/194. 2017 (cit. on pp. 104, 109, 112, 113).

[Wag99] David Wagner. “The Boomerang Attack”. In: *FSE’99*. Ed. by Lars R. Knudsen. Vol. 1636. LNCS. Springer, Heidelberg, Mar. 1999, pp. 156–170. DOI: 10.1007/3-540-48519-8_12 (cit. on p. 78).

[Wan+11] Meiqin Wang, Yue Sun, Nicky Mouha, and Bart Preneel. “Algebraic Techniques in Differential Cryptanalysis Revisited”. In: *ACISP 11*. Ed. by Udaya Parampalli and Philip Hawkes. Vol. 6812. LNCS. Springer, Heidelberg, July 2011, pp. 120–141 (cit. on p. 144).

[WJ17] Qian Wang and Chenhui Jin. “Upper bound of the length of truncated impossible differentials for AES”. In: *Designs, Codes and Cryptography* (2017). DOI: 10.1007/s10623-017-0411-z (cit. on p. 126).

[Wat+02] Dai Watanabe, Soichi Furuya, Hirotaka Yoshida, Kazuo Takaragi, and Bart Preneel. “A New Keystream Generator MUGI”. In: *FSE 2002*. Ed. by Joan Daemen and Vincent Rijmen. Vol. 2365. LNCS. Springer, Heidelberg, Feb. 2002, pp. 179–194. DOI: 10.1007/3-540-45661-9_14 (cit. on p. 120).

[#23830] Friedrich Wiemer. *Add SBox instances: DBlock Cipher*. #23830. 2017 (cit. on p. 10).

[#22988] Friedrich Wiemer. *Add SBox instances*. #22988. 2017 (cit. on p. 10).

[#14549] Friedrich Wiemer. *Memory leak in algebraic_immunity of BooleanFunction class*. #14549. 2017 (cit. on p. 10).

[#22986] Friedrich Wiemer. *Refactor SBox code (move from crypto.mq to crypto)*. #22986. 2017 (cit. on p. 10).

[#22453] Friedrich Wiemer. *A mistake in the mq.Sbox.polynomials*. #22453. 2018 (cit. on p. 10).

[#25765] Friedrich Wiemer. *Add some more SBox constructions*. #25765. 2018 (cit. on p. 11).

[#24566] Friedrich Wiemer. *BooleanFunction evaluated on Integer computes IndexError wrongly*. #24566. 2018 (cit. on p. 10).

[#25516] Friedrich Wiemer. *Huge delay introduced in SBox nonlinearity*. #25516. 2018 (cit. on p. 11).

[#25708] Friedrich Wiemer. *Rename SBox methods for difference_distribution_matrix and similar to _table*. #25708. 2018 (cit. on p. 11).

[#25766] Friedrich Wiemer. *SBox.boomerang_uniformity*. #25766. 2018 (cit. on p. 11).

[#25633] Friedrich Wiemer. *Speed up SBox module*. #25633. 2018 (cit. on p. 11).

[#25733] Friedrich Wiemer. *Speed up SBox.boomerang_connectivity_matrix*. #25733. 2018 (cit. on p. 11).

[#25744] Friedrich Wiemer. *construct sbox from univariate polynomial*. #25744. 2018 (cit. on p. 11).

[#24819] Friedrich Wiemer. *sbox linear approximation matrix scaling*. #24819. 2018 (cit. on p. 10).

[#26009] Friedrich Wiemer. *Add derivative method for SBox*. #26009. 2019 (cit. on p. 11).

[#25735] Friedrich Wiemer. *Crypto Linear Layer Module*. #25735. 2019 (cit. on p. 11).

[#28001] Friedrich Wiemer. *Rename BooleanFunction.absolut_indicator to .absolute_indicator*. #28001. 2019 (cit. on p. 11).

[#25739] Friedrich Wiemer and Jean-Pierre Flori. *sig_check in boolean_function cython code*. #25735. 2018 (cit. on p. 11).

[WW12] Shengbao Wu and Mingsheng Wang. “Automatic Search of Truncated Impossible Differentials for Word-Oriented Block Ciphers”. In: *INDOCRYPT 2012*. Ed. by Steven D. Galbraith and Mridul Nandi. Vol. 7668. LNCS. Springer, Heidelberg, Dec. 2012, pp. 283–302. DOI: `10.1007/978-3-642-34931-7_17` (cit. on pp. 126, 140).

[WZ11] Wenling Wu and Lei Zhang. “LBlock: A Lightweight Block Cipher”. In: *ACNS 11*. Ed. by Javier Lopez and Gene Tsudik. Vol. 6715. LNCS. Springer, Heidelberg, June 2011, pp. 327–344. DOI: `10.1007/978-3-642-21554-4_19` (cit. on p. 140).

[ZZ96] Xian-Mo Zhang and Yuliang Zheng. “GAC — the Criterion for Global Avalanche Characteristics of Cryptographic Functions”. In: (1996). Ed. by Hermann Maurer, Cristian Calude, and Arto Salomaa, pp. 320–337. DOI: `10.1007/978-3-642-80350-5_30` (cit. on pp. 56, 61).

[Zha+00] Xian-Mo Zhang, Yuliang Zheng, and Hideki Imai. “Relating Differential Distribution Tables to Other Properties of of Substitution Boxes”. In: *Designs, Codes and Cryptography* 19.1 (2000), pp. 45–63. ISSN: 1573-7586. DOI: `10.1023/A:1008359713877` (cit. on pp. 56, 57, 60, 61).

[Zha+16] Ruoxin Zhao, Baofeng Wu, Rui Zhang, and Qian Zhang. *Designing Optimal Implementations of Linear Layers (Full Version)*. Cryptology ePrint Archive, Report 2016/1118. `http://eprint.iacr.org/2016/1118`. 2016 (cit. on p. 111).

[Zhe+93] Yuliang Zheng, Josef Pieprzyk, and Jennifer Seberry. “HAVAL - A One-Way Hashing Algorithm with Variable Length of Output”. In: *AUSCRYPT’92*. Ed. by Jennifer Seberry and Yuliang Zheng. Vol. 718. LNCS. Springer, Heidelberg, Dec. 1993, pp. 83–104. DOI: `10.1007/3-540-57220-1_54` (cit. on p. 51).

[Zho+17] Lijing Zhou, Licheng Wang, and Yiru Sun. *On the Construction of Lightweight Orthogonal MDS Matrices*. Cryptology ePrint Archive, Report 2017/371. `http://eprint.iacr.org/2017/371`. 2017 (cit. on pp. 104, 109, 110).

Versicherung an Eides Statt

Ich versichere an Eides statt, dass ich die eingereichte Dissertation selbstständig und ohne unzulässige fremde Hilfe verfasst, andere als die in ihr angegebene Literatur nicht benutzt und dass ich alle ganz oder annähernd übernommenen Textstellen sowie verwendete Grafiken, Tabellen und Auswertungsprogramme kenntlich gemacht habe.

Außerdem versichere ich, dass die vorgelegte elektronische mit der schriftlichen Version der Dissertation übereinstimmt und die Abhandlung in dieser oder ähnlicher Form noch nicht anderweitig als Promotionsleistung vorgelegt und bewertet wurde.

Datum

Unterschrift