Roberto Avanzi

A Salad of Block Ciphers

The State of the Art in Block Ciphers and their Analysis

July 28, 2017

This book has been typeset using the LaTeX document preparation system, the XeTeX engine and the TikZ graphics package.

The text has been mostly set to the TeX Gyre font family, in particular TeX Gyre Pagella, TeX Gyre Heros, and TeX Gyre Cursor, with the exception of Cyrillic text, which has been set to Linux Libertine, and japanese text, which has been set to Utsukushi Mincho.

Mathematical formulas are set to TeX Gyre Pagella Math, with the mathematical calligraphic symbols from XITS Math, and a few more individual symbols taken from the `newpxmath` and the AMS Math fonts.

© 2013-2017 Roberto Avanzi

Some rights reserved:

This work is licensed under the following Creative Commons license

Attribution-NonCommercial-NoDerivatives 4.0 International

`http://creativecommons.org/licenses/by-nc-nd/4.0/`

To Greg and Pat in friendship.

Foreword

This book is a survey on the state of the art in block cipher design and analysis. It is work in progress, and it has been for the good part of the last three years – sadly, for various reasons no significant changes habe been made for about one year and a half, and only recently I have started updating some parts again.

However, it is also in a self-contained, useable, and *relatively* polished state, and for this reason I have decided to release this *snapshot* onto the public as a service to the cryptographic community, both in order to obtain feedback, and also as a means to give something back to the community from which I have learned much.

At some point I will produce a final version – whatever being a "final version" means in the constantly evolving field of block cipher design – and I will publish it. In the meantime I hope the material contained here will be useful to other people.

Roberto Avanzi

Munich, July 2017

Introduction

> *Einstein, the most creative physicist since Newton, is one of my heroes. I wrote a book on relativity to teach myself the theory.*
>
> Martin Gardner

The *topic* of this book is the state of the art in block ciphers. These are arguably the best understood primitives in the field of symmetric cryptography: Over the course of more than four decades, solid theoretical foundations have been developed and as a result several established design principles are now available to the cryptographer to construct block ciphers resistant against all known attacks. At the same time, many challenges are still as interesting as ever, if not more: for instance to as compact as possible or to minimise latency, while at the same time taking no compromises in regard to security; to develop new and better attacks to break existing ciphers; and to invent new designs or design methodologies.

The *purpose* of this book to get a comprehensive overview of the research done so far in this huge and complex field. Only cryptographic methods whose details have been publicly disclosed are reported here, and all information contained herein comes from openly available sources, except for the occasional explanation obtained by private communication.

The *focus* of this book is on ciphers whose design criteria have been carefully analysed by the cryptographic community, as well as on historically relevant ciphers whose design criteria have proven influential, recent developments, and off-the-beaten-path designs.

The main design types of block ciphers are discussed, as well as the cryptanalytic techniques used to assess their security. This information has been put into its proper historical context as well, for instance the rationale for new designs, and the way a cipher influenced successive cryptographic algorithms.

It is very difficult to talk about design without talking about cryptanalysis, and one cannot discuss cryptanalysis without referring to at least some specific design techniques. In fact, one cannot talk about one without discussing the other, since the best way to learn how to design block ciphers is by breaking them. Block cipher analysis is driven by the concrete instances of the designs themselves, and therefore it is advisable to consider concrete examples when studying cryptanalytic techniques. However, a separation of the subjects of design and cryptanalysis, even if at times somewhat arbitrary, is necessary, especially in a work of reference. A constant intermixing of design principles, ciphers, and analysis would only make the exposition confusing. Therefore, we treat these subjects separately: The first two chapters will be on design (Chapter 1 on page 21) and on cryptanalysis (Chapter 2 on page 71), and only after this we shall move to a discussion of several concrete ciphers that have marked the history of the last three decades of block cipher design (Chapter 3 on page 125). However, some cryptanalytic concepts are mentioned already in the design chapter to provide context.

We do not define complexity classes. We do not discuss adversarial models. We do not treat modes of operation. We are also sparse on the mathematical background. There are excellent books and lecture notes out there for all this stuff.

This document is *not* a textbook nor an encyclopaedia on block ciphers. It is not meant as a

replacement for Lars Knudsen and Matt Robshaw's excellent book *The Block Cipher Companion* [KR11]. In fact, that book belongs to the desk beside he present one to properly understand the designs documented herein. It is assumed that the reader will either know or learn the necessary mathematical background as necessary. It also is part of the learning process of its author, so a particular balance in exposition is not necessarily to be expected, even though we attempted to present the information in a uniform way.

More than 80 block ciphers are described – to variable level of detail – from the DES to algorithms disclosed in early 2014. The advantages and disadvantages of these ciphers, their weaknesses, the types of attacks, and their performance have been compared whenever there was publicly available information about them. Performance comparisons are mostly in HW and in SW on micro-controllers, but also general purpose CPUs are briefly considered for a few ciphers. Full references are given for all information collected here.

The document originated from my own (LaTeX-ed) notes while I was learning block cipher design and cryptanalysis, as part of my work at Qualcomm. As such the depth of exposition, as well as its quality, may vary wildly from place to place. Ciphers, design concepts, or cryptanalytic methods that I considered more important - or more difficult to understand - will often be described in a more detailed way than matters that I - maybe mistakenly - considered less important or were easier for me to grasp. This also means that the depth of treatment is not necessarily correlated to an objective evaluation of the subject. I did also quote papers in several places in a more literal way than desirable. In the worst case, consider this text as a verbose annotation of an extensive bibliography – even as such it may show some usefulness.

Even though in some cases I aimed at completeness, there are areas where completeness is difficult or impossible to achieve. In particular, this applies to intellectual property. I have tried to determine the intellectual property situation for all the ciphers that I have described, but in some case it was not possible to determine whether aspects of a given cipher were patented or not. In some cases the information is just not easy to find. In no case there is warranty (explicit or implied) about the correctness of information regarding intellectual property. Readers that needs conclusive evidence should talk to a legal counsel and have a patent search performed on their behalf.

I want to thank Cameron McDonald for some content, suggestions and references, and Billy Bob Brumley for several suggestions, a few tables, references and spell checking and style suggestions. I am also indebted to Markku-Juhani Olavi Saarinen for providing precious material and for interesting discussions. Any error is my fault only, and nobody else is responsible.

Since one should never stop learning, this document is by necessity also a work in progress. As such, I am always looking to improve this document, and any polite suggestion is encouraged. Please use the following email address to inform the author about errors and inaccuracies, and send suggestions for improvements and updates: `roberto.avanzi@gmail.com`. All contributions will be acknowledged.

Roberto Avanzi

Munich, January 2016

Table of Contents

List of Figures

List of Tables

Notation

We attempt to maintain a consistent notation throughout the document. The following notations hold except when explicitly stated otherwise. Accordingly, in several cases our descriptions of ciphers and attacks have been modified with respect to the reference papers. Another convention is in the numbering of rounds: we usually start with round 0, and the descriptions of some ciphers have been modified not only to adhere to the notation set below, but also to our round numbering.

$\mathscr{A}, \mathscr{I}, \mathscr{K}, K[\mathscr{I}]$ and $\langle A, \mathscr{I} \rangle$: The symbols $\mathscr{A}$, $\mathscr{I}$, and $\mathscr{K}$ denote set of key bits, usually to extract a partial key from a key, for instance as in $k_1 = K[\mathscr{I}]$.
Key bit sets are identified with vectors whose entries are equal to one for the included bits and zero otherwise. Hence, for a vector A of length ℓ over $\mathbb{F}_2$ and a subset $\mathscr{I}$ of $[0..\ell - 1]$, the notation $\langle A, \mathscr{I} \rangle$ shall denote the sum (parity) of the bits in A at the fixed positions indexed by the numbers in $\mathscr{I}$, i.e. the parity bit of the corresponding choice of bits.

$\mathscr{A}i$: Algebraic immunity of a Boolean function.

b: Where different from the key size n, b is used to denote the block size.

$\mathscr{B}$: The branch number of a linear mapping.

C: A ciphertext (also in derived forms such as C_i, C', etc.).

$\mathfrak{c}, \mathfrak{e}$: In the F-Function of a DES-like cipher, the compression and expansion blocks, that follow and precede the key mixing, respectively.

D, E: E is normally used to denote an encryption function, in which case the letter D represents the corresponding decryption function.
This applies also to indexed or accented variants, i.e. the inverse of E_i, resp. $E^{(1)}$, $\hat{E}$, and E' is denoted by D_i, resp. $D^{(1)}$, $\hat{D}$, and D', etc.
For all other letters the inverse of a function f is denoted by f^{-1}.

D, S, T: When analysing space-time tradeoffs, these are shorthands for (collected) *data*, runtime *space* (or *storage*) and running *time*, usually in conjunction with $N = 2^n$ for the state (and key space) cardinality.

Δ, ∇: Used to denote input, output, state, or key differences / (higher order) differentials.

δ_i: δ_i is often used as a shorthand for ΔP_i.

δ_F: The differential immunity of a function F.

f, F: f can be a generic function, often a Boolean function. In the context of the latter case, F is used to denote a vectorial Boolean function.

F: The F-Function of Feistel ciphers or generic random permutations in Even-Mansour constructions.

$\mathbb{F}$: A finite field. $\mathbb{F}_a$ denotes the finite field with a elements, as in $\mathbb{F}_2$. $\mathbb{F}_{2^8}$, $\mathbb{F}_{17}$ etc.

$\gamma, \lambda, \pi, \psi, \rho, \sigma, \theta$: In the wide trails design strategy, these greek letters denote respectively: a non-linear transformation, a linear transformation, a word permutation, the key schedule, the round function, a key addition, and a layer of S-boxes.

K, k, rk, wk: The letter K denotes (master) keys, such as K, K_i, K' etc.
The lower case k denotes subkeys and round keys k_i, which are usually derived from a master key K. It is sometimes used also in composite symbols, such as rk for a round key, wk for a whitening key, etc.

L, R: In the context of Feistel ciphers the left and right halves of a state, respectively.

ℓ: Use to denote the length of the key or of the keys used in a cipher, when different from the block size n.

$\mathcal{M}, \mathcal{N}$: Sets.

$\mathcal{M}^d$: Set of all sequences which consist of d elements of a set $\mathcal{M}$.

n: When studying block ciphers, n is used to denote the block length of the cipher; when studying S-boxes n often denotes the S-box size. In cryptographic schemes which explicitly use more than one key K_1, K_2, etc. that by design are not considered as a sub keys of a larger key, n usually denotes the size of each of these keys. In these cases $N = 2^n$.

n: In a few cases n denotes the length of a vector, especially when the state of a cipher is represented as a vector of n elements over a given algebraic structure. Used when studying diffusion layers and wide-trail designs.

nl: The nonlinearity of a (vectorial) Boolean function.

$\mathcal{O}$: Encryption or decryption oracle.

P: Usually a plaintext (including derived forms like P_i, P', etc.), or a permutation (P-box).

$\mathcal{P}$: Probability.

$P_f(\alpha, \beta)$: is defined as

- $\mathcal{P}[f(\alpha) = \beta]$ in the case of linear cryptanalysis;
- $\mathcal{P}[f(x) + f(x + \alpha) = \beta]$ in the context of differential cryptanalysis; and
- $\mathcal{P}[\Delta_\alpha^{(i)} f(x) = \beta]$ where $\alpha = (\alpha_1, \ldots, \alpha_i)$, for higher order differential cryptanalysis.

r: The number of rounds of an iterative cipher.

$\mathbb{R}$: The field of real numbers.

S: Commonly used to denote S-boxes in the description of ciphers.

$\mathsf{wt}(a)$: The Hamming weight of the number, finite field element, or vector a, i.e. the number of non-zero bits, or elements, of its argument.

w: For a function f, its Walsh transform/spectrum is denoted by f^w.

$\mathbb{Z}$: The ring of the integers. The notation $\mathbb{Z}/n\mathbb{Z}$ is used for the ring of integers modulo n.

$\bar{}$: For a bit string a, $\bar{a}$ denotes the bitwise complement of a.

$\oplus$: Bitwise exclusive or, or XOR.

$\wedge$ and $\oslash$: Logical (bitwise) AND. $\oslash$ is used in diagrams.

$\vee$ and $\circledvee$: Logical (bitwise) OR. $\circledvee$ is used in diagrams.

$\boxplus$ and $\boxminus$: Modular addition and subtraction, respectively (especially in diagrams).

$\circ$: Functional composition where by the notation $g \circ f$ it is understood that g follows f, i.e. $(g \circ f)(x) = g(f(x))$.

$*$, $\circledast$, $\circledcirc$ and $\odot$: Various compositions of values.

$\|$: Concatenation of bit strings. Often used for states of Feistel constructions. For matrices M_1 and M_2 with equal number of rows, $M_1 \| M_2$ denotes the matrix obtained by juxtaposition (also written as $(M_1 | M_2)$).

$\ll \alpha$, $\ll_\alpha$: Left shift by α bits (zero-filled).

$\gg \alpha$, $\gg_\alpha$: Right shift by α bits (zero-filled).

$\lll \alpha$, $\lll_\alpha$: Left rotate by α bits.

$\ggg \alpha$, $\ggg_\alpha$: Right rotate by α bits.

$\cup$, $\cap$, $\setminus$: Set theoretic union, intersection, and subtraction.

$\#$: The cardinality of a set.

$\emptyset$: The empty set.

$[a..b]$ and $[n]$: $[a..b]$ is the set of integers from a to b.

Chapter 1

Design

> *Lots of people working in cryptography have no deep concern with real application issues. They are trying to discover things clever enough to write papers about.*
>
> Whitfield Diffie

In 1883 Auguste Kerckhoffs published two papers in *La Cryptographie Militaire* [Ker83a, Ker83b] in which he stated six axioms of cryptography. In modern words, these axioms can be rephrased as follows: can be summarised as:

1: The system must be mathematically and physically indecipherable;

2: It must not rely on on secret algorithms or parameters;

3: It must be easy/efficient to transmit the key and change it;

4: It must be applicable to telecommunications;

5: It must allow lightweight implementations; and

6: It must be user friendly and its use must not require technical prowess.

We want to focus our attention on the second axiom, now known as **Kerckhoffs' Principle**. The original text is

> *Il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi.*

which, in English, translates as:

> *It [the method] must not be required to be secret, and it must be able to fall into the enemy's hands without inconvenience.*

This principle implies that the security shall rely only on the secrecy of the key. It is also known as **Shannon's Maxim** after Claude Shannon who formulated it as "*the enemy knows the system.*" This is in fact the most important design principle for *any* cipher, not only block ciphers, but also stream ciphers, or even public key cryptosystems: *avoid secret designs or design principles*. We strongly believe that obscurity does not bring security: Not only does peer scrutiny lead to better ciphers, but obscurity often gives a false sense of security, leading to design mistakes.

Bruce Schneier argues in a CRYPTO-GRAM newsletter [Sch02][1] that Kerckhoffs' papers are probably the earliest papers that support the non-secrecy of cryptographic algorithms.

[1] See also [Coh87], Chapter 2.1 "A Short History of Cryptography".

Kerckhoffs' Principle does not only apply to the "blueprints" of a cryptographic algorithm, but also to the methods used to determine any constants used in the system. When the US National Bureau of Standards (NBS) introduced the Data Encryption Standard, it was known that the US National Security Agency (NSA) had changed some of the hardwired constants. At the time there was suspicion that this was done to insert a mathematical backdoor into the system – and if the NSA had it, somebody else could also find it.

It was later disclosed that this was not the case – in fact it was later disclosed that these modifications strengthened the cipher against differential cryptanalysis, an attack that was not publicly disclosed at the time – but this was sufficient to cast doubts on the cipher. For this reason several recent cryptographic methods contain **nothing up my sleeve numbers**, i.e. values obtained in a way that is "above suspicions," for instance from the binary expansions of important mathematical constants such as π, e, φ, etc.

In the following sections we will formalise concrete design principles and methodologies which are specific to block ciphers. The material is roughly organised in four parts:

1: In Section 1.1 we introduce fundamental concepts, such as the definitions of block cipher, product and iterated block ciphers, rounds, key schedule, avalanche, confusion, diffusion and the security reduction principle.

2: Sections 1.2 to 1.7 describe the main design topologies and components of block ciphers, namely substitution-permutation networks, Feistel ciphers, Lai-Massey designs, etc.

3: Sections 1.8 and 1.9 deal with the main building blocks of block ciphers, that is the functions that provide diffusion and confusion.

4: Finally, the last section serves as a sort of appendix to this chapter, collecting information about less common design types and building blocks.

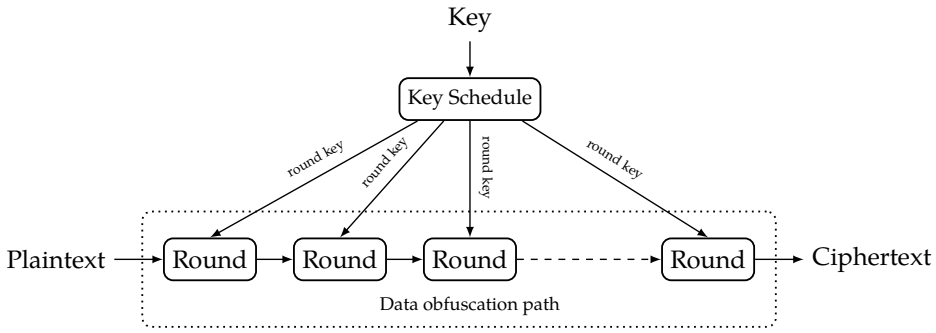
1.1 Fundamental Definitions and Design Principles

A **block cipher** is a pair of deterministic algorithms, called **encryption** and **decryption**, that respectively apply and undo a permutation, selected by a **key**, of the set of all **blocks** (bit strings) of a fixed length: The encryption converts a **plaintext** into a **ciphertext**, and the decryption converts a ciphertext into the original plaintext.

A cipher is considered secure when it is not feasible to determine the key, even with a large set of plaintext/ciphertext pairs at disposal.

Ideally, the best attacks against a block cipher should not run considerably faster than brute force on the key space. For instance, it is necessary that a block cipher, unlike, say, Caesar's cipher, does not show any kind of linearity. Caesar's cipher can be interpreted as modular addition of a secret key. Whereas brute force search requires on average 13 attempts before guessing the correct key, a known plaintext attack determines the secret key with just one encryption operation and a modular subtraction. Linearity also exposes biases in the distribution of the characters in the plaintext through the ciphertext: Taking again Caesar's cipher as an example, if we know that the plaintext is in English and the most common character in the ciphertext is "g", then we can guess it to be the encryption of the letter "e" and thus the secret key is probably 2.

Figure 1.1: A Product Block Cipher



In fact, a block cipher should not be approximable by any easily computable function of the input with a nonnegligible bias. This includes linear or affine functions as a special case, and there are countless ways to represent the states of a cipher: the most common way is to see them as vectors over $\mathbb{F}_2$, but we can also view the bytes of the states as integers modulo 256, or elements of the Galois field $\mathbb{F}_{2^8}$.

Apart from non linearity, another desirable property is the **avalanche effect**, which means that a small change in either the key or the plaintext should induce a drastic change in the ciphertext. This was formalised by A. F. Webster and Stafford Tavares as the **Strict Avalanche Criterion (SAC)** [WT85]: *Complementing a single bit in the input or in the key should change any bit in the output with probability* $1/2$*, for any input or key bit and for any output bit.*

In order to achieve these goals we could take a table of random permutations: For block and key lengths of n and ℓ bits, we just pick at random 2^ℓ permutations of the elements of the n-dimensional vector space over $\mathbb{F}_2$. But such a table would be huge for any acceptable security level – and we are not yet taking into account the difficulty to obtain a good randomness source.

Hence, a block cipher is constructed by chaining simpler operations, which are individually weak, but combined form a much stronger function. Such a cipher is called a **product cipher**.

Claude Shannon first formalized this approach in [Sha49]. The encryption algorithm of a product block cipher is a sequence of **rounds**, where the input to the first round is the plaintext, the output of each round is passed to the following round as its input, and the output of the last round is the ciphertext. Each round is a simpler cipher itself, i.e. it accept as one of its inputs also a key, called the **round key**, which is derived from the encryption key.

Hence, a block cipher can also be defined as a *triple* (instead of a pair) of algorithms, namely, encryption, decryption and **key schedule**, also called **key expansion**. The latter is the process of *expanding* the given secret key into a set of values used in different places of the encryption and decryption algorithms. If the key schedule is considered as a separate algorithm, then encryption (resp. decryption) is redefined as an algorithm that takes as inputs the plaintext (resp. ciphertext) and the outputs of the key schedule and outputs the ciphertext (resp. plaintext). The encryption part of the cipher is sometimes called the **data (obfuscation) path**, as depicted in Figure 1.1. These definitions are also useful when the key schedule process is computationally expensive and it thus it makes sense to separate it from encryption and decryption when the same key is used repeatedly.

In order to simplify the construction of concrete proposals, Shannon also suggested to build

a product cipher by repeating the same steps over and over until the desired robustness is achieved. Intuitively, increasing the number of rounds improves the security of a cipher at the expense of speed and, conversely, reducing the number of rounds can improve performance at the expense of security.

The resulting cipher is called an **iterated (product) cipher**. The internal **state** of the cipher is initially the plaintext. Each iteration of the round updates the state, i.e. it replaces the state with its output. The ciphertext is then just the final state.

The operations that are used to build the rounds should provide **confusion** and **diffusion**. To quote Shannon [Sha49]:

> *Two methods (other than recourse to ideal systems) suggest themselves for frustrating a statistical analysis. These we may call the methods of diffusion and confusion.*

Diffusion means that the statistical structure of the plaintext is dissipated in the long range statistics of the ciphertext. That is, each part of the ciphertext should depend on as much of the plaintext as possible. Inside a plaintext block, diffusion is attained by means of state permutation operations.

Confusion means that the key is used in such a way that even when an attacker knows the statistics of the plaintext, or has many plaintext/ciphertext pairs, it is still difficult to deduce the key. In Shannon's understanding, each part of the ciphertext should depend on as much of the key as possible. This can be attained when the mathematical dependence of the ciphertext from plaintext and key is very complex, for instance by repeatedly using non-linear functions to mix the state with the round keys. One way to improve the mixing of plaintext and key, especially in the presence of keys larger than the plaintext block, is to use a complex key schedule.

At this point it is noteworthy to remark a fundamental difference between block ciphers, on one side, and stream ciphers, the One Time Pad (invented by Frank Miller in 1882 [Mil82] as shown by Steven M. Bellovin in [Bel11]), and the Vernam Cipher [Ver26] on the other side: Block ciphers build their security on both confusion and diffusion, whereas the other types of ciphers rely exclusively on confusion to achieve security.

Shannon introduced two further design principles:

1. The first is called **reduction**: make the security of the system reducible to some known difficult problem. This principle has been used widely in public-key cryptography, much less so in secret-key cryptography.

2. Shannon's second principle is very pragmatic: *make the system secure against all known attacks.* For secret-key ciphers this is still the best known design principle. Indeed, in the words of Bruce Schneier [Sch98]: *"The only way to become a good algorithm designer is to be a good cryptanalyst: to break algorithms. [...] Only after a student has demonstrated his ability to cryptanalyze the algorithms of others will his own designs be taken seriously."*

We shall put the second principle into practice in an exemplary way after we have discussed *Substitution-Permutation Networks* (SPNs), a large family of block ciphers. This will allow us to understand how some common attack strategies work, and also how a cryptographer hedges her designs against these attacks. The latter will directly translate to concrete requirements of

the building blocks used in the construction of such a cipher. Since SPNs comprise almost all block cipher types encountered in practice, this will be general enough to set the ground for later, more in-depth treatments.

1.2 Substitution-Permutation Networks

Shannon suggested the use of two logically separated types of functions to achieve confusion and diffusion in a product cipher. A **substitution function** or **substitution layer** provides confusion, whereas a **diffusion layer** provides diffusion. This functional separation of the devices that provide confusion and diffusion allows to construct the rounds from simpler operations that can be analysed mathematically on their own terms. This principle also leads to a certain minimalism in the design approach and in its analysis, which has proven to be beneficial for security: Indeed, most designs that have better withstood cryptanalysis are very regular designs built from simple blocks, whereas many an apparently clever design that mixed different types of complex and obscure operations has been broken in a surprisingly short time.

The substitution function (also: **confusion function**, **confusion layer**) can be represented as a table mapping state values to other values, but the actual implementation can be different, for instance a simple circuit or a short program using non-linear operations. This function is sometimes called an **S-box**. Today, the name S-box is usually refers to a simple non-linear operation that modifies just a part of the state, and the full substitution layer consists of piecewise, parallel applications of several S-boxes (not necessarily distinct from each other) to its input. Means to achieve confusion will be discussed in Section 1.9 on page 55.

The permutation layer or **diffusion layer** is sometimes called a **P-box**. The permutation layer can be realised by a simple fixed permutation of the bits of the state, or more general invertible linear transformation of the state as a vector over some ring. (There is nothing in principle that prevents non-linear diffusion layers.) Diffusion will be discussed in Section 1.8 on page 42.

A combination of these two operation types together with mixing of key derived material constitutes a round of a **Substitution-Permutation Network** (also **SP Network** or **SPN**).

Simple functions cannot be expected to bring much confusion and diffusion, but if sufficiently many such functions are suitably chained, the resulting iterated cipher can achieve very high levels of confusion and diffusion, until the avalanche effect is reached.

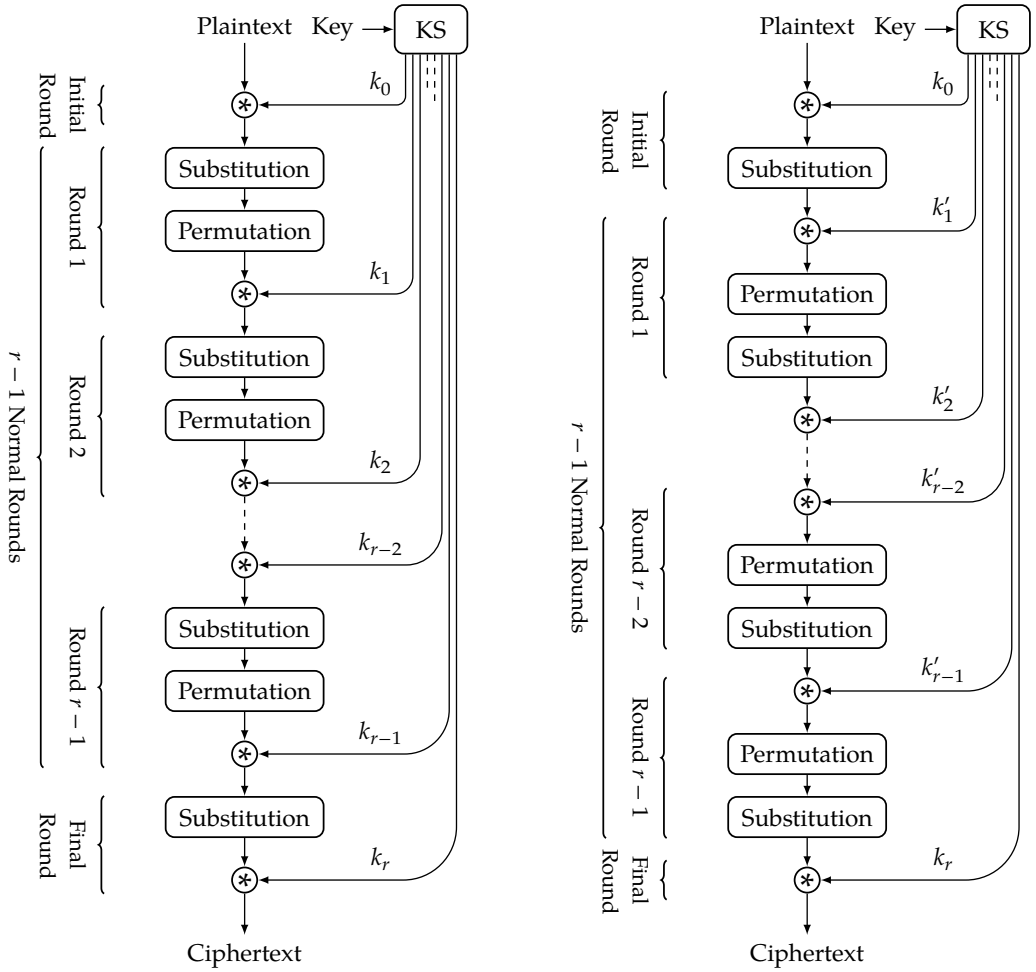
Two variants of the modern generic scheme of a SPN are depicted in Figure 1.2 on the following page, where the reader can immediately note how the general idea of product cipher (as presented in Figure 1.1 on page 23 is implemented.

Several variants are possible, for instance there can be more substitution and permutation layers in each round. If S denotes a substitution layer, P a permutation layer, and K a key mixing layer, them the rounds in Figure 1.2 are of type SPK, with the final round being just SK. Key mixing usually follows the substitution and permutation layers, but there are ciphers where the key mixing is defined as being at the beginning of a round, in which case it is the final round that consists of just key mixing. In some ciphers key material is mixed more than once per round, such as the KSKP round type used in the SAFER family (Section 3.8 on page 148). Other ciphers other can have very complex rounds built out of several simple layers: for instance ICEBERG's round structure (Section 3.24 on page 195) is SPSPS-PKP.

Figure 1.2: Simple Substitution-Permutation Networks

A simple SPN with a KSP round and without final representation is depicted to the left.

The SPN to the right is equivalent, provided that the permutation layer is $\circledast$-linear, in which case round key k'_i is the inverse image of k_i under the i^{th} permutation function.



An important remark is that placing substitution and permutation layers at the very beginning or end of a cipher (even though it may facilitate implementation) does not increase security, because they are fixed modifications of the plaintext or ciphertext. Therefore it is a sensible design decision to ensure that key mixing instead occurs at the very beginning and very end of the cipher. (This is related to the **FX construction**, cf. Section 1.6 on page 38.)

Returning to Figure 1.2 (left), the initial key mixing is called the **initial round**. Several additional rounds composed of substitution and permutation layers and key mixing follow. Only the **final round** may sometimes omit the permutation layer, since it is often simply a linear operation, and omitting it is equivalent to mixing a different, linearly transformed round key, at the end of the round. Again, for ease of implementation or for design reasons the permutation layer may be present in the last round in a simplified form (such as in SQUARE, cf. Section 3.11 on page 158, and its successors, including the AES, cf. Section 3.20 on page 180).

For this type of ciphers, if the permutation layer is linear (that is $P(x \circledast y) = P(x) \circledast P(y)$), then decryption can be implemented with a similar control structure as the encryption, i.e. the inverse of the circuit to the *right* is used to invert the circuit to the *left*: It will still be necessary to reverse the order of the round keys and to linearly transform some of them, and possibly add control logic to choose whether to use the inverse substitution and permutation layers for decryption, but resources can be saved. A further development of this intuition is part of the Wide Trail design strategy (Section 1.4 on page 33) and in particular of involutory ciphers such as Khazad and Shark (Subsection 3.21.2 on page 188) and Anubis (Subsection 3.21.3 on page 190).

Key mixing is usually a simple operation, and in most cases it performed by a XOR. The second most popular operation is modular addition, as in GOST (Section 3.4 on page 138), RC2 (Section 3.5 on page 140), RC5 (Section 3.10 on page 156), TEA (Section 3.12 on page 159), Twofish (Section 3.13 on page 162), RC6 (Section 3.15 on page 165), SEA (Section 3.26 on page 197), Threefish (Section 3.30 on page 205), and Bel-T (Section 3.33 on page 212) – to name just a few examples. In software, modular addition is often as expensive as XOR, so it is an inexpensive way to improve diffusion and confusion. This is not always true as in some ciphers there may be optimisations relying on the use of XOR that would not be possible if a different operation is used. Also, in hardware addition is always more expensive than XOR, as the increased latency of addition is not hidden by the pipeline of the CPU. Some ciphers use more than one operation to mix the subkeys with the state. For instance, in IDEA (Section 3.6 on page 142) and MESH (Subsection 3.6.6 on page 145) addition modulo 2^{16} and multiplication modulo $2^{16}+1$ are used, and in the CAST (Section 3.7 on page 146) and SAFER (Section 3.8 on page 148) families both XOR and modular addition/subtraction are used. SHARK (Subsection 3.21.2 on page 188) takes an even more interesting approach, as key mixing is done both by simple XORs and an affine transform, i.e. the multiplication by a matrix that is derived from key material.

Most of the block cipher designs ever proposed and nearly all those that are in use today are SPNs. Some of the designs that are often presented as belonging to a different family, such as Feistel networks (Section 1.3) or Lai-Massey designs (Section 1.5 on page 36), are in fact just particular types of SPNs: Indeed, in the early history of block ciphers the term SPN was used in works describing Feistel designs, such as [Ada90, AT93], and until recently it was not uncommon to state that a cipher "*uses the Feistel structure [...] to implement the SPN*" [Ada97]. Today Feistel Networks and SPNs are treated as separate design families: when a cipher is clearly a Feistel or Lai-Massey design, it is denoted only as such and rarely referred to as an instance of a SPN, whereas SPN usually denotes a narrower class of ciphers such as bricklayer designs following the Wide Trail strategy (Section 1.4 on page 33). More examples of "classic" SPNs are given by SQUARE (Section 3.11 on page 158), Serpent (Section 3.17 on page 168), Rijndael (Section 3.20 on page 180), mCrypton (Section 3.25 on page 196) and PRESENT (Section 3.29 on page 204).

1.3 Feistel Networks (Luby–Rackoff Ciphers)

A SPN as described in Section 1.2 on page 25 often needs two different circuits or routines to perform both encryption and decryption. Even if they can be optimised somewhat (as we saw when comparing the two example designs of Figure 1.2 on the preceding page), the issue can still adversely impact performance tradeoffs in resource constrained environments. An elegant solution to this problem was presented in the late 60s by IBM cryptographer Horst Feistel. During the development of the Lucifer family of block ciphers he invented a structure that allows

iterated ciphers to be built in a way such that encryption and decryption are essentially the same operation.

Feistel's idea, in its simplest form, constructs a round as follows: The state is split into two halves, called **branches**; a function, called the **F-function**, is computed on the first half, called the **source** branch; the result of this operation is composed with the other half of the state, called the **target**, in a reversible way – the source branch remains unchanged until now; finally, the two branches are swapped.

For the composition of the transform of the source to the target, usually the XOR is used because the effect of XORing with some value is reversed by XORing again with the same value, but, as we mentioned in the previous section, other composition operations may be used.

Let the state be written as $L\|R$, as the concatenation of the two branches. The plaintext is the initial state $L_1\|R_1$. A round computes state $L_{i+1}\|R_{i+1}$ from $L_i\|R_i$ as follows:

$$\begin{cases} R_{i+1} = L_i \circledast F(R_i, k_i) \\ L_{i+i} = R_i \end{cases}$$

The ciphertext is the state $L_{r+1}\|R_{r+1}$ after r rounds. The round keys $k_1, \ldots, k_r$ are derived from the key though a key schedule. A graphical depiction of the cipher is given in Figure 1.3 on the facing page that includes an initial and final permutation (which have no influence on security) like the DES (Section 3.2 on page 127), and treats the key schedule as a black box.

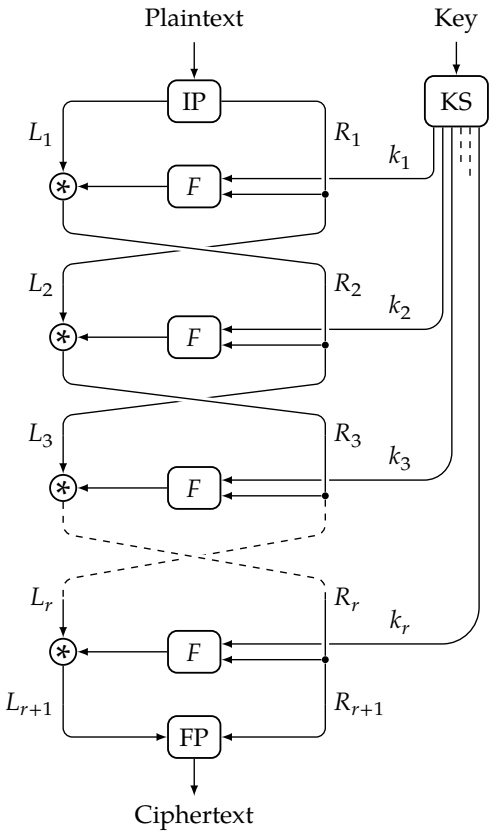
Feistel networks are just as a special case of SPNs: only half of the state is modified at each round, but in fact the state is a function of the whole previous state, and the permutation function is very simple. This also means that confusion and diffusion must be in fact applied to the source branch. Since confusion and diffusion are applied only to part of the state, so a higher number of rounds is necessary, in fact roughly twice as much – but these rounds are also potentially less expensive (roughly a half) since they have to operate only on half of the state. Moreover, the steps are easy to revert in the case the composition is the XOR, since only the round keys have to be fed in the opposite order to decrypt.

Finally, the F-function can be a simple SPN itself, so all the theory developed for SPNs is recycled – this is very well exemplified by the development of the Lucifer family of block ciphers (Section 3.1 on page 126) whose first instances were SPNs, and then the experience of the design of these SPNs was exploited to design the round function of the "final" Lucifer described in [Sor84].

In fact, the F-function itself can be an existing *cipher*, and therefore this construction can be seen as a simple way to double the block size of a cipher: the original cipher is used as the F-function, so it operates on *half* of the state of the new cipher, and the key-schedule is extended to generate different round keys for the various instances of the original cipher. Ladder-DES and DEAL (Subsection 3.19.1 on page 177) are two ciphers that use DES as the round function in this way, and MAGENTA (Subsection 3.19.6 on page 179) uses a quite complex 12-layer SPN as its F-function.

In general both confusion and diffusion are slower in a Feistel network than in a SPN, assuming they are equally well designed. This is general due to the fact that confusion and diffusion operations usually work only on a part of the state at each round, and will be discussed further

Figure 1.3: Schema of a Feistel Cipher



in Subsection 1.8.4 on page 53. Hence, a Feistel network usually needs more rounds than a SPN to achieve comparable levels of security at the same block and key sizes. On one hand Feistel rounds are also "lighter" (operating on smaller inputs) this usually is reflected in similar performance in simple implementations, with Feistel ciphers offering the advantage of unified code paths for encryption and decryption. On the other hand, the intrinsic higher parallelism in a SPN can be exploited well both in SW and HW to achieve a performance advantage in well optimized implementations.

But there is more: the power of this construction comes from the fact that the F-function needs not be bijective, with serious positive implications for performance, flexibility of design, and also for cryptographic theory. Michael Luby and Charles Rackoff analyzed the Feistel cipher construction [LR86], and proved that if the round function is a cryptographically secure pseudorandom function (where the round key is interpreted as the seed) then three rounds are sufficient to make the block cipher a pseudorandom permutation, while four rounds are sufficient to make it a "strong" pseudorandom permutation (which means that it remains pseudorandom even to an adversary who gets oracle access to its inverse permutation). Because of this very important result, Feistel ciphers are also called Luby-Rackoff ciphers.

Finally, David Goldenberg et al. [GHL+07a, GHL+07b] show that the Luby-Rackoff construction makes it particularly easy to construct tweakable block ciphers from conventional block ciphers: In the method we have mentioned to double the size of a block cipher, tweaking keys are added before the F-function and after the mix to the target branch.

1.3.1 A Taxonomy of Feistel Networks

There are many types of Feistel networks and generalisations, with the treatment going back to at least [ZMI89]. In Figure 1.4 on the facing page we show some of the most important types, where just one round is visualised for each variant. In place of the symbol $\oplus$ for the XOR we have used a "wildcard" composition symbol $\circledast$ in the figure because different operations can be used to mix the output of the F-function with the target branch. In particular, Hoang and Rogaway define an **arithmetic Feistel network** to be a Feistel network in which the composition with the target is a modular addition.

The represented types are:

1: Classic Feistel network, also called a balanced Type 1 Feistel network, where **balanced** means that the two branches have the same size.

   This is the topology of several notable ciphers such as DES (Section 3.2 on page 127), GOST (Section 3.4 on page 138), Blowfish (Section 3.9 on page 155), TEA and XTEA (Section 3.12 on page 159) CAST-128 (Section 3.7 on page 146), Twofish (Section 3.13 on page 162), Camellia (Section 3.18 on page 170), Simon (Subsection 3.36.1 on page 221) and many others.

2: Mitsuru Matsui's variation as used in various components of the ciphers MISTY-1, MISTY-2 and KASUMI (see Subsection 3.18.8 on page 174), is here given as the "L-network," i.e. the version where the F-function is on the target branch. There is also another version, the "R-network," studied by various authors, such as Yasuyoshi Kaneko, Fumihiko Sano, and Kouichi Sakurai in [KSS97] and Henri Gilbert and Marine Minier in [GM01], where the F-function is on the source branch.

   Note that in these constructions the F-function must be invertible.

3: Unbalanced Feistel network. The bit length of in_1 and out_0 is different from the bit length of in_0 and out_1. The output of a round can then be repartitioned before being input to the next round, but there are variants where there is no repartitioning, two different F-functions are used, with one "expanding" its input to be applied to the smaller branch and the other one "compressing" the input to apply the output to the larger branch, such as BEAR, LION and LIONESS (cf. Subsection 1.11.3 on page 68).

   The most extreme example of repartitioning unbalanced Feistel network is the Thorp shuffle, where the n bit state is divided into a 1 bit part and a $n-1$ bit part [MRS09] (see also [NR99] for an earlier realization of the usefulness of the Thorp shuffle in cryptography). The function F takes the $n-1$ bit part and outputs a single bit that is xored to the 1 bit part.

   NLFSR-based ciphers such as Keeloq (Subsection 3.3.3 on page 136) and KATAN (Section 3.31 on page 207) can be described in terms of unbalanced Feistel networks.

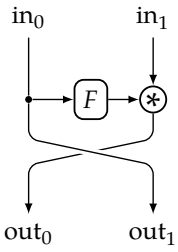
   Most unbalanced Feistel networks fall into the types that follow.

4: generalised Feistel Network (GFN) Type 1. It is one of the many ways to describe unbalanced networks: instead of having two branches of different width, we split the state into more than two branches (which, in the examples that follow, always have equal widths, but can of course still be unbalanced as well). The particular generalised Type 1 network depicted here has four branches, but there can be more than four branches.

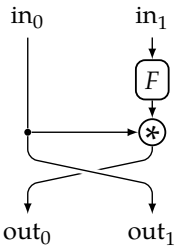
   CAST-256 (Section 3.7 on page 146) is a classic example of this design.

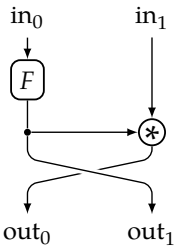
Figure 1.4: A (Partial) Taxonomy of (Generalised) Feistel Networks

1 : Classic (Type 1)

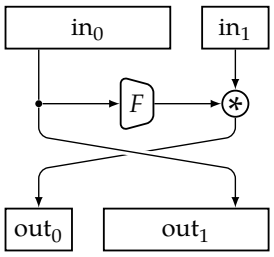


2 : Matsui's L- and R-Networks

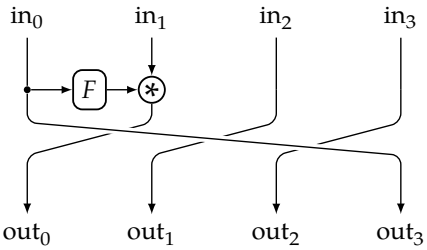




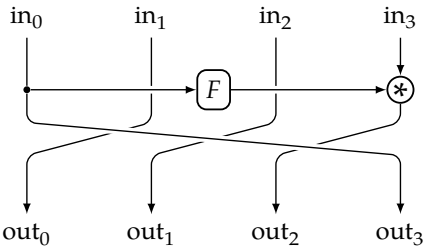
3 : Unbalanced Type 1



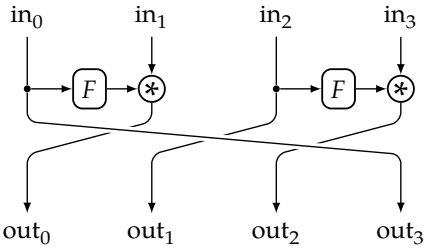
4 (a) : generalised Type 1



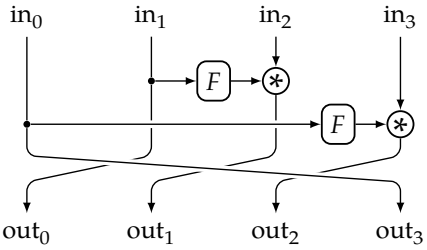
4 (b) : generalised Type 1 (inverse)



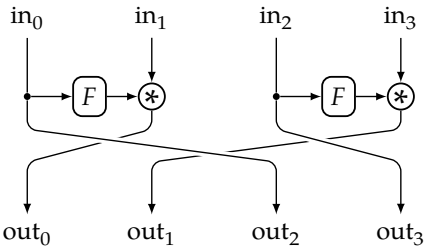
5 : Type 2



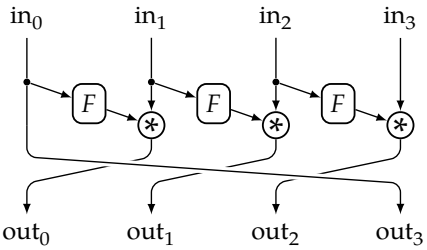
6 (a) : Nyberg Type



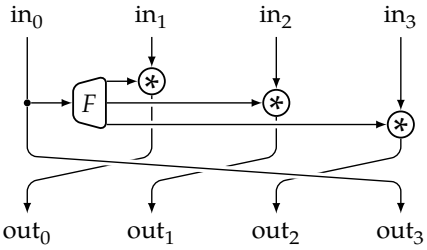
6 (b) : Nyberg Type (alternate view)



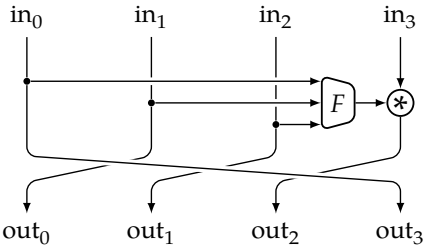
7 : (Zheng-Matsumoto-Imai) Type 3



8 : MARS Type 3 (Target-Heavy)



9 : Source-Heavy



Note that in the subfigure maked 4 (a) there are two branches that are nor source nor target, and that the target gets permuted onto the source branch. Other variants are possible, for instance where the output of the F-function is combined to the in_3 branch. In this case the source branch gets permuted onto the target, as represented in the subfigure maked 4 (b), and this variant is actually the *inverse* of the previous design, up to numbering of the branches.

The reader may now ask what happens if (in a four-branch Type 1 Feistel with branch permutation $(0\ 1\ 2\ 3) \rightarrow (3\ 0\ 1\ 2)$) the source and target branches are the first and third, respectively. It is easy to see, in this case, that the cipher splits into two non-intermixed ciphers, of two branches each, and therefore this case is not interesting.

5: Type 2, notable examples being RC6 (Section 3.15 on page 165), CLEFIA encryption and key schedule (Section 3.28 on page 201), and HIGHT (Section 3.27 on page 200). Note that it may seem at first sight that, if the branches are rotated in the opposite direction, we obtain a different topology, but in fact the two variants are equivalent.

6: The topology studied by Kaisa Nyberg in [Nyb96]. It is a variation of Type 2 with a different permutation of the branches. The subfigure marked 6 (a) follows Kaisa Nyberg's original depiction, whereas the subfigure marked 6 (b) shows how the permutation is different from the usual Feistel cyclic rotation of branches.

The significative difference between Type 2 and Nyberg Type is that in the former source and target branches are always swapped, whereas in the latter some source and target branches are permuted to branches of the same type.

7: Type 3, according to the definition given in [ZMI89]. To distinguish it from the next type, we call it also Zheng-Matsumoto-Imai Type 3 (see also Hoang and Rogaway [HR10a, HR10b]).

8: MARS Type 3, also called target-heavy, which is used in MARS (Section 3.16 on page 167).

9: Source-heavy, as used in RC2 (Section 3.5 on page 140), MacGuffin [BS94], SPEED [Zhe97], and in the LOKI97 key schedule (Subsection 3.19.5 on page 178).
This is a dual of the target-heavy topology.

The F-functions can be explicitly keyed, but for clarity we omitted arrows from the pool of round keys to the F-functions. The F-functions can be unkeyed internally by still be implicitly keyed by other transformations. Let us consider the classic Feistel network as an example. The cipher designer may want to add a round key to in_0 before the value is fed to the F-function. This value can be kept as the "new" in_0 (which then goes to become out_1) or be used only as an input to the F-function. Despite these different design choices, all these variants can be considered a classic Feistel network design.

The outputs of the F-functions can also be used to "tweak" other inputs (this should not be confused with the concept of tweakable block ciphers). For instance, let us consider RC6 (Section 3.15 on page 165). It has essentially a Type 2 topology. Referring to the graphics of Figure 1.4 on the previous page, the output of the F-functions applied to inputs in_0, resp. in_2, is not only XORed to in_1, resp. in_3, but it also determines the amount of a further rotation of the already XORed values of in_3, resp. in_1.

Henri Gilbert and Marine Minier in [GM01] develop for L- and R-networks analogue results to the work of Michael Luby and Charles Rackoff about Feistel networks. They prove that if

the round function is a cryptographically secure pseudorandom function, then: a L-network is not pseudorandom with three rounds but four, and it is strong pseudorandom with five rounds (but not four); a R-network is already pseudorandom with three rounds but, similarly to the L-network, it is strong pseudorandom only with five rounds. Makoto Sugita [Sug96] and Kouichi Sakurai [SZ97] already proved independently that four rounds are not sufficient to make L-networks strong pseudorandom.

Another set of variations on the Feistel design was discussed by Yasuyoshi Kaneko, Fumihiko Sano and Kouichi Sakurai at SAC 1997 [KSS97]. They consider all possible variants of two-branch Type 1 and Matsui L- and R-network rounds with keyed functions (or permutations) at three possible places: at the bottom, i.e. on the connection from split to composition (such as classic Feistel networks); on the target branch before the composition (such as in Matsui's L-network); or on the source branch before the split (as in the R-network). The bottom placement is denoted by the letter B. Since there are two possibilities for each of the three places (i.e. there is a keyed function or there it none), and one case is trivial, they study seven combinations: B, L, R, LB, RB, LR and LRB. There are of course countless ways to combine these variants with the network topologies of Figure 1.4 on page 31.

There of course more variations on the original Feistel design, and some of the above design classes are not necessarily distinct from each other. For instance, BEAR, LION and LIONESS (cf. Subsection 1.11.3 on page 68), introduced by Ross Anderson and Eli Biham in [AB96], can be also seen as unbalanced Feistel designs that alternate source-heavy and target-heavy steps. On the other hand, the Thorp Shuffle can be seen as an extreme source-heavy cipher where each branch consists of just one bit of the state.

1.4 The Wide Trail Design Strategy

Expanding and specialising Shannon's concepts, in 2001 Belgian cryptographers Joan Daemen and Vincent Rijmen introduced in [DR01, DR02a] the **wide trail strategy**. Their focus is on so-called *key-alternating block ciphers*, i.e. iterative block ciphers that can be written as

$$E[K] = \sigma[k_r] \circ \rho_r \circ \sigma[k_{r-1}] \circ \cdots \circ \sigma[k_1] \circ \rho_1 \circ \sigma[k_0]$$

where the unkeyed function ρ_i is the i-th round of the block cipher, $\sigma[k]$ denotes addition of round key k, $k_i = \psi_i(K)$ is the i-th round key, and ψ_i is the i-th step of the key schedule.

Daemen and Rijmen then further specialise to ciphers with a "$\gamma\lambda$ round structure," i.e. where the round function ρ is factored as the composition

$$\rho = \lambda \circ \gamma$$

of two functions γ and λ that are characterised as follows:

- γ is a **local** non-linear transformation, providing confusion.

  By local it is meant that any output bit depends on only a limited number of input bits and that neighbouring output bits depend on neighbouring input bits.

  A typical construction for γ is the so-called **bricklayer mapping** consisting of a number of invertible S-boxes transforming each a **word** or **bundle** of the state in parallel and independently of each other. The bundles and the S-boxes they pass through are identified with each

other. In this construction the S-boxes are usually implemented by a table lookup (where a bundle of the state is replaced by a fixed value) or via short straight line programs consisting of logical operators.

A bundle is usually a byte or a nibble, but other sizes are possible (bundles of 3, 6, 7 or 9 bits have been used).

Requirements for the confusion layer are discussed in Section 1.9 on page 55.

- λ is a linear mixing transformation providing diffusion. Often, the bundles of the state are considered as elements of a finite field $\mathbb{F}$, the state as a vector space V over $\mathbb{F}$, and λ is a $\mathbb{F}$-automorphism of V. In some cases a bit permutation is used instead.

  Such transformations are the subject of Section 1.8 on page 42.

Round key addition is usually performed by XORing a round key. The wide trail strategy however does not mandate the type of operation for the key mixing and does not deal with the key schedule. The resulting cipher is depicted in Figure 1.5 on the facing page.

The name "wide trail strategy" comes from the probability "trails" used in linear and differential cryptanalysis:

- In linear cryptanalysis (which will be described in detail in Section 2.2 on page 89 – here we just present a rough idea) the attacker tries to find a linear approximation of some output bits as a function of input bits, and he will follow the influence of the input bits through a "path" in an unrolled representation of the cipher. Such a path is called a **linear trail**.
- In differential cryptanalysis (again, this will be described later in Section 2.1 on page 71) the attacker tries to find a correlation between changes in the input bits and corresponding changes in the output – say he is considering pairs of inputs P and $P + \Delta P$ for a fixed ΔP and he searches for differences ΔC in the ciphertext, i.e. $\Delta C = E(P) - E(P + \Delta)$ that arise with a significant positive bias ϵ. In this case the attacker will follow the trail along which such a input difference ΔP propagates through the cipher to reach the output difference ΔC. Such a path (or trail) is called a **differential characteristic**.

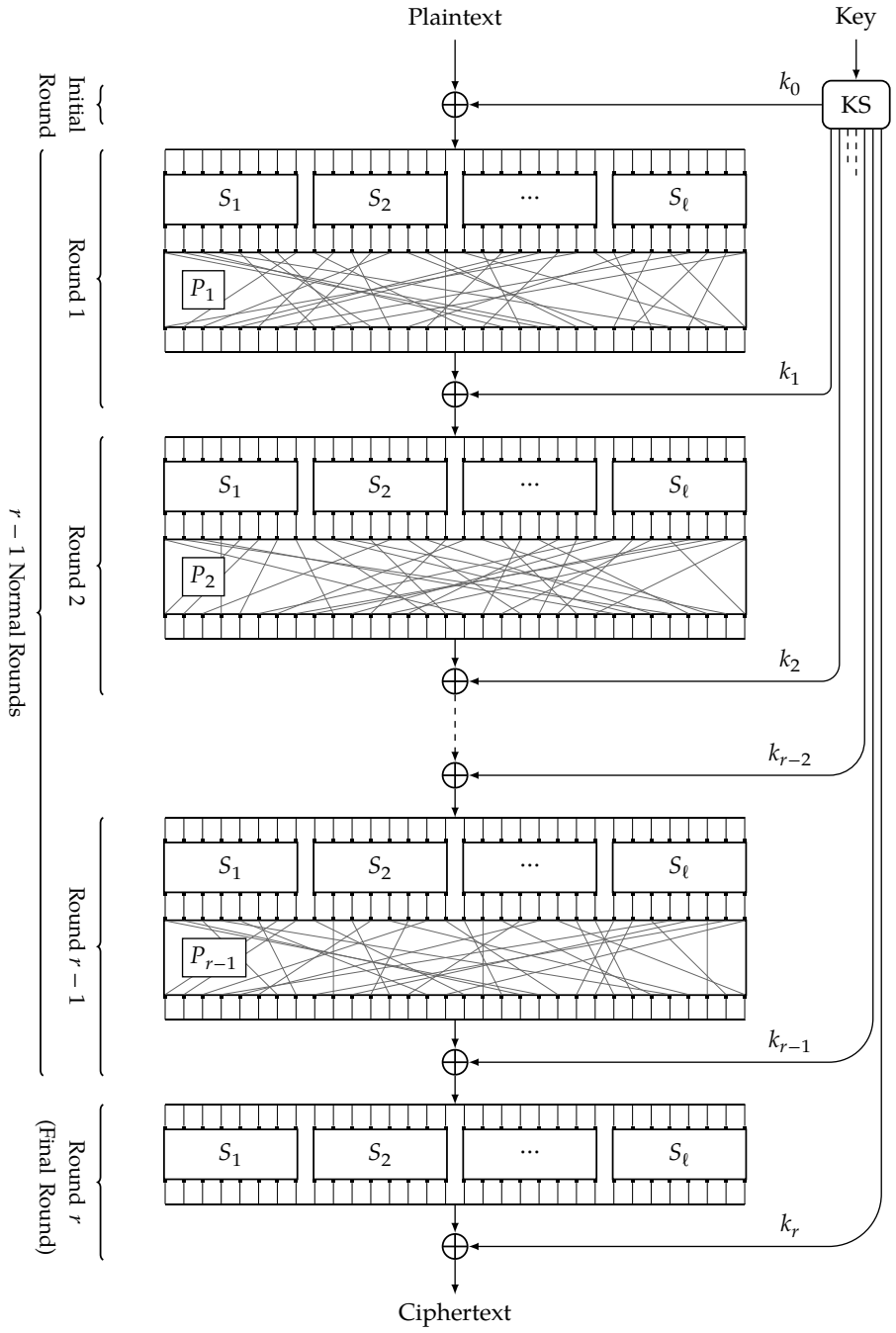
In both cases one usually starts observing how a small change or a small difference influences the intermediate values in a cipher. In the case of the wide trail design strategy, the propagation of these changes is observed through the bundles of the state. If an S-box is (guaranteed to be) affected by a change or differential (regardless of the state of the other S-boxes of the initial input), then this S-box is called **active**. The amount of active S-boxes for a given trail is called its **bundle weight**. A trail with large bundle weight is said to be **wide**.

An obvious desirable property is that these trails be as wide as possible. It is intuitively clear that in a properly designed cipher the amount of bundles influenced at each round increases. The fundamental intuition here is that *the wider these trails are, the harder they are to exploit*.

In particular, in the wide trail strategy, the substitution and diffusion layers can be examined on their own, and then combined together without worrying (too much) about how they will interact. In fact, a common design goal is that good diffusion should be attained independently from the characteristics of the S-boxes (provided the latter satisfy some minimal requirements).

This clear separation of roles also allows the designer to optimise the S-boxes for a given purposes without taking into account the interaction with the diffusion layer, at least initially.

Figure 1.5: A Wide Trail SPN (Without Final Permutation)



The hardest part of the design is usually the linear transformations, since they are the most involved. Indeed, in Daemen and Rijmen's own words [DR01]:

> *Instead of spending most of the resources on large S-boxes, the wide trail strategy aims at designing the round transformation(s) such that there are no trails with a low bundle weight. In ciphers designed by the wide trail strategy, a relatively large amount of resources is spent in the linear step to provide high multiple-round diffusion.*

Since in general constructing large linear transformations is very difficult, they are computationally expensive, and require considerable amount of code or gate equivalents to implement, Daemen and Rijmen suggest to construct the diffusion layer λ itself as the functional composition of simpler operations, while at the same time guaranteeing full diffusion only over two rounds instead of after just one. In particular, in SQUARE (Section 3.11 on page 158), and in several ciphers after that, we have

$$\lambda = \theta \circ \pi$$

where π is permutation of the words of the state (what is usually called a shuffle, see also Subsection 1.8.3 on page 52) and θ a linear transformation, usually chosen to be easily parallelisable: For instance, in many constructions V is a 16-dimensional vector space over $\mathbb{F}$, represented as the direct sum of four 4-dimensional vector spaces $V = V_0 \oplus V_1 \oplus V_2 \oplus V_3$, where π permutes the basis elements of the four subspaces V_i among each other by mapping each of the four basis vectors of a subspace to a different subspace; and θ operates independently on each one of the four V_i and in the same way. By means of these two operations full diffusion is guaranteed over two rounds. An implementation of this approach is shown in Section 3.20 on page 180 (the operations π and θ are called `ShiftRows` and `MixColumns`, respectively).

By choosing involutory π, θ and γ and by reversing the output of the key schedule, it is possible to use the same data processing part for encryption and decryption, as done for instance in KHAZAD (Subsection 3.21.2 on page 188). In fact, if encryption is defined as

$$E[K] = \sigma[k_r] \circ (\pi \circ \gamma) \circ \sigma[k_{r-1}] \circ (\theta \circ \pi \circ \gamma) \circ \sigma[k_{r-2}] \circ \cdots \circ \sigma[k_1] \circ (\theta \circ \pi \circ \gamma) \circ \sigma[k_0] \quad (1.1)$$

– note that the final round omits θ – then decryption is

$$\begin{aligned} D[K] &= \sigma[k_0] \circ (\gamma \circ \pi \circ \theta) \circ \sigma[k_1] \circ \cdots \circ \sigma[k_{r-2}] \circ (\gamma \circ \pi \circ \theta) \circ \sigma[k_{r-1}] \circ (\gamma \circ \pi) \circ \sigma[k_r] \\ &= \sigma[k_0] \circ (\pi \circ \gamma \circ \theta) \circ \sigma[k_1] \circ \cdots \circ \sigma[k_{r-2}] \circ (\pi \circ \gamma \circ \theta) \circ \sigma[k_{r-1}] \circ (\pi \circ \gamma) \circ \sigma[k_r] \\ &= \sigma[k'_0] \circ (\pi \circ \gamma) \circ \sigma[k'_1] \circ (\theta \circ \pi \circ \gamma) \circ \sigma[k'_2] \circ \cdots \circ \sigma[k'_{r-1}] \circ (\theta \circ \pi \circ \gamma) \circ \sigma[k'_r] \end{aligned} \quad (1.2)$$

where $\sigma[k'_t] \circ \theta = \theta \circ \sigma[k_t]$ for $1 \leqslant t \leqslant r-1$, which implies

$$k'_t = \theta(k_t)$$

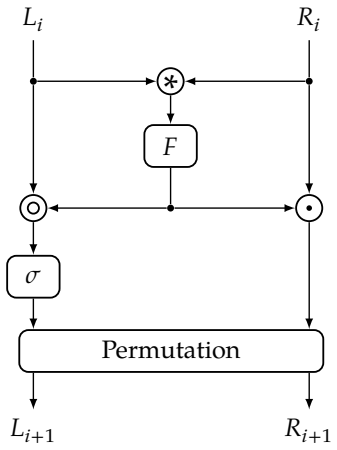
and thus a modified key schedule can deliver the sequence k_i with almost no overhead. For instance, if the key schedule functions ψ_i are affine, the overhead is a single application of θ.

If some components are not involutory, such as the substitution layer γ, then in the decryption path the code or circuit must accomodate the selection of one layer or its inverse. In this case Equation (1.1) remains unchanged but Equation (1.2) must be adapted.

1.5 The Lai-Massey Design

An obvious problem with Feistel networks is that part of the state is left unchanged – in other words confusion is not applied to the whole state – in each round. On the other hand, Feistel networks allow for more freedom in the choice of the round functions than more general SPNs, and make it easier to design ciphers where the data obfuscation path is the same for encryption and decryption.

Figure 1.6: The Lai-Massey Scheme (one Round)



IDEA (Section 3.6 on page 142) attempts to combine the advantages of both Feistel networks and more general SPNs. Despite being still, in principle, a SPN, its design is original enough to warrant its own terminology: some authors indeed call IDEA and similar ciphers a **Lai-Massey Design**, which we represent in Figure 1.6, after the names of its inventors.

With respect to Feistel networks, there are two important differences: the two halves of the state are first combined before being fed to the F-function, and then the output of the F-function is combined with *both* halves of the state. The purpose is to accelerate both diffusion and confusion.

In order to be able to reverse this scheme the input to F must be reconstructed from the outputs. For the operations as we pictured them, it is necessary that

$$L \circledast R = (L \circledcirc \Delta) \circledast (R \odot \Delta)$$

is satisfied for all L, R and Δ. This is clearly satisfied if all three operations $\circledast$, $\circledcirc$ and $\odot$ are the bitwise XOR, but other combinations of operations are possible. For instance we can take $\circledast$ and $\circledcirc$ to be integer addition, while $\odot$ is integer subtraction.

We note that $L \circledast R$ is a constant, therefore a permutation layer is added after the F-function, and often an additional operation σ (called an orthomorphism) is necessary to avoid $\circledast$-differentials that an attacker may otherwise be able to exploit. The permutation layer alone, if it can guarantee good diffusion over sufficiently many layers, may make the orthomorphism σ superfluous – other designs may include σ but no permutation.

Further Lai-Massey ciphers include Akelarre [AdlGMP96], MESH [JRPV03], and FOX, known also as IDEA NXT (Section 3.23 on page 193).

Serge Vaudenay studies in [Vau99b] security properties of Lai-Massey schemes and proves in that for Lai-Massey schemes with an orthomorphism the same security bounds as for Luby-Rackoff constructions, i.e. if the F-function is a a random function, then three rounds are sufficient to make the block cipher a pseudorandom permutation, while four rounds is sufficient to make it a strong pseudorandom permutation.

1.6 The Even-Mansour Schemes and the FX Construction

In 1991, Shimon Even and Yishay Mansour considered the question: *What is the simplest possible construction of a block cipher which is provably secure in some formal sense?* To answer this question they came up with a minimalistic design [EM91, EM97], inspired by Ron Rivest's *FX construction*. Let us introduce the latter first.

The FX Construction was introduced by Ron Rivest in order to strengthen DES and, in general, any existing cipher: a block cipher E^X is built with $(2n + \ell)$-bit key and n-bit block from a block cipher E with ℓ-bit key and n-bit block by xoring the input and output of E with a pre-whitening key and a post-whitening key:

$$E^X_{k_0,k_1,k_2}(x) := E_{k_1}(x \oplus k_0) \oplus k_2 \ .$$

In the case of DES the resulting cipher is called DESX (see Subsection 3.2.9 on page 132). In general, the name of the construction refers to "F", i.e. any function, and "X", meaning that the function F is e*X*tended.

The Even-Mansour (EM) scheme instead uses whitening keys – each as large as the plaintext block – around *a publicly known, fixed, non-keyed random permutation* in the middle. The resulting scheme is extremely simple: To encrypt an n-bit plaintext, XOR it with a n-bit key, apply a publicly known permutation, and XOR the result with a second n-bit key. It is easy to see that the scheme is minimal in the following sense: if one of the two whitening steps or the permutation is omitted the reduced cipher is easy to break.

The main difficulty, of course, is to construct the permutation, which must be a pseudorandom function, but even assuming that this is the case, Joan Daemen immediately attacked this scheme [Dae91] with a chosen ciphertext attack of complexity $O(2^{n/2})$ – we note that the scheme needs $2n$ key bits. Orr Dunkelman, Nathan Keller and Adi Shamir prove in [DKS12] a bound $\Omega(2^n/S)$ on attacks on the Even-Mansour scheme where S is the number of known plaintexts, thus matching Joan Daemen's upper bound. They also show that one key can be used in place of two (of course both ends must be whitened), hence only n key bits are needed to attain the same security bounds.

Hence, even assuming the central permutation is ideal, the security that can be attained is at most $2^{n/2}$ – which may be more than enough in some cases. Better security bounds can be attained by alternating more key additions and state permutations – we will discuss this in Section 2.6.5 on page 113. Such generalised Even-Mansour scheme suggest that not all rounds in an interative key alternating-design (cf. Section 1.4 on page 33) must necessarily be keyed, even though one must be careful not to lose security. In fact some block cipher designs do not key every round: MARS (Section 3.16 on page 167) is one such example, and Threefish (Section 3.30 on page 205) and the lightweight design LED (Subsection 3.37.4 on page 227) also apply key mixing only every four rounds.

The differences between the EM-scheme and the FX-construction are fundamental: the internal cipher of the FX-construction is not assumed to be a pseudorandom function, and in general it is not (it is often a cipher for which the existence of distinguishers, even practical ones, has been proved), and it is keyed. But the attacks on the EM scheme can still be mounted, for instance by fixing the internal ℓ-bit key and mounting a generic attack on EM schemes on the resulting function.

As a result, Joe Kilian and Phillip Rogaway [KR96b, KR01a] show that, if the core cipher E is ideal, the FX construction achieves $(\ell + n - 1 - \log T)$-bit security where T is the number of pairs of inputs and outputs for E known by the attacker.

The FX Construction is not only a very inexpensive way to strengthen an existing cipher (against brute force attacks, differential and linear cryptanalysis) but it has become a design criterion on its own: Nowadays several ciphers are designed *including* a key whitening step.

It must be kept in mind that the construction gives a tradeoff security, which it inherits from the EM scheme: in order to attain a classical level of $\ell + n$ bits of security (i.e. there are no attacks taking less time than $2^{\ell+n}$, including data collection and preprocessing) a different design strategy is required.

A noteworthy form of key whitening – as introduced by the cipher FEAL, followed by Khufu and Khafre – uses keys derived from the initial key material. In a Feistel cipher, this can increase security by concealing the specific inputs to the first and last round functions. This version of key whitening offers no additional protection from brute force attacks, but it can make other attacks more difficult.

Some ciphers that employ key whitening are: FEAL (Subsection 3.3.4 on page 137), MARS (Section 3.16 on page 167), Twofish (Section 3.13 on page 162), RC6 (Section 3.15 on page 165), Camellia (Section 3.18 on page 170), HIGHT (Section 3.27 on page 200), CLEFIA (Section 3.28 on page 201), PRINCE (Section 3.35 on page 215) and Piccolo (Subsection 3.37.2 on page 226).

Pierre-Alain Fouque and Pierre Karpman in [FK13a, FK13b] describe a stronger version of the FX construction where in place of simple key whitening more complex keyed functions are used. Such functions, called decorrelation modules, are described in Subsection 1.10.3.

1.7 The Key Schedule

The key schedule is a crucial component of a block cipher – and probably the least well understood. A weakness in it can break down completely designs which are otherwise quite clever: It can give rise to classes of weak keys (Section 2.5 on page 105), i.e. keys for which the cipher is easy to break, or make the cipher amenable to attacks specifically targeting relations between keys (Section 2.6 on page 105), MITM attacks and their variants (Section 2.4 on page 94), and perhaps also zero correlation cryptanalysis (Subsubsection 2.2.8.2 on page 93). At the same time, there are strong ciphers with very simple key schedules.

In fact, Rijmen and Daemen already stated in the AES design book [DR02b, P. 77] that: "*There is no consensus on the criteria that a key schedule must satisfy*."

1.7.1 The Role of the Key Schedule

A key schedule can be used in various ways in a block cipher:

1. In a product cipher it can simply specify the subkeys that are mixed to the state at each round. Classic examples are given by the DES (Section 3.2 on page 127) and most SPNs, including AES (Section 3.20 on page 180).

2. It may be used to initialise some fixed elements of a cryptographic transform, such as substitution tables. This is done, for instance, in Blowfish (Section 3.9 on page 155) and Twofish

(Section 3.13 on page 162), as well as in the block ciphers C [BF06a] and KFC [BF06b], designed by Thomas Baignères and Matthieu Finiasz and briefly described in Subsection 1.10.3 on page 63.

3. Its output can be used to dynamically select functions from fixed families to be used in certain places of the cipher. These families of functions can be S-boxes from a list, or variable rotations. This was the approach used in some early ciphers such as Lucifer (Section 3.1 on page 126), the DES, CAST (Section 3.7 on page 146), as well as in variants of C and KFC with precomputed families of S-boxes.
   The perhaps most extreme example is given by FROG (Subsection 3.19.3 on page 178), that essentially expands the key into a program describing the data obfuscation path.

At least in the first of these three cases, the key schedule *seems* to have a subordinate role to the data obfuscation path: if the latter presents weaknesses, the former cannot repair it. This is what ultimately breaks DES with any key schedule, *even with independent round keys* [BS93]. Also, the principle of confusion states that the key bits must be combined with the plaintext bits in a as complex way as possible. In an iterative block cipher with round key mixing between the rounds, the use of diffusion and confusion to destroy any linear relation between plaintext and ciphertext already partially contributes to the desired mixing of the key bits with the plaintext, which intuitively should be "as good" as the non-linearity between plaintext and ciphertext. But, it is not clear how complex the key schedule must be in order to achieve a sufficient diffusion of the key bits in combination with the round functions. A first study of this is done by Jialin Huang and Xuejia Lai in [HL12, HL14], where they propose a criterion to evaluate for necessary key schedule diffusion that takes into account the round diffusion.

Even though a theoretically good data obfuscation path is necessary to achieve security (as DES shows), it is not sufficient: for instance, a periodic key schedule of length $t \geqslant 1$ reduces the security of the full cipher to that of a t rounds-reduced one, *regardless of the actual length of the cipher* because of slide attacks (Subsection 2.6.2 on page 107). On the other hand, slide attacks require the ability to mount an *efficient* known plaintext attack on the t rounds-reduced version of the cipher, which in turn means that if t is a number large enough to make such an attack sufficiently slow (even though, technically, the t rounds-reduced version of the cipher may be broken), *then* using that periodic key schedule may still be fine.

1.7.2 Construction of the Key Schedule

We have just mentioned that the simplest key schedule, i.e. the constant key schedule, is fundamentally weak in simple product ciphers, because it reducing the security of the cipher to that of a single round. One possible opposite extreme of the spectrum is represented by independent round keys, or the use of a complex key derivation function – a strong PRNG (pseudo-random number generator) – to derive the keys from a seed.

Most proofs of security for block ciphers rely on strong assumptions about the independence of the round keys, or more generally about the pseudo randomness of the output of the key schedule process – an approach that is fruitful in cryptanalysis, as proved by the usefulness of the theory of Markov ciphers (cf. Subsection 2.1.2 on page 76). Using independent round keys would make the effective key length very big, while mixing only parts of it in each round and at the same time breaking the principle of confusion – rendering in fact the cipher *less* secure with respect to the effective key length Also, the resulting cipher would be trivially susceptible

to a meet-in-the-middle attack (Section 2.4 on page 94) with complexity exponential in half of the effective key length. Hence, depending on the actual structure of the data obfuscation path the actual security level could be even lower.

Therefore many designs put a strong emphasis on complex key schedules, some of which approach the complexity of pseudo-random number generators (PRNGs), seeded with a master key that has a length matching the desired security level. The key schedules of KHAZAD (Subsection 3.21.2 on page 188), RC5 (Section 3.18 on page 157), RC6 (Section 3.21 on page 166), and Blowfish (Section 3.9 on page 155), are such examples. Indeed, some block ciphers even *do* use a cryptographically strong PRNG as their key schedule, such as "C" and "KFC." With a complex key schedule, the hope is that confusion, as expressed in key avalanche (avalanche effect of the key bits), can be achieved also with a relatively simple data obfuscation path.

On the other hand, the intuition expressed earlier, that combining a data obfuscation path providing good confusion together with a simple key schedule (but still eschewing trivial key schedule design mistakes), suggests that the key schedule construction issue itself may be overblown. A proof of this might be indeed represented by the later members of the SAFER family (Section 3.8 on page 148) and by IDEA (Section 3.6 on page 142).

So there are two extremes, represented by trivial key schedules and very complex ones, where the latter give a more solid foundation at the price of reduced key agility. And it is perhaps because of this consideration that some design methodologies, such as the wide trail design strategy, that favour the design of an "ideal" data obfuscation path, do not include key schedule design techniques,

There are several ways to construct a key schedule. Some common strategies (with blurry mutual boundaries) are:

1. Use a linear key schedule, combining bit permutations and extractions and other linear operations. This is done in Lucifer, DES, KeeloQ (Subsection 3.3.3 on page 136) GOST (Section 3.4 on page 138), IDEA, Skipjack (Section 3.14 on page 164), and Bel-T (Section 3.33 on page 212).

2. The same as the previous method, but also masking with fixed constants, such as in the SAFER family (Section 3.8 on page 148), SQUARE (Section 3.11 on page 158), SIMON (Section 3.36 on page 220), and Piccolo (Subsection 3.37.2 on page 226).

   PRINCE (Section 3.35 on page 215) is an extreme example because each round key is just the master key XORed with a different round constant.

3. Sometimes, a non-linear component is added to the previous two processes, such as in the AES, PRESENT (Section 3.29 on page 204), and SPECK (Section 3.36 on page 220).

4. Encrypt the master key using a block or stream cipher with fixed keys, and use the intermediate states of the block cipher or the output of the stream cipher as the subkeys.

   The same algorithm as the data obfuscation path (possibly shortened) can be reused to generate some or all of the subkeys, as in Camellia (Section 3.18 on page 170), or NOEKEON (Subsection 3.21.4 on page 191); or a different one, possibly reusing some of the components of the the data obfuscation path, as in FEAL (Subsection 3.3.4 on page 137), KHAZAD, RC5, RC6, Twofish, ICEBERG (Section 3.24 on page 195), SEA (Section 3.26 on page 197), CLEFIA (Section 3.28 on page 201), and KLEIN (Subsection 3.37.1 on page 224).

Two more extreme cases are Blowfish, that repeatedly uses the encryption routine to generate the key expansion, and SHARK (Subsection 3.21.2 on page 188), whose key schedule is based on cipher itself in CFB mode. (We note that in SHARK key mixing is done both by simple XORs and more complex affine transforms.)

It has not been conclusively proved that key schedule plays a part in providing strength against linear and differential cryptanalysis. Some experimental evidence is given in Knudsen and Mathiassen in [KM04]: using a toy cipher, they observed that complex schedules can reach a uniform distribution for the probabilities of differentials and linear hulls faster than those with poorly designed key schedules.

1.8 Diffusion

The state of a SPN can usually be viewed as an array of smaller words or bundles. It is these bundles that are usually transformed by some non-linear operation such as S-boxes, then combined with a round key, and permuted among each other or, if they are interpreted as elements of some algebraic structure (for instance as a vector space over a finite field) even composed with each other using some kind of linear operation.

Suppose an attacker changes a bit in the input. The corresponding S-box ideally changes on average a half of its bits – this means that, other than confusion, an S-box already performs a limited diffusion process. These altered bits are then further *diffused* by the diffusion layer and thus affect the input to one or more S-boxes in the following round. Ideally, even more S-boxes are affected in the following round and so on. The affected S-boxes are called **active** S-boxes and, intuitively, to increase both efficiency and security it is desirable that the number of active S-boxes becomes maximal after as few rounds as possible. In other word, we want to attain good *diffusion*. There are different techniques to achieve it, and we shall present the most significant ones.

1.8.1 Bit Permutations

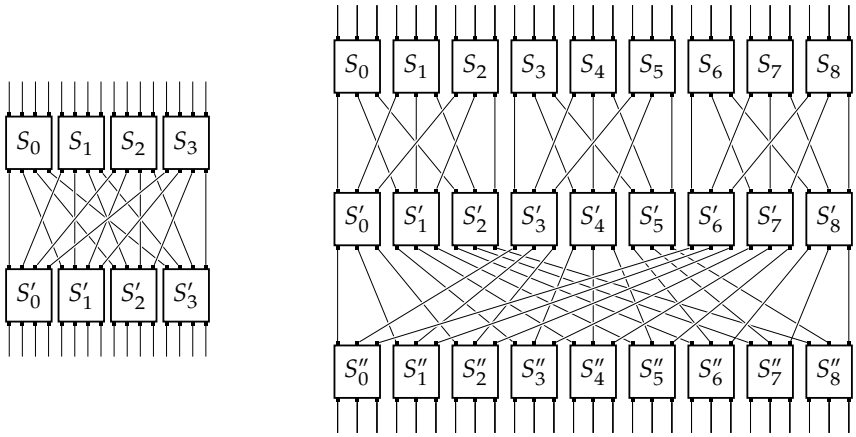
Several recent ciphers, such as PRESENT (Section 3.29 on page 204) and PRINTcipher (Section 3.32 on page 209) use a simple bit permutation of the S-box outputs to achieve diffusion. Therefore additional conditions on the S-boxes are placed to improve the avalanche effect. (For instance single input differences should not trigger the same single input difference in another S-box in the next round).

However, these ideas have been developed very early in the history of block ciphers, and bit permutations were commonplace in early designs, such as the Data Encryption Standard 3.2 on page 127 and GOST 28147-89 3.4 on page 138.

One of the earliest treatments of bit permutation networks in SPNs is due to John Kam and George Davida [KD79], who introduce the notion of **completeness**: *A bijective function* $f : \{0,1\}^n \rightarrow \{0,1\}^n$ *(for instance a S-box) is said to be* ***complete*** *if, for every* $i, j \in [0..n-1]$*, there exist two n-bit vectors* x_1, x_2 *such that* x_1 *and* x_2 *differ only in the* i^{th} *bit and* $f(x_1)$ *differs from* $f(x_2)$ *at least in the* j^{th} *bit. Similarly, a keyed cipher is said to be* ***complete*** *if it is complete function for all keys.* (It is easily seen that the Strict Avalanche Criterion is a strengthening of this property.)

Kam and Davida also show how to construct complete ciphers. The fundamental idea consists

Figure 1.7: Complete Tree-Structure SPNs Following the Kam-Davida Construction



in alternating substitution layers, where the S-boxes are assumed to be complete themselves, with specially defined wired permutation layers. Each output bit in a layer should be wired to input bits to distinct (but not necessarily different) S-boxes at the next layer. The corresponding graph, with the S-boxes as nodes and the wirings as edges directed from an S-box in a layer to an S-box in the next layer, should connect each input bit to the SPN to each output bit through exactly one path, in other words it should be a polytree, i.e. be acyclic. Usually the S-boxes are either keyed or the key is mixed before the S-Box, one key bit per input bit.

Each SPN output bit may thus be viewed as a tree function of the SPN input bits, where each tree function is composed of S-boxes. Similarly, the network inputs may be viewed as tree functions of the network output bits. Therefore, several authors follow Howard M. Heys and Stafford E. Tavares in [HT93, HT95] and refer to SPNs constructed using the Kam and Davida methodology as **tree-structured SPNs** or TS-SPNs.

Kam and Davida provide a concrete construction method, by means of which, a complete SPN with block size of $n = m^t$ bits can be built using t substitution layers and $t - 1$ bit permutation layers. Examples for $(m, t) = (4, 2)$ and $(3, 3)$ are shown in Figure 1.7. However, there is a problem with their construction: it can be shown that if a single input bit is changed, the probability of an output bit changing will always be 2^{-t}. This was shown in [Web85] and termed avalanche damping (see also [WT85]).

John Kam and George Davida obtained US Patent 4,275,265 on their design in 1981. A similar idea, where *pairs* of bits are permuted as bundles, but otherwise in an entirely analogous way, is described in Thomson-CSF's US Patent 4,751,733 (filed in 1986, lapsed).

It is these ideas that influenced the wiring used in PRESENT (where completeness is achieved over two rounds, as it can be seen from Figure 3.33 on page 204) and, to a lesser extent, PRINTcipher. The design of composite S-boxes such as those of Khazad (Subsection 3.21.2 on page 188) and Anubis (Subsection 3.21.3 on page 190), or of ICEBERG (Section 3.24 on page 195) is, on the other hand, more similar to the scheme of the Thomson-CSF patent.

Simple bit permutations are particularly efficient in hardware, as they amount to just wiring. Some exceptional cases may be implemented efficiently also in software. However, they suffer from the problem that an output bit may at most influence a single input bit of the next layer of the cipher, and therefore diffusion is slower than with more sophisticated linear layers. and

more rounds may be necessary than with more complex linear diffusion layers. As a consequence, bit permutation based diffusion layers are more suitable to very compact hardware implementations than to efficient implementations (mostly in software, but also in hardware).

A historical remark: Kam-Davida construction have been used repeatedly to build example ciphers for the purposes of explaining cryptanalysis (as in Howard M. Heys's excellent tutorial on linear and differential cryptanalysis [Hey02]) or to show new attacks (for instance, the data obfuscation path of SmallPRESENT-[4], as defined in [Lea10] and then repeatedly used, for instance in [BG10] and [AL12], is the first Kam-Davida construction of Figure 1.7 on the preceding page). However, this SPN is almost always used or rediscovered without attribution.

1.8.2 Diffusion Layers Based on Linear Algebra

For simplicity let us assume that the words of the state are elements of a module V over a ring R (V can be the ring itself) so that its elements can be added to each other and multiplied by elements of R (scalars). More generally, we can consider modules over rings instead. The state is this just a n-tuple v of elements of V and we consider the following type of transformation: v is multiplied by a $n \times n$ matrix M over R. Multiplication by M should be invertible at least in the case where the diffusion is used directly in a "classic" SPN.

Note that the diffusion layer can sometimes be described as a matrix even when this operation is described in a different way. For instance, in SAFER (Section 3.8 on page 148) the diffusion layer (which can be seen in Figure 3.14 on page 149) is constructed from simpler operations over the ring $R := \mathbb{Z}/256\mathbb{Z}$ of integers modulo 256, but a matrix representation is clearly possible. SAFER is perhaps the oldest cipher to use a linear diffusion layer in place of a permutation of the bits of the state to achieve diffusion, so we want to have a closer look at its design. The structure of the diffusion layer can be written as TPTPT, where P means permutation (of the bundles) and T is the layer of pseudo-Hadamard transforms (PHT). T transforms all pairs of adjacent bundles x_i, x_{i+1} with i even using the following PHT:

$$\begin{pmatrix} x'_i \\ x'_{i+1} \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_i \\ x_{i+1} \end{pmatrix} \pmod{256} \ .$$

Hence, the T layer is represented by the 8×8 block diagonal matrix A with the matrix $\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$ on the diagonal four times. The permutation of the bundles is called a **shuffle**. It is the permutation

$$(0\,2\,4\,6\,1\,3\,5\,7)$$

and corresponds to the matrix

$$B = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \pmod{256} \ .$$

Hence, the R-matrix representing the entire diffusion layer of SAFER is

$$M = A \cdot B \cdot A \cdot B \cdot A = \begin{pmatrix} 8 & 4 & 4 & 2 & 4 & 2 & 2 & 1 \\ 4 & 2 & 2 & 1 & 4 & 2 & 2 & 1 \\ 4 & 4 & 2 & 2 & 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 & 2 & 2 & 1 & 1 \\ 4 & 2 & 4 & 2 & 2 & 1 & 2 & 1 \\ 2 & 1 & 2 & 1 & 2 & 1 & 2 & 1 \\ 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \pmod{256} .$$

Let us now consider the effect of this matrix on the state, interpreted as a vector of length eight over the ring R. Since the operator defined by M is linear, in order to determine how differences in the input propagate it suffices to consider the differences as relative to the zero vector, i.e. to study the images of individual vectors. Most vectors v of weight one are mapped to vectors $v \cdot M$ of weight eight, but not all, for instance, $v = (32\ 0\ 0\ 0\ 0\ 0\ 0\ 0)^t$ is mapped to a vector $v \cdot M = (0\ 128\ 128\ 64\ 128\ 64\ 64\ 32)^t$ of weight seven, and the image of $v' = (128\ 0\ 0\ 0\ 0\ 0\ 0\ 0)^t$ has weight just one: $v' \cdot M = (0\ 0\ 0\ 0\ 0\ 0\ 0\ 128)^t$. Intuitively, this means that some inputs or some differences do not diffuse well through the layer. Catastrophic consequences are in fact avoided only by the fact that SAFER uses good S-boxes and the fact that the single bundle difference on the eighth vector element will completely diffuse in the following round.

1.8.2.1 Multipermutations and MDS Matrices

The obvious question is: what are the matrices that guarantee the *most complete* diffusion? The question is somewhat ill posed because a desirable property of any component of a block cipher is its fast evaluation. Hence, a good diffusion matrix must strike the right balance between good diffusion and fast evaluation: a less perfect but much faster diffusion layer could still lead to a cipher that it faster and not less secure than another cipher making use of an ideal, but slower, diffusion layer. Also the question of performance is per se difficult to formalise: for instance, a sparse matrix is not necessarily good if some entries represent elements which are expensive to multiply with.

This said, the first problem remains that of measuring the quality of diffusion and determining optimal matrices – performance considerations, including compromises, come later.

To address this first problem, Serge Vaudenay suggested [Vau94] (generalising previous work by himself and Claus-Peter Schnorr [SV94]) to use multipermutations: *Given an alphabet $\mathscr{M}$ and integers n, s, a a* ***(s, n)-multipermutation*** *over the alphabet $\mathscr{M}$ is a function f from $\mathscr{M}^s$ to $\mathscr{M}^n$ such that two different $(s+n)$-tuples of the form $(x, f(x))$ cannot collide in any s positions.* Serge Vaudenay in particular first observed that the PHT in SAFER (and hence the whole diffusion layer) is not a multipermutation.

To construct multipermutations, if the alphabet is representable as a finite field, he suggested to use (the redundancy part) of **MDS matrices**, i.e. matrices of MDS (maximum distance separable) codes, which are the codes which reach the Singleton bound: In other words a $n \times s$ matrix M over a finite field $\mathbb{F}$ is an MDS matrix if it is the transformation matrix of a linear transformation $f : \mathbb{F}^s \to \mathbb{F}^n$, $x \mapsto Ax$ with the following property: if x and x^* differ in exactly t components, then $f(x)$ and $f(x^*)$ must differ in at least $n - t + 1$ components. The latter property is called

perfect diffusion. Vaudenay also showed how to exploit imperfect diffusion for cryptanalysis (as in the case of reduced rounds of SAFER with suboptimal S-boxes, cf. Section 3.8 on page 148).

Now, to see why this is optimal and indeed a desirable cryptographic property, let us assume $s = n$ and consider first the case of a single changed input word. Then the change should spread to all outputs – a property that, as we have seen at the beginning of this section, is not satisfied by the SAFER diffusion later. If we now change two words, we may always choose them to that *one* of the outputs of the linear transformation is equal to the corresponding input (this is a simple linear algebra exercise) so we cannot do better than requiring that at least $n - 1$ inputs are changed.

Note however, that the MDS condition, for $s = n$ is stronger than being invertible (i.e. non-singular), as exemplified by the identity matrix, and non-singularity is of course not a sufficient condition for being an MDS matrix, since it applies only to square matrices. Non-singularity, however, gives a way to characterise an MDS matrix: Theorem 8 (page 321) of [MS77] states that *a matrix is an MDS matrix if and only if every square sub-matrix is non-singular.* In particular, a MDS matrix cannot have zero entries.

The first notable cipher to use MDS matrices for diffusion is Shark [RDP+96], designed by Vincent Rijmen, Joan Daemen, Bart Preneel, Antoon Bosselaers and Erik De Win (cf. Subsection 3.21.2 on page 188). For the design of the diffusion layer, the m-bit outputs of the S-boxes are considered as elements of $\mathbb{F}_m$. The diffusion layer takes n m-bit values as input, and gives n m-bit outputs. Such a vector or n m-bit values represents the state of the cipher. Joan Daemen defines optimal diffusion using the *branch number* [Dae95]: The **branch number** $\mathscr{B}$ of an invertible linear mapping θ is

$$\mathscr{B}_\theta = \min_{a \neq 0} \left(\mathsf{wt}(a) + \mathsf{wt}(\theta(a)) \right)$$

where $\mathsf{wt}(a)$ is the Hamming weight of a (here a is considered as a tuple of elements over some algebraic structure - the bundles - so by Hamming weight it is understood the number of non-zero elements of the tuple). $\mathscr{B}_\theta$ gives a measure for the worst case diffusion: it is a lower bound for the number of active S-boxes in two consecutive rounds of a linear trail or a differential characteristic.

Note that $\mathsf{wt}(a) \leqslant n$, for every choice of θ; if $\mathsf{wt}(a) = 1$, this implies that $\mathscr{B}_\theta \leqslant n+1$. An invertible linear mapping θ for which $\mathscr{B}_\theta = n+1$ is called **optimal**. If the vector a represents, for instance, an input differential, we see how this definition of optimality corresponds to the multipermutation property. In fact, it follows directly from the definitions of branch number and of MDS codes that the generator matrix of an MDS code defines an optimal linear transformation θ. Furthermore, this θ must be invertible.

Other examples of block ciphers that use MDS matrices for diffusion are SQUARE [DKR97] (see Section 3.11 on page 158), Twofish (Section 3.13 on page 162), the AES contest winner Rijndael (Section 3.20 on page 180), Hierocrypt [OMSK00], IDEA NXT (Section 3.23 on page 193), Clefia (Section 3.28 on page 201), Piccolo (Subsection 3.37.2 on page 226), and LED (Subsection 3.37.4 on page 227), MDS matrices are used also in the stream cipher MUGI [WFY+04] and in the cryptographic hash function WHIRLPOOL [BR11a].

It is worth noting that the entries in the MDS matrices are usually chosen as to be elements of low Hamming weight, in order to make multiplication by them as inexpensive as possible.

This is often done by exhaustive search within certain classes of MDS matrices, such as generator matrices of Reed-Solomon codes. Also, since a MDS matrix cannot have zero entries, the desirable type of sparseness is a small amount of entries not equal to one.

Even multiplication by low Hamming weight elements of a finite field can be too expensive for some applications. Therefore some ciphers, such as mCrypton (Section 3.25 on page 196), define their diffusion matrix in a different, ad hoc, way. The corresponding study of the diffusion properties is also ad hoc and the S-boxes have to be chosen carefully.

We shall return to the problem of constructing efficient MDS diffusion layers in Subsubsection 1.8.2.3 on the next page.

Another problem arises with states that consists of many words, for instance 16, as in 128-bit SPNs with 8-bit S-boxes (or with 64 bit SPNs that use 4-bit S-boxes), namely that the diffusion matrix becomes too large - in the examples we just made it would be a 16×16 matrix. The solution adopted in ciphers such as SQUARE and Rijndael, with 16 words (of one byte each) is to only apply diffusion to each of four blocks of four words independently during a round - and then to simply permute the words in such a way that full diffusion will be completed in the *following* round. Therefore instead of the multiplication of a diffusion matrix times a column vector, in such ciphers the diffusion operation is implemented as a multiplication of two matrices: the diffusion matrices and a matrix whose columns are segments of the state. In mathematical notation this is described in Section 3.20 on page 180.

1.8.2.2 Types of MDS Matrices

We recall that what we called a MDS matrix M is, formally, the *non-systematic (or redundancy) part of the generator matrix of an MDS code*. This means that a basis for the corresponding $n + k$-dimensional codeword space over a finite field K is given by the rows of the generator matrix $G = (M|I_n)$, where M is a $k \times n$ matrix and I_n is the identity $n \times n$ matrix.

Here we are chiefly interested in *square* MDS matrices, i.e. with $k = n$, however we must observe there are further uses in cryptography: the F-function of the block cipher PICARO (Subsection 3.38.1 on page 228), a Feistel network, uses a full generator matrix G of an MDS code to embed a eight-dimensional vector space over $\mathbb{F}_{2^8}$ into a 14-dimensional one, and then the transpose of G to compress back 14 dimensions to eight. In this case the generator matrix has a 6×8 redundancy part.

There are two ways of constructing MDS matrices: one can start with a known MDS code, for instance the code used in Shark is a Reed-Solomon code, or search for matrices that satisfy the non-singular sub-matrix condition.

Cauchy matrices are a classic example of MDS matrices. They are of the form $\left(\frac{1}{x_i - y_j}\right)_{0 \leqslant i,j < n}$ with all $x_i - y_j \neq 0$ over a field K. In general they do not lend themselves readily to optimisation. Amr Youssef, Serge Mister and Stafford Tavares define in [YMT97] a special class of Cauchy matrices for the design of diffusion layers in block ciphers: they construct their matrices A over a binary field K by first choosing the x_i's such that the least significant r bits of x_i are the binary representation of the number i, and then putting $y_i = x_i \oplus v$ where v is a non-zero field element such that its least significant r bits are all zero. This matrix satisfies $A^2 = cI_n$ where $c = \bigoplus_{i=0}^{n-1} \left(\frac{1}{x_1 \oplus y_i}\right)^2$ over K. The matrix A is then normalised dividing all its entries by $\sqrt{c}$, so that it becomes involutory. Such a $n \times n$ matrix also has only n different entries, which are

used for both encryption and decryption, reducing the number of circuits or short programs to implement for the multiplication by constants.

Vandermonde matrices (see [Yca13] for their history and naming) are matrices where each row is of the form $1, \alpha_i, \alpha_i^2, \ldots, \alpha_i^{n-1}$ for pairwise distinct α_i's. They are MDS matrices and there are very efficient algorithms for multiplication of vectors by them, as this operation amounts to multi-evaluation of a polynomial of degree $n-1$ at n points. These algorithms are DFT based (cf. Chapter 3 of [Pan01]) and therefore suitable only for large matrices. We are not sure which is the first mention of Vandermonde matrices for the construction of diffusion layers in SPNs: Often, in the literature, a 2004 paper by Jérôme Lacan and Jérôme Fimes [LF04] is cited, which however deals with a clever use of Vandermonde matrices to build erasure codes, not with cryptographic applications.

Hadamard matrices H have the property that $H \cdot H^t = nI_n$ (here H^t denotes the transpose of H). The first such matrices were originally constructed by James Joseph Sylvester [Syl67] and Jacques Hadamard [Had93] as real matrices with entries equal to ± 1, but over finite fields the latter condition is relaxed. The property $H \cdot H^t = nI_n$ makes them suitable to construct involutory diffusion layers, upon scaling, and they are used for this purpose in Anubis (Subsection 3.21.3 on page 190) and Khazad (Subsection 3.21.2 on page 188).

Mahdi Sajadieh et al. in [SDMO12], construct involutory MDS matrices using Vandermonde matrices over fields $\mathbb{F}_{2^r}$. Their idea is to take n pairwise distinct and non-vanishing values α_i for $0 \leqslant i < n$, a non-zero δ in $\mathbb{F}_{2^r}$, and to put $\beta_i = \alpha_i \oplus \delta$. If A and B are the Vandermonde matrices associated to the n-uples $(\alpha_0, \alpha_1, \ldots, \alpha_{n-1})$ and $(\beta_0, \beta_1, \ldots, \beta_{n-1})$, then $B \cdot A^{-1}$ is an involutory MDS matrix. They then go on to construct $2^d \times 2^d$ Hadamard involutory matrices recursively, starting from slightly modified 4×4 Vandermonde matrices: Kishan Chand Gupta and Indranil Ghosh Ray [GR13a] show that these matrices can be constructed also starting from Cauchy matrices.

1.8.2.3 Constructing Efficient MDS Matrices

We now focus on the problem of constructing *efficient* MDS matrices. We have already mentioned that choosing matrices with entries of low Hamming weight is often desirable, but the actual problem is the minimisation of the cost of the multiplication by the whole MDS matrix.

Multiplication of a variable vector or matrix by a fixed matrix with hard-wired constant entries has a code or area complexity often proportional to (or at least strongly correlated with) the number of entries different from one, as well as the values of such entries. Therefore, Pascal Junod and Serge Vaudenay introduce in [JV04b] the following criterion: *if $v_1(M)$ is the number of entries equal to one in the matrix M and $c_1(M)$ is the cardinality of the set $C(M)$ of distinct entries in M which are different from one, then the goal is to maximise $v_1(M)$ and to minimise $c(M)$.* Of course this does not take into account special situations which may make some matrices more efficient than others in some cases: for instance, multiplication the generator x of the polynomial basis of the field is inexpensive, as is also multiplication by x^{-1}; and the structure of the set $C(M)$ is not taken into account, as when some elements are the sum or product of other elements in the set. Junod and Vaudenay then start constructing candidates for MDS matrix from the concept of **bi-regularity**: *a 2×2 array with nonzero entries in a field K is **bi-regular** if at least one row and one column have two different entries.* It is clear that bi-regularity is a prerequisite for non-singularity. MDS matrices are constructed iteratively by extending bi-regular arrays, and lower bounds for

$v_1(M)$ and $c_1(M)$ are given as a function of the dimensions of the matrices.

To support their point through examples, Junod and Vaudenay consider the 4×4 MDS matrix M over $\mathbb{F}_{2^8} = \mathbb{F}_2[x]/(x^8+x^4+x^3+x+1)$ used in Rijndael – i.e. the matrix used in the `MixColumns` step described in Section 3.20 on page 180. It has $c_1(M) = 2$, which according to [JV04b] is optimal, but $v_1(M) = 8$, whereas a lower bound of $v_1(M) = 7$ is possible. Multiplication by the Rijndael matrix can be implemented using 15 XORs, four table lookups in one table (to implement multiplication) and using three temporary variables. Junod and Vaudenay show that the family of matrices of the form

$$\begin{pmatrix} a & 1 & 1 & 1 \\ 1 & a & 1 & b \\ 1 & b & a & 1 \\ 1 & 1 & b & a \end{pmatrix}$$

can be implemented using 10 XORs and seven table lookups in two tables, using two temporary variables. Using the sub-matrix non-singularity criterion, it is easily seen that such a matrix is a MDS matrix over a field extension of $\mathbb{F}_2$ if and only if 1, a, b, and $a+b$ are pairwise distinct from each other, $a \neq b^2$, and $a^2 \neq b$. This matrix is at the basis of the diffusion layer in IDEA NXT-64 (Section 3.23 on page 193). Junod and Vaudenay also construct a 8×8 matrix over $\mathbb{F}_{2^8}$, which is used in IDEA NXT-128. Being MDS, these matrices all have optimal branch numbers, i.e. 5 and 9 respectively.

A different line of research, followed by several authors during the last few years, and that is particularly advantageous for ciphers whose design criteria are compactness of code and data or of area, is to construct MDS matrices *iteratively*. The idea is simple, if a matrix N exists with a very compact and sparse representation such that the k^{th} power of N is a MDS matrix M, then one can just apply k times the matrix N in place of M. (For some reason, such constructions are often called *recursive* in the literature.)

Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew Robshaw design the diffusion layers of the hash function PHOTON [GPP11] and of the block cipher LED [GPPR12, GPPR11] by constructing their MDS matrix as the power of the companion matrix of a LFSR. Recall that if

$$y_{n+k} = c_{n-1}y_{n+k-1} + c_{n-2}y_{n+k-2} + \cdots + c_1 y_{k+1} + c_0 y_k \tag{1.3}$$

is a recursive relation with $c_0, c_{n-1} \neq 0$, then its characteristic polynomial is

$$g(X) = X^n - \left(c_{n-1}X^{n-1} + c_{n-2}X^{n-2} + \cdots + c_1 X + c_0\right) \tag{1.4}$$

and its companion matrix is the matrix C such that

$$\underbrace{\left(\begin{array}{c|ccc} 0 & & I_{n-1} & \\ & & & \\ \hline c_0 & c_1 & \cdots & c_{n-1} \end{array}\right)}_{C} \cdot \left(\begin{array}{c} y_k \\ \hline y_{k+1} \\ \vdots \\ y_{n+k-1} \end{array}\right) = \left(\begin{array}{c} y_{k+1} \\ \vdots \\ y_{n+k-1} \\ \hline y_{n+k} \end{array}\right), \tag{1.5}$$

which is denoted by $\mathsf{Serial}(c_0, c_1, \ldots, c_{n-1})$ in [GPP11]. The inverse of C has a simple form as well

$$\left(\begin{array}{c|ccc} 0 & & I_{n-1} & \\ \hline c_0 & c_1 & \cdots & c_{n-1} \end{array}\right)^{-1} = \left(\begin{array}{ccc|c} -\frac{c_1}{c_0} & \cdots & -\frac{c_{n-1}}{c_0} & \frac{1}{c_0} \\ \hline & I_{n-1} & & 0 \end{array}\right) \tag{1.6}$$

and its application is therefore also very efficient.

In the cipher LED, 16 bits of the state are interpreted as four nibbles and each nibble as an element of $\mathbb{F}_{2^4}$ via the definition polynomial $x^4 + x + 1$; if we identify the bits of a hexadecimal number with the corresponding polynomial representation of a field element, we have

$$A := \begin{pmatrix} 0_x & 1_x & 0_x & 0_x \\ 0_x & 0_x & 1_x & 0_x \\ 0_x & 0_x & 0_x & 1_x \\ 4_x & 1_x & 2_x & 2_x \end{pmatrix}^4 \quad \text{and} \quad M := A^4 = \begin{pmatrix} 4_x & 1_x & 2_x & 2_x \\ 8_x & 6_x & 5_x & 6_x \\ \mathtt{B}_x & \mathtt{E}_x & \mathtt{A}_x & 9_x \\ 2_x & 2_x & \mathtt{F}_x & \mathtt{B}_x \end{pmatrix},$$

where A is chosen so that M is MDS, hence it has optimal branch number 5. Hence, instead of using M directly in the diffusion layer, in LED the matrix A is applied four times. Since all entries have weight one, this can be implemented very efficiently. The actual entries of the last row of the companion matrix are found by exhaustive search so that M is MDS.

Mahdi Sajadieh, Mohammad Dakhilalian, Hamid Mala and Pouyan Sepehrdad in [SDMS12] construct their matrices by fixing $c_0 = 1$ in the companion matrix as given in Equation (1.5). By means of this the repeated application of the companion matrix can be represented as a unkeyed source-heavy GFN with n branches (similar to Figure 1.4 on page 31, Subfigure 9)

$$\begin{cases} y_0 = y_0 \oplus (c_{n-1}y_{n-1} \oplus c_{n-2}y_{n-2} \oplus \cdots \oplus c_1 y_{k+1}) \\ (y_0, y_1, \cdots, y_{n-1}) \leftarrow\!\shortmid (y_1, \cdots, y_{n-1}, y_0) \end{cases},$$

and thus the application of the inverse transform, being just the inversion of the Feistel network, is easy (see also Equation (1.6) with $c_0 = 1$). They also describe their transformations in terms of general linear transformations L, which are initially not fixed – instead the matrix M and the determinants of its sub-matrices are first computed symbolically. *Then* these linear transformations are specialised to integral algebraic elements over $\mathbb{F}_2$ and a binary field containing them is constructed.

Shengbao Wu, Mingsheng Wang and Wenling Wu in [WWW12] further reduce the search to companion matrices where the entries of the last row are all powers of the same linear transformation L, including negative powers. Thus the number of building blocks to be implemented is further reduced. By focusing on transformations that use a single XOR they achieve diffusion layers with an impressively small number of XOR gates.

The main problem with last two approaches is that they require to perform a large exhaustive search over very large spaces. For instance, to find a full diffusion layer for an AES-like cipher, the search space has cardinality 2^{128}.

Kishan Chand Gupta and Indranil Ghosh Ray consider in [GR13b, GR13c] 4×4 matrices computed as the fourth power of companion matrices such as $\mathsf{Serial}(1, \alpha, 1, \alpha^2)$, $\mathsf{Serial}(1, \alpha, 1, \alpha + 1)$ and $\mathsf{Serial}(1, \alpha, \alpha^2, \alpha + 1)$, where α is an element of the field extension, and characterise when

these are MDS. They also consider other forms of companion matrices and 5×5 examples.

Daniel Augot and Matthieu Finiasz in [AF13a, AF13b] improve upon the results of [WWW12, SDMS12] to produce 8×8 diffusion matrices over $\mathbb{F}_{2^4}$ and 16×16 matrices over $\mathbb{F}_{2^5}$. One of their key optimisations is that they fix the minimal polynomial that L must satisfy *in advance*. Still, the search is too slow for large ciphers.

The same authors propose in [AF14] a different approach, where the construction is direct instead of being based on an exhaustive search. The idea is that if we have an LFSR (1.3) with characteristic polynomial (1.4) we can write the companion matrix as

$$C := \left(\begin{array}{c|c} 0 & I_{n-1} \\ \hline X^n \bmod g(X) & \end{array}\right). \tag{1.7}$$

Now, C^n is MDS if and only if the matrix $G = (C^n | I_n)$ is the generator matrix of a MDS code, which means that we are looking for MDS codes generated by

$$G := \left(\begin{array}{c|c} \begin{array}{c} X^n \bmod g(X) \\ X^{n+1} \bmod g(X) \\ \vdots \\ X^{2n-1} \bmod g(X) \end{array} & I_{n-1} \end{array}\right).$$

Each line of the matrix (a codeword) is a multiple of $g(X)$, hence we have a cyclic code. In what follows, for terminology, definitions and proofs regarding cyclic codes we refer to [MS77].

Since in general, given a generator polynomial $g(X)$, computing the minimal distance of the associated cyclic code is a hard problem, Augot and Finiasz restrict their attention to a class of codes for this the problem admits an easy solution: BCH codes. *A **BHC code** over a finite field $\mathbb{F}_q$ is the cyclic code defined by a polynomial $g(X)$ that is the least common multiple of the minimal polynomials over $\mathbb{F}_q$ of the elements β^ℓ, $\beta^{\ell+1}$, ... $\beta^{\ell+d-2}$.* This $g(X)$ defines a cyclic code of length n equal to the order of β (i.e. n is the smallest positive inter such that $\beta^n = 1$) and minimal distance at least d. The dimension of the code is $n - \deg(g)$, which means that the code is MDS if $\deg(g) = d - 1$, and since $g(X)$ has $d - 1$ roots, and the $\beta^{\ell+i}$ for $i = 0, \ldots, d-2$ are among those, they form a complete orbit of algebraic conjugates.

The final observation is that in a diffusion layer the input and output size are equal, so we need a code of dimension, say, n and length $2n$ – i.e. β must have order $2n$, which is not possible in a binary field. Therefore a *longer* BCH code is built, and then shortened – in other words, β is chosen as an element of order $2n + z$ for some odd positive z, the corresponding BCH code is constructed, and then it is shortened by the last z positions. Shortening removes some words from the code, which means that it can not decrease the minimal distance, and if the code is MDS, shortening it preserves the MDS property. For the remaining (technical) details, we refer to the paper.

This method is efficient and permits the construction of large examples. For a diffusion layer of n bundles over $\mathbb{F}_q$ all possible BCH codes can be searched in time polynomial in q and n. However, not all MDS matrices cannot be obtained thusly – hence matrices which are optimal with respect to some additional and independent properties may not be found by this procedure.

Finally, the shortening length z can be very big. Shortening a code can increase its minimal distance: This is what happens with the matrix of the Photon hash function, which was reconstructed by Augot and Finiasz: the 4×4 matrix comes from a code of length $2^{24} - 1$, that has minimal distance 3. Once shortened to length 8, the minimal distance grows to 5.

1.8.3 Shuffles

The shuffle of the branches of a (generalised) Feistel network is usually just a simple circular rotation. On the other hand, in the SAFER cipher family (Section 3.8 on page 148) diffusion is achieved by alternating a layer of Pseudo-Hadamard Transforms with a bundle shuffle; the latter is carefully chosen to guarantee that after three such layers a change is diffused to all bundles (with some exceptions that we discussed earlier).

Taking a cue from Kaisa Nyberg's work on GFNs [Nyb96], that includes, for instance, the design depicted in Figure 1.4 on page 31, Subfigure 6 (b), Tomoyasu Suzaki and Kazuhiko Minematsu in [SM10] consider shuffles of the branches in GFNs, with the goal of minimising the number of rounds necessary to achieve full diffusion.

Consequently, they consider the setting where a single shuffle is performed per round – as opposed to the multi-shuffle design of SAFER. We see immediately that on one hand, shuffles are just bit permutations, but on the other hand they are compatible with the partitioning of the state in branches or bundles. This has two important consequences: all the output bits of a S-box (or of a Feistel target) influences the input to just one bundle or branch in the following round; this influence can be intuitively controlled better, since we do not have to take into account the influence of single unchanged output bits.

Let π be a shuffle of the branches of the Feistel network, where the branches are identified with the corresponding index set (for a k-branch GFN this set would be $[0..k-1]$). Define the quantity $\mathrm{DR}_\pi(j)$ as the minimum number of rounds necessary to diffuse the j-th sub input block of the first round, $x_0^{(j)}$ to all sub output blocks (of the $\mathrm{DR}_\pi(j)$-th round), and $\mathrm{DRmax}(\pi)$ to be the maximum of all such $\mathrm{DR}_\pi(j)$. For a k-branch Type 2 GFS it is $\pi(j) = (j+1) \bmod k$ and it is easy to see that $\mathrm{DRmax}(\pi) = k$. However, for a different shuffle π of the branches, the corresponding $\mathrm{DRmax}(\pi)$ can be different.

Now, let us first define

$$\mathrm{DRmax}^{\pm}(\pi) := \max\left\{\mathrm{DRmax}(\pi)\,,\, \mathrm{DRmax}(\pi^{-1})\right\}$$

(so that both encryption and decryption are taken into consideration) and

$$\mathrm{DRmax}_k^* := \min\left\{\mathrm{DRmax}^{\pm}(\pi) : \pi \in \Sigma([0..k-1])\right\}$$

where $\Sigma([0..k-1])$ is the full symmetric group over the set $[0..k-1]$. An exhaustive search gives $\mathrm{DRmax}_4^* = 4$, $\mathrm{DRmax}_6^* = 5$, $\mathrm{DRmax}_8^* = 6$, $\mathrm{DRmax}_{10}^* = 7$, and $\mathrm{DRmax}_k^* = 8$ for $k = 12, 14, 16$.

Suzaki and Minematsu then searched for optimal block shuffles, and for their optimal π_k^*

$$\mathrm{DRmax}(\pi_k^*) = \mathrm{DRmax}((\pi_k^*)^{-1})$$

holds true. Interestingly, for $k = 8$ a permutation π was found such that $\mathrm{DRmax}(\pi) = 5$ and $\mathrm{DRmax}(\pi^{-1}) = 7$, which is not optimal w.r.t. the above definition of DRmax_k^*. A cipher de-

signed around that permutation could have decryption easier to analyze than encryption! This is an uncommon occurrence, but, for instance, FROG (Subsection 3.19.3 on page 178) is such a cipher.

All optimum block shuffles π_k^* found by Suzaki and Minematsu also satisfy the property that any even-indexed input block is mapped to an odd-indexed output block and vice versa – so that the output of a target branch is permuted to the input of a source branch. Such shuffles are called **even-odd shuffles**.

A lower bound for DRmax* for even-odd shuffles can be derived as follows. For a fixed one block input difference, let N_i^o, resp. N_i^e, be the number of odd-numbered, resp. even-numbered, sub blocks in the i-th round output affected by that input block. Initially we have that one of N_0^e and N_0^o is 0 and the other one is 1. Assuming that the shuffle works ideally, we have $N_i^e = N_{i-1}^e + N_{i-1}^o$, and $N_i^o = N_{i-1}^e$ and from this we see that $N_i^e = N_{i-1}^e + N_{i-2}^e$ holds. Hence $\{N_i^e\}_i$ is a Fibonacci sequence. For a GFS with an even-odd shuffle, if a certain number of rounds is sufficient to achieve the diffusion to all even output blocks, the full diffusion is achieved by one more round. Therefore, if i is the smallest integer that satisfies $N_i^e \geqslant k/2$, $i+1$ is the lower bound for DRmax* for all even-odd shuffles for k blocks (not necessarily achievable). The sequence $\{N_i^e\}_i$ takes lower values with $N_0^e = 0$ and $N_0^o = 1$ and gives the Fibonacci numbers. Hence $N_i^e \approx \varphi^i/\sqrt{5}$, where φ is the golden ratio, and the lower bound for DRmax* is roughly $\log_\varphi \sqrt{5}k/2 \approx \log_2 1.44k$. The optimal results mentioned above for even k, $4 \leqslant k \leqslant 16$ are very close to this bound.

In [SM10] Suzaki and Minematsu show how to use colored de Bruijn graphs to build a block shuffle for $k = 2^{s+1}$ (for any $s \geqslant 2$) whose DRmax* is at most $2s + 2 = 2\log_2 k$. This is quite close to the $\log_2 1.44k$ lower bound just proved. For the details of the construction we refer to the paper. The important remark here is that this gives an upper bound that proves the logarithmic growth of DRmax$_k^*$.

The authors also compare their results to those of James Massey for the branch permutation used for diffusion in the SAFER family, in particular to the Armenian Shuffle used in SAFER+ (cf. Subsection 3.8.3 on page 150). Even though the Armenian Shuffle is also based on a de Bruijn graph, it is not an even-odd shuffle – bit this is not a problem for SAFER+, since it is a bricklayer cipher, not a generalised Feistel.

It is still an open question if better shuffle families can be found – i.e. with a smaller DRmax – or how it can be achieved by mixing different types of shuffles.

1.8.4 More on the Diffusion in Feistel Networks

Diffusion in Feistel Networks is a more complex topic than in the case of SPN ciphers. We have already seen in the previous subsection that a desirable property of branch shuffles in multi-branch GFNs is that they map odd branches to even branches and viceversa – property that the shuffles used in the SAFER family (Section 3.8 on page 148) need not satisfy.

However, in most cases the main reason for the added complexity is that in Feistel networks, diffusion happens at two different levels:

1. At the F-function level, changes in one input bit should be diffused to the output. In most cases this diffusion is only partial: not only these changes remain initially restricted to a

block whose size if a fraction of the cipher's state, but, even though some ciphers have a multi-round F-function (a notable example is LOKI97, described in Subsection 3.19.5 on page 178), this diffusion is not necessarily very strong from a cryptographic point of view, and it is made stronger by iteration in the *whole cipher*.

2. At the Feistel network level, changes in one branch of the state are "applied" to other branches in the following rounds, thus diffusing the changes propagated at the F-function level.

These two facts imply that diffusion is usually a slower process in Feistel networks than in SPNs, making a higher number of rounds a necessity. Hence, there have been a few attempts to improve diffusion in (generalised) Feistel networks. The first such attempt was made by Kaisa Nyberg in [Nyb96] – cf. Subfigures 6 (a) and 6 (b) of Figure 1.4 on page 31: It is a Type 2 Feistel network with a different permutation of the branches, and it was originally intended to be used with a simple round function that only consists of a single S-box, leaving diffusion to be performed by the more complex permutation. Good resistance against linear and differential cryptanalysis was proved.

After the work of Kaisa Nyberg, most efforts went into improving diffusion within the F-function or across the F-functions of consecutive rounds. Taizo Shirai and Kyoji Shibutani [SS04] study the use of MDS matrices in the F-function of a classic Feistel Network. Their observation if that the F-function is a SPN where the permutation layer is a linear transformation, the use of the same transformation will lead to the overlapping of similarly linearly transformed inputs in two successive rounds. The use of different matrices, alternated in a regular pattern, improves resistance against differential cryptanalysis and achieves a number of active S-boxes comparable to that of a pure SPN such as AES. They call this approach *Diffusion Switching Mechanism*. A theoretical explanation why this new design using three different matrices with maximal branch number achieves the better immunity is given by Taizo Shirai and Bart Preneel in [SP04]. Taizo Shirai and Kyoji Shibutani later prove in [SS06] that the conditions on the matrices and be relaxed somewhat. We observe that even though the Diffusion Switching Mechanism was conceived for classical two branch Feistel networks, it was lated adopted to more general Type 2 Feistel networks by the designers of CLEFIA (Section 3.28 on page 201).

Research focused then again on the optimization of the branch permutation with Tomoyasu Suzaki and Kazuhiko Minematsu's seminal paper [SM10]. Suzaki and Minematsu explicitly prove a lower bound for the number of rounds necessary for full diffusion of a single branch difference and show how to construct branch permutations (shuffles) that come very close to this bound. Their results will be discussed in detail in Subsection 1.8.3 on page 52, where they are applied to the design of the TWINE block cipher.

Kyoji Shibutani [Shi10] analyzes the combination of these two approaches (MDS matrices inside an SP type F-function and branch shuffles). Tight bounds for the number of active S-boxes after a few rounds are given, that depend only on the branch number of the matrices used if a branch permutation of the type studied in [SM10] is used. These results allow to significantly reduced the required number of rounds to be secure against differential and linear attacks.

Andrey Bogdanov and Kyoji Shibutani consider Feistel networks where the F-function consist of S-box layers interleaved with linear diffusion. The motivation for this research is that the instantiation of a Feistel network with an SP-type F-function is deployed in many cryptographic algorithms including E2 (cf. Section 3.18), Twofish (Section 3.13 on page 162), Camellia (Sec-

tion 3.18), CLEFIA (Section 3.28 on page 201), PICCOLO (Subsection 3.37.2 on page 226), and the hash function SHAvite-3 [BD10]. Several results have been published so far:

1. For 2 branch balanced Feistel ciphers, in [BS13b] they prove that SPS and SPSP F-functions are optimal in terms of the proportion of active S-boxes in all S-boxes – a common efficiency metric for substitution-permutation ciphers. Interestingly, one SP-layer in the F-function is not enough to attain optimality whereas taking more than two S-box layers does not increase the efficiency either.

2. In [BS11] three-branch Type 1 generalised Feistel networks (GFN) where the F-functions are two round SP networks are studied. The F-functions use MDS matrices. The authors show that, when applicable, three-branch GFNs can be more efficient in practice than those with four branches.

3. The line of research from [BS11] is then continued in [BS13a], where all variants of Type 1 and Type 2 four-branch GFNs with two-round SP F-functions are considered. The rationals for studying invertible two-round SP F-functions (we recall that the F-function in a Feistel Network needs not a priori to be invertible) is summarized by the authors as follows:

   (a) Due to the second S-box layer, double SP-functions allow, on the one hand, to limit the analysis to the differential and linear activity patterns of functions and, on the other hand, to effectively increase the number of active S-boxes.

   (b) The second diffusion layer of a double SP-function constrains the *differential effect* (i.e. there are many differential trails contributing to the same differential) which may affect SPS-functions.

   (c) Having an odd number of SP-layers does not enable to prove tight bounds on the number of active S-boxes by working with functions. It is also conjectured that functions with more than 2 SP-layers do not add to the efficiency of the construction. Hence double SP-functions provide an optimal efficiency.

   (d) The invertibility prevents a function from absorbing differences: If a nonzero difference enters a bijective function the output difference will also be nonzero.

   Advantages over single-round SP-type F-functions are proven explicitly. For all considered GFN topologies several types of cryptanalitic attacks and lower bounds for differential and linear probabilities are explicitly given.

Finally, in [Bog11] the less analyzed case of unbalanced Feistel networks is considered. In particular, three- and four -branch source-heavy Feistel networks are studied, where the contributions coming from the various rounds are XORed after the SP round.

1.9 Confusion

A crucial aspect in the design of a substitution-permutation network (including Feistel ciphers and other types) is the substitution layer – without it a cipher is in most cases simply a linear operation.

Some ciphers achieve the desired resistance while dispensing with S-boxes altogether. In IDEA (Section 3.6 on page 142) three operations on mutually "incompatible" algebraic structures are combined: multiplication in the multiplicative group of the integers modulo 257, integer addition modulo 256, and bitwise XOR. Any two of these three operations do not satisfy any distributive or associative law. This makes it very difficult to find linear approximations oF-functions obtained by chaining these operations and provides also good differential properties. Further example of such ciphers are given by the RC5 (Section 3.10 on page 156), RC6 (Section 3.15 on page 165), TEA and XTEA (Section 3.12 on page 159), HIGHT (Section 3.27 on page 200), Threefish (Section 3.30 on page 205), and SIMON and SPECK (Section 3.36 on page 220). A lot of work done to assess the strength of a cipher designed around different types of operations is ad-hoc. However, Serge Vaudenay's decorrelation theory, explained in Subsection 1.10.3 on page 63, offers some insights, especially when multiplications are used.

Sometimes mutually incompatible algebraic structures are used together with S-Boxes, to increase the non-linearity achieved by the latter. This has been done in GOST (Section 3.4 on page 138), in the the SAFER family (Section 3.8 on page 148), Blowfish (Section 3.9 on page 155), the cast CAST family (Section 3.7 on page 146), and many other ciphers.

In many cases, however, the non-linear components of a block cipher – or at least the most important ones – can be represented as **vectorial Boolean functions** that map n bits to m bits:

$$\begin{aligned} F : \mathbb{F}_2^n &\to \mathbb{F}_2^m \\ (x_0, \ldots, x_{n-1}) &\mapsto (y_0, \ldots, y_{m-1}) \ . \end{aligned}$$

Such a function is also called a ***(n,m)*-function**. It can also be viewed as a vector of m Boolean functions with the same n inputs, that can also be studied independently.

Regardless of how the actual function is implemented, we can consider it as a S-box.

A very thorough discussion of Boolean functions and of vectorial Boolean functions for cryptography can be found in Claude Carlet's chapters of [CH10], i.e. Chapter 8 [Car10a] and Chapter 9 [Car10b].

In the following sections we shall recall the principal design or selection criteria for S-boxes. To add some confusion (sic) to our treatment, some of these criteria actually deal with diffusion, such as the strict avalanche criterion, but, as we shall see, are ultimately related to confusion properties of the S-box as well.

1.9.1 Balancedness

A (n, m)-function F is called **balanced** if every value of $\mathbb{F}_2^m$ is taken by F the same number 2^{n-m} of times. In other words, the function is surjective and the distribution of the values is uniform. The balanced (n, n)-functions are the permutations on $\mathbb{F}_2^n$.

Balancedness is an important property for nonlinear components of a cipher in order to avoid statistical dependences between plaintext and ciphertext. Per se, balancedness is not a rare property: Any nonconstant affine function is balanced, but affine S-boxes are useless. On the other hand, it can be tricky to find a S-box that is balanced as well as satisfying other desirable cryptographic properties, as we shall see in the following subsections.

A (n, m)-function F is balanced if and only if its component functions are balanced, that is, if

and only if, for every nonzero $b \in \mathbb{F}_2^m$, the **component (Boolean) function**

$$F_b : x \mapsto \langle v, F(x) \rangle$$

is balanced (i.e. it takes the values 0 and 1 each 2^{n-1} times). A proof of this result can be found in [LN83] or [Car10b].

1.9.2 Algebraic Degree

The S-boxes should be algebraic functions of high degree in other to resist higher order differential attacks (Subsection 2.1.5 on page 80), which aim at eliminating low degree functions contributing to confusion in block cipher, as well as algebraic attacks.

We can write any Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ as a sum of monomials:

$$f(x) = \sum_{u \in \mathbb{F}_2^n} a_u \prod_{i=0}^{n-1} x_i^{u_i} \quad .$$

This is the **algebraic normal form** (short ANF, also known as Zhegalkin normal form, or Reed-Muller expansion) of the Boolean function f. The degree of the function f is the degree of the largest monomial in its ANF. It is a desirable property that for an S-Box s, any linear function of its output bits $x \to \langle s(x), a \rangle$ (with $0 \neq a \in \mathbb{F}_2^n$) has as largest degree as possible (or, at least, those linear combinations that are actually used in the cipher).

For large n, random Boolean functions have almost always algebraic degrees at least $n - 1$, In fact, number of Boolean functions of algebraic degrees at most $n-2$ equals $2^{\sum_{i=0}^{n-2} \binom{n}{i}} = 2^{2^n - n - 1}$ which is a fraction of $1/2^{n+1}$ of the set of all 2^{2^n} Boolean functions. However, functions of optimal algebraic degrees do not allow achieving some other characteristics, for instance we shall see that bent and almost bent functions (defined in Subsection 1.9.4 on the following page) have algebraic degree at most $n/2$ and $(n+1)/2$ respectively.

A high algebraic degree is important because lower degree functions will be more easily attacked by means of algebraic attacks (see Section 2.8 on page 120 and in particular Subsection 2.8.4 on page 122) or higher-order differential cryptanalysis (Subsection 2.1.5 on page 80): using lower degree functions can be offset by increasing the complexity of the cipher in other places, for instance by increasing the number of rounds – at the price of worse performance.

The algebraic degree is an affine invariant.

1.9.3 Algebraic Immunity

Considering just the degree of a Boolean function is not sufficient, since Boolean functions may have multiples of low degrees, and these can be used instead – these multiples can arise from the way the various bits are combined in the cipher, and thus their corresponding functions.

Hence the concept of **algebraic immunity** of a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$, defined by Willi Meier, Enes Pasalic and Claude Carlet [MPC04] as the minimum degree of all annihilators of f or $f + 1$:

$$\mathcal{AI}(f) := \min \left\{ \deg(g) \mid fg = 0 \vee (f+1)g = 0, g \neq 0 \right\} \quad .$$

In other words, only a function of degree at least $\mathcal{AI}(f)$ in the same inputs will "kill" the output

of f once a round constant or a fixed key bit is added to f. It is also known [CM03, MPC04] that any function of degree n must have an annihilator at the degree $\lceil n/2 \rceil$, whence $\mathcal{AI}(f) \leqslant \lceil \deg(f)/2 \rceil$. This important characteristic is an affine invariant.

There are several generalisations of the concept of algebraic immunity to (n, m)-functions, some of which are more useful for function where m is small with respect to n (a common situation in stream ciphers) and some which are better when m and n are comparable, as in block ciphers. For the latter case a natural generalisation is the **component algebraic immunity**, defined as the minimal algebraic immunity of the component functions F_b of the S-box F for $b \neq 0$.

1.9.4 Nonlinearity

A single biased linear approximation of a block cipher – possibly with a few initial or final rounds omitted – is often sufficient to mount a successful linear attack against it (see Section 2.2 on page 89). Therefore we need to ensure that such an approximation cannot possibly exist.

This can be achieved by choosing highly nonlinear S-boxes and then ensuring that the diffusion in the cipher forces all characteristics to cross a sufficiently high minimal number of "active" S-boxes. We consider here the non-linearity of the S-boxes.

In order to approximate a Boolean function $f := \mathbb{F}_2^n \to \mathbb{F}_2$ we can consider its **Walsh transform**

$$f^{w}(a) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+\langle a,x \rangle}$$

where $f^{w}(a)$ is called the **Walsh coefficient** of f at the place a. A Boolean function f is balanced if and only if $f^{w}(a) = 0$. The Walsh transform of f is the Fourier transform of the sign function f_χ of f, defined as $f_\chi(x) = 1$ if $f(x) = 0$ and $f_\chi(x) = -1$ if $f(x) = 1$, or, equivalently, $f_\chi(x) = (-1)^{f(x)}$. Also, note that if

$$\mathscr{L}_a(f) = \left\{ x \in \mathbb{F}_2^n \mid \langle a, x \rangle = f(x) \right\}$$

is the set of inputs where f and the linear function $x \mapsto \langle a, x \rangle$ agree, we have

$$f^{w}(a) = 2\#\mathscr{L}_a(f) - 2^n \ .$$

If the Walsh coefficient of f at place a is positive, resp. negative, it is clear that the larger it is the better the function f will be approximated by the linear function $x \mapsto \langle a, x \rangle$, resp. the affine function $x \mapsto 1 + \langle a, x \rangle$. So we want Walsh coefficients that are close to zero, and such that the largest of them in absolute value is as small as possible. This leads to the definition of **nonlinearity** of a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$:

$$nl(f) := 2^{n-1} - \frac{1}{2} \max_{a \in \mathbb{F}_2^n} \left| f^{w}(a) \right| \ .$$

It is clear that we aim at finding functions with the highest nonlinearity possible: the reason being that $nl(f)/2^{n-1}$ is an upper bound for the likelihood that an affine relation holds between some input bits and the output bit. Now, Parseval's Theorem states that $\sum_{a \in \mathbb{F}_2^n} f^{w}(a)^2 = 2^{2n}$, i.e. average of the squares of the Walsh coefficients is 2^n. From this it follows that the "best" nonlinear functions have Walsh coefficient equal to $f^{w}(a) = \pm 2^{n/2}$ at all places a, that $\max_{a \in \mathbb{F}_2^n} \left| f^{w}(a) \right| \geq$

$2^{n/2}$ and, finally,

$$nl(f) \geq 2^{n-1} - 2^{n/2-1} \quad . \tag{1.8}$$

This bound is tight, and is also called the **covering radius bound**, since this is the value of the covering radius of the Reed-Muller code of order 1 for n even. The functions that achieve bound (1.8) are called **bent functions** because they are as different as possible (in the sense of Hamming weight of the difference) – and in fact equidistant – from all linear and affine functions. They have been investigated in the '60s by Oscar Rothaus in research that was not published until 1976 [Rot76]. Clearly, bent functions exist only for n even.

The concept of Walsh coefficient is easily generalised to (n, m)-functions F by simply replacing the single-bit valued f in the original definition with the component functions of F:

$$F_b^w(a) = \sum_{x \in \mathbb{F}_2^4} (-1)^{\langle b, F(x) \rangle + \langle a, x \rangle} \quad . \tag{1.9}$$

Then, we define the **nonlinearity** of a (n, m)-function F as the minimum of the nonlinearities of all its component functions, i.e.:

$$nl(F) := 2^{n-1} - \frac{1}{2} \max_{\substack{a \in \mathbb{F}_2^n \\ b \in \mathbb{F}_2^n, b \neq 0}} \left| F_b^w(a) \right| \quad .$$

As for Boolean functions, $nl(F)/2^{n-1}$ gives an upper bound for the likelihood that an affine relation between some of the input bits and at least one of the output bits holds.

Vladimir Sidelnikov [Sid71] and, independently, Florent Chabaud and Serge Vaudenay [CV94], proved following bound on the nonlinearity of a (n, m)-function with $n \geqslant m - 1$:

$$nl(F) \leqslant 2^{n-1} - \frac{1}{2} \sqrt{3 \times 2^n - 2 - 2 \frac{(2^n - 1)(2^{n-1} - 1)}{2^m - 1}} \quad . \tag{1.10}$$

This is called the **Sidelnikov-Chabaud-Vaudenay bound** in [Car10b], and it can be tight only if $n = m$ with n odd.

Similarly to the definition of bent Boolean function, we say that a vectorial Boolean function F is bent if (1.8) holds with equality. It is easy to see that the notion of bent vectorial function is invariant by addition of affine functions and under composition on the left and on the right by affine automorphisms. By definition, a (n, m)-function is bent if and only if all of the nonzero component functions are bent. Bent (n, m)-functions exist only if n is even and $m \leqslant n/2$ – in particular there are no bent permutations of $\mathbb{F}_2^n$, as proved by Kaisa Nyberg in [Nyb91]. Also, the algebraic degree of a bent function with $n \geqslant 4$ is at most $n/2$, as proved in [Rot76]. More refined bounds on the degree of bent functions have been proved by Xiang-Dong Hou in [Hou00] (the result is also given as Proposition 19 in [Car10a]).

Nyberg suggested two constructions of bent S-boxes, one based on the Maiorana-McFarland construction and one based on Dillon's construction of difference sets. For more details and references see [Nyb91]. Further references and constructions can be found in [Car10a, Car10b].

Also, already in 1988 Réjane Forré observed that the Walsh transform of a function could be used to verify whether a vectorial Boolean function satisfies the strict avalanche criterion (cf.

Subsection 1.9.6 on the next page). It turns out that the functions satisfying the SAC to the highest possible order (i.e. functions that are SAC even if an arbitrary number of input bits are fixed) are bent [AT90b] – in other words they are ideal candidates to achieve good diffusion.

Ideal diffusion properties, resistance to linear cryptanalysis as well as to differential cryptanalysis (as we shall see in the next Subsection) may lead us to the conclusion that bent functions are the perfect choice to construct secure cryptographic primitives. However, they are not balanced, which means that they cannot be used directly to construct invertible S-boxes, and they also "funnel" the input to a half-size output at best, which also complicates the constructions. Therefore a more common approach is to start with bent functions and modify them by augmenting the output in order to obtain balanced functions that still attain high nonlinearity [MS89].

An important relaxing of the bentness condition is given by **almost bent (AB)** functions. A (n, n)-function F is almost bent if they achieve the bound (1.10) with equality, i.e. $F_b^w(a) \in \{0, \pm 2^{(n+1)/2}\}$ at all places a. Almost bent functions exist only for n odd. They have degree at most $(n + 1)/2$ [CCZ98]. The name "almost bent" may be confusing because it may lead to think that they are not optimal, but they are – bent (n, n)-functions do not exist.

Natalia Tokareva in [Tok11] gives a survey of generalisations of bent functions.

1.9.5 Differential Uniformity

To make differential cryptanalysis difficult (Section 2.1 on page 71), for each input difference Δ, the set of output differences $S(x) - S(x + \Delta)$ of an S-box S should have a as uniform distribution as possible, so that even when some output differences occur more often for given input differences, they do not stand out in a particular way. This is measured by **(differential) uniformity**: For a (n, m)-function F we define the differetial of F at the point c

$$\Delta_c F(x) := F(x \oplus c) \oplus F(x)$$

(a more general definition for functions over rings is $\Delta_c F(x) := F(x + c) - F(x)$) and the value

$$\delta_F := \max_{\substack{c \in \mathbb{F}_2^m . c \neq 0 \\ a \in \mathbb{F}_2^m}} |\Delta_{F,c}^{-1}(a)|$$

is called the **(differential) uniformity** of F.

Since $\Delta_c F(x)$ is a (n, m)-function, we trivially have the tight bound $\delta \geqslant 2^{n-m}$. If this bound is attained, the function F is called **perfect nonlinear (PN)** – this means that each $\Delta_c F(x)$ with $c \neq 0$ is balanced, and conversely if each $\Delta_c F(x)$ with $c \neq 0$ is balanced then F is PN. It was proven by Willi Meier and Othmar Staffelbach in [MS89] that a function is PN if and only if it is bent. This result was generalised to fields of arbitrary characteristic by Kaisa Nyberg in [Nyb90]. This is a strong link between resistance to linear and differential cryptanalysis, and also implies that PN (n, m)-functions only exist if $m \leqslant n/2$.

Let us now consider the cryptographically important case $n = m$: it is clear that all $|(\Delta_c F)^{-1}(a)|$ are even (so they cannot be $2^{n-n} = 1$) and not all can be zero, so $\delta_F \geqslant 2$. Functions with $m = n$ attaining this lower bound are called **APN (Almost-Perfect Nonlinear)** functions.

Since, for modern cipher design the S-box usually is bijective, of particular interest are bijective APN functions, called **APN permutations**. If a function is APN and bijective, then the inverse

is also APN.

APN permutations exist, and they are plenty in odd dimension – if we identify $\mathbb{F}_2^n$ for odd n with the Galois field $\mathbb{F}_{2^n}$ we can use either cubing or inversion (other exponents can be used as well). As a result, for instance, $S(x) = x^3$ in any odd binary field is immune to differential and linear cryptanalysis. This is in part why the MISTY designs (see Subsection 3.18.8 on page 174) use 7- and 9-bit functions in the 16-bit non-linear function. (What these functions gain in immunity to low order differential and linear attacks they lose to higher order differential cryptanalysis and algebraic attacks, i.e. they can be described and solved via a SAT solver.)

The search for APN permutations in even dimensions, which are highly desirable, is more difficult. Until recently it was not known whether they existed at all. At the Fq9 conference in 2009 John Dillon announced an APN permutation on $\mathbb{F}_{2^6}$ [BDMW10]. It is not known whether there are other examples. Hence, in general, the best one can realistically hope to find for a (n, n)-function F with n even is $\delta_F = 4$ – there are plenty of functions with this property, for instance field inversion in $\mathbb{F}_{2^n}$. The fact that field inversion in $\mathbb{F}_{2^n}$ has $\delta_F = 2$ for n odd and $\delta_F = 4$ for $n = 2$ is proved by Kaisa Nyberg in [Nyb93], where she attributes the observation to Lars Knudsen. This result will later influence the choice of the AES S-box cf. Section 3.20 on page 180.

In the same paper, Nyberg studies other power functions as well. She also consider mappings derived from exponential functions in prime fields (such as those used in the SAFER family – cf. Section 3.8 on page 148): she proves (Proposition 7 in [Nyb93]) that a mapping from the of integers modulo a prime p defined as exponentiation of element of order $p - 1$ in $\mathbb{F}_p$ is differentially 2-uniform with respect to addition modulo p (the binary differential uniformity is usually different). The type of differential uniformity determined also the principal type of operation in the cipher, and this explains why the main operation in SAFER is the modular addition.

There are some important relations between almost bent functions and APN functions:

1. AB functions are APN. To formulate a more precise result, let us first define plateaued functions: A (n, m)-function is called plateaued if, for every nonzero $c \in \mathbb{F}_2^m$, the component function F_c is plateaued, that is, there exists a positive integer ν_c (called the amplitude of the plateaued Boolean function) such that the Walsh spectrum of F_c is $\{0, \pm\nu_c\}$.

   Now, *a (n, n) function F is AB if and only if it is APN and all its non-zero component functions are plateaued with the same amplitude.* A proof of this result can be found in [Car10b].

2. A quadratic (n, n)-function F with n odd is almost bent.

Further requirements are discussed in Subsection 2.1.10 on page 88.

1.9.6 Strict Avalanche Criterion and Propagation Criterion

We already defined the strict avalanche criterion [WT85]. For a S-box it is simply formatted as: *Complementing a single bit in the input to an S-box should change a bit in the output with probability* 1/2, *for any input bit and for any output bit.*

A more general definition is the following: *A Boolean function on n variables satisfies the SAC of order k, $0 \leqslant k \leqslant n - 2$, if whenever k arbitrary input bits are fixed, the resulting function of $n - k$ variables satisfies the SAC.*

The SAC and the higher order SAC are strong criteria that guarantee that every input bit change diffuses to all output bits with equal likelihood, and thus contributes in a significant way to diffusion.

Many generalisations of the SAC exist, such as the **propagation criterion (PC)** [Pre93, PLL+90, PGV91] due to Bart Preneel et al. A (n, m)-function F satisfies the propagation criterion with respect to the set $E \subseteq \mathbb{F}_2^n$ if, for all $c \in E$, the differential $\Delta_c F(x)$ is balanced. The set E is usually taken to be the set of non-zero vectors of Hamming weight up to ℓ, and the criterion is then written as PC(ℓ). Also, if k inputs are kept fixed, we have the PC or the PC(ℓ) of order k. In general, these criteria are not affine invariant. The propagation criterion can be viewed as a weaker form of bentness or of perfect nonlinearity, and, indeed, bent functions satisfy the propagation criterion with respect to the set of non-zero vectors.

1.9.7 Other Criteria

Countless criteria had been suggested in the literature to measure how a S-box contributes to confusion and diffusion. We briefly summarise here just two more.

1. **Non-existence of Nonzero Linear Structure** [Eve87]: Nonlinear cryptographic functions used in block ciphers should have no nonzero linear structure, i.e. vectors c such that $\Delta_c F(x)$ is constant. If such a vector c existed, this could give rise to differential characteristics of high likelihood.

2. In [MS89] Willi Meier and Othmar Staffelbach introduced an interesting generalisation of nonlinearity. The **distance to linear structures** of a Boolean function is defined as its distance to the set of all Boolean functions admitting nonzero linear structures. The latter include all affine functions, and therefore the distance to linear structures this distance is bounded from above by the nonlinearity, but also other functions, for instance the non bent quadratic functions. The distance of a Boolean function in n variables to linear structures equals 2^{n-2} if and only if it is bent, as proved in [Car10a, Section 4.1.5].

1.10 Less Beaten Paths

1.10.1 Taking Inspiration From Stream Ciphers

In 1999 [GG99] Guang Gong and Solomon Golomb observed that many block ciphers can be viewed as a Non Linear Feedback Shift Register (NLFSR) with input. This includes most SPN and Feistel Network designs.

From their analysis they concluded that the S-box should not only not be approximated by a linear function, but it should also not be approximated by a monomial. One can thus ask whether one should turn their remark into a design concept, and explicitly use NLFSRs to design block ciphers: the plaintext is used to initialize the state, the key provides the input, and the state after sufficiently many rounds is the ciphertext. The key observation here is that the changes per round may be minimal, but they can be expressed as nonlinear, non-monomial transforms of low degree, and a round can be executed extremely quickly – therefore one can pile up enough rounds to get the desired confusion and diffusion and at the same time guarantee that the cipher can only be approximated by a polynomial of prohibitively high degree.

This approach was in fact realized already in the mid 80's in the design of KeeLoq (Section 3.3.3 on page 136). KATAN (Section 3.31 on page 207) is another, more recent, example of such a cipher.

Other ciphers take inspiration from stream ciphers for the *key schedule*. For instance, SIMON and SPECK (Section 3.36 on page 220), use LFSRs to generate the round keys.

1.10.2 Hybrid Designs

Because of the countless variations in both Feistel, SPNs and Lai-Massey designs, unavoidably there are ciphers which resemble more then one design, or even combine them.

TWINE (Section 3.34 on page 213) blurs the line between the classic Feistel and SP networks: It is a Feistel-like design with a total of 16 branches, and the branches are shuffled by a complex permutation based on colored de Bruijn graphs in place of a simple cyclic rotation. Therefore we have a "partial" substitution layer alternated with a bit-permutation layer that operated on 16 nibbles. We discussed shuffle design in Subsection 1.8.3 on page 52.

ZORRO (cf. Subsection 3.38.2 on page 229) is an AES-like cipher that represents its state as 16 nibbles, so it is a SPN. But, at each round the S-boxes are applied only to four of the 16 nibbles. This means that the cipher can be viewed represented as Matsui-like Feistel Network with linear mixing layers in place of the branch permutations.

SC2000 (cf. Section 3.22 on page 192) is unique in that it mixes SPN-like and Feistel-type rounds in the data obfuscation path: it is the only such cipher we are aware of.

The Bielorussian standard block cipher BEL-T (Section 3.33 on page 212) combines Feistel rounds with Lai-Massey rounds in a single round.

1.10.3 Decorrelation Theory

Serge Vaudenay presented in [Vau98b] and further analysed in [Vau03] methods to harden a cipher against linear and differential attacks. To achieve this he introduced the concept of decorrelation, to measure the "distance" of a cipher from a perfect cipher with respect to linear and differential properties.

Decorrelation is defined as follows:

- *Given a function F from a given set $\mathcal{M}$ to a given set $\mathcal{N}$ and an integer d, the* ***d-wise distribution matrix*** *$[F]^d$ of F is defined as the $\mathcal{M}^d \times \mathcal{N}^d$-matrix where the (x, y)-entry of $[F]^d$ corresponding to the multipoints $\mathbf{x} = (x_1, \ldots, x_d) \in \mathcal{M}^d$ and $\mathbf{y} = (y_1, \ldots, y_d) \in \mathcal{N}^d$ is the probability that we have $F(x_i) = y_i$ for $i = 1, \ldots, d$.*
- *Given two functions F and G from a given set $\mathcal{M}$ to a given set $\mathcal{N}$, an integer d and a multiplicative distance L over the vector space $\mathbb{R}^{\mathcal{M}^d \times \mathcal{N}^d}$ defined by a matrix norm, we call $L([F]^d, [G]^d)$ the d-wise* ***L-decorrelation*** *between F and G.*

The goal of the designers of block ciphers is to minimise the decorrelation between their functions and an ideal cipher with respect to suitable metrics. If we define $L([F]^d)$ as $L([F]^d, [C^*]^d)$ where C^* is an ideal cipher, from the multiplicative property of matrix norms it holds that

$$L([F \circ G]^d) \leqslant L([F]^d) \cdot L([G]^d) \ .$$

This enables the designers to build ciphers with bounded low decorrelation as block ciphers. Taking $d = 1$ allows to bound resistance against linear cryptanalysis and $d = 2$ against linear and (first order) differential cryptanalysis. The number d is called the **order** of the attack, hence in Vaudenay's terminology [Vau99c] linear cryptanalysis is an iterated attack of order one and differential cryptanalysis is an iterated attack of order two.

Vaudenay considers various norms, such as the L_2 norm, the infinity weighted pseudo-norm N_∞, the L_∞-associated matrix norm $|||\cdot|||_\infty$, and a new norm $\|\cdot\|_a$ defined in [Vau99a] as

$$\|M\|_a = \max_{x_1} \sum_{y_1} \max_{x_2} \sum_{y_2} \cdots \max_{x_d} \sum_{y_d} |M_{\mathbf{x},\mathbf{y}}|$$

which is designed to model adaptive attacks. These norms serve to easily compute bounds on the success probabilities of guessing the output of a cipher, which can then be readily translated into attack complexities.

An important result of Vaudenay's regards Feistel ciphers. Theorem 9 of [Vau98b] states:
Let $F_1, \ldots, F_r$, R be r independent functions on $\mathscr{M}$ where R has a uniform distribution and such that $\left|\left|\left|[F_i]^d - [R]^d\right|\right|\right|_\infty \leqslant \epsilon$ for $i = 1, \ldots, r$. Let $\Psi(F_1, \ldots, F_r)$ denote the Feistel cipher with $F_1, \ldots, F_r$ as F-functions and C^* the corresponding perfect cipher. For any $k \geqslant 3$ we have:

$$\left|\left|\left|[\Psi(F_1, \ldots, F_r)]^d - [C^*]^d\right|\right|\right|_\infty \leqslant \left((1+\epsilon)^k - 1 + \frac{2d^2}{\sqrt{\#\mathscr{M}}}\right) .$$

This makes the $|||\cdot|||_\infty$-decorrelation a useful tool for constructing Feistel ciphers. (Similar results are proved in [Vau03] for Lai-Massey ciphers with orthomorphisms; this type of ciphers is defined in Section 1.5 on page 36).

Decorrelation is achieved using *decorrelation modules*, i.e. simple functions for which decorrelation cane easily computed, which are then composed to construct a product cipher. An important class of decorrelation modules has the form

$$F(x) = k_1 + k_2 \cdot x \tag{1.11}$$

over a finite field $\mathbb{F}$, where k_1 and k_2 are secret keys taken uniformly from $\mathbb{F}$ and $\mathbb{F}^*$. In Vaudenay's terminology, this is a Type II NUT (n-Universal Transformation), and this function offers perfect decorrelation, i.e. it has the same decorrelation as a perfect cipher over $\mathbb{F}$.
If on the other hand the function defined in (1.11) is considered modulo p where $p = (1-\delta)2^m$ with $0 < \delta < 1/14$, but k_1, k_2 are independent uniformly distributed random variables in $\mathscr{M} := [0..2^m - 1]$, and F^* is a uniformly distributed random function from $\mathscr{M}$ to $\mathscr{M}$, then $\|[F]^2 - [F^*]^2\|_2 \leqslant \sqrt{8\delta}$. Thus, this type of decorrelation modules can be used to provide resistance against differential cryptanalysis.

A total of six different types of NUTs are discussed in [Vau03]. An important class is the Type IV NUT, of the form

$$F(x) = (k_1 + k_2 x + k_3 x^2 + \cdots + k_d x^{d-1} \bmod p) \bmod 2^m$$

where p is a prime that this time is just slightly *larger* than 2^m. These NUTs are designed to provide resistance against adaptive adversaries evaluated by the $\|\cdot\|_a$ norm. The bound on the

decorrelation is in this case given by $\|[F]^d - [F^*]^d\|_a \leqslant 2((1+\delta)^d - 1)$.

The DFC cipher (Subsection 3.19.2 on page 177) uses a Type IV linear NUT (i.e. with $d = 2$) that mixes arithmetic modulo $2^{64} + 13$ and 2^{64}.

Because of the form of many of Vaudenay's decorrelation modules, his approach can thus be viewed as a formalisation, generalisation and quantification of the use of different, and mutually algebraically incompatible arithmetic operations for the purpose of achieving non-linearity (Section 1.9 on page 55).

Besides DFC, ciphers constructed according to this principle include the COCONUT, PEANUT and WALNUT families (presented in [Vau98b]), and DONUT [CLLL00]. The 64-bit block cipher COCONUT consists of two small Feistel ciphers with a state-wide decorrelation module (a Type II NUT over $\mathbb{F}_{2^{64}}$) in the middle - the drawback being that decryption requires a field inversion.

Most of the decorrelation theory is developed under the assumption that different functions used in a single cipher are assumed to be independent. In practice, however, these are instances of just a few types of keyed functions with different round keys. Indeed, in Vaudenay's own words [Vau03], *One problem with the COCONUT, PEANUT, or WALNUT constructions is that they require a long key (in order to make the internal random functions independent). In real-life examples we can generate this long key by using a pseudorandom generator fed with a short key, but the results on the security based on decorrelation are no longer valid. However, provided that the pseudorandom generator produces outputs which are indistinguishable from truly random sequences, we can still prove the security.* This puts considerable weight on proper key schedule while in the design of using decorrelation theory, and makes the theory difficult to use in the context of lightweight block cipher design in most circumstances.

Serge Vaudenay has made a wealth of information about DFC and the theory of decorrelation modules available online [Vau00, Vau02].

In 2006 two papers coauthored by Thomas Baignères and Matthieu Finiasz have been published that present an interesting application of decorrelation theory. These deal with the block ciphers "C" [BF06a] and KFC, the "Krazy Feistel Cipher" [BF06b].

The cipher C follows the same SPN as the AES (Section 3.20 on page 180), i.e. the wide trails design represented in Figure 1.5 on page 35, but with following differences: there is no round key addition, and the substitution layer is formed by 16 independent *perfectly random permutations* instead of 16 copies of a fixed substitution box. Here, perfectly random permutations refers that the permutations are uniformly chosen among all permutations of a given set, and *independent* refers to the fact that a (pseudo) random number generator is used to select them. Since an arbitrary permutation of the set $S = [0, .., 255]$ can be described by $\lceil \log_2(2^8!) \rceil = 1,684$ bits, an algorithm is provided to turn 1,684 bits into a permutation of S, and the key schedule expands the secret key to $160 \times 1,684 = 269,440$ to describe a total of 160 random permutations on S. The key expansion uses the Blum-Blum-Shub (BBS) PRNG, i.e. the PRNG proposed in 1986 by Lenore Blum, Manuel Blum and Michael Shub in [BBS86], that consists of repeated squaring of a seed modulo a RSA modulo.

Resistance to linear and differential cryptanalysis, including impossible differential cryptanalysis is proved - but the key schedule makes the algorithm impractical in many contexts.

In order to optimise this construction, a smaller set $\mathscr{D}$ of mutually decorrelated S-boxes is used in [BF06a]. Some options are given by the following families of permutations that guarantee

protection against order one and two attacks, i.e. against linear and differential cryptanalysis:

- The set $\mathscr{D} = \{X \mapsto A \oplus B \cdot S(X) \mid A, B \in \mathbb{F}_{2^8}, B \neq 0\}$ defined by Kazumaro Aoki and Serge Vaudenay in [AV03], where S is any fixed permutation of $\mathbb{F}_{2^8}$ and "$\cdot$" is field multiplication.
- The set $\mathscr{D} = \{X \mapsto A \oplus B \cdot X^{-1} \mid A, B \in \mathbb{F}_{2^8}, B \neq 0\}$, defined in [BF06a].

Each element in the sets can be defined by 16 output bits of the BBS PRNG, where the values corresponding to $B = 0$ can be skipped, and the next value is used, so on average 2,570 output bits are needed in place of 269,440. It is proved that the permutation family reduced versions of the cipher are no less secure than the general version. Another optimisation option is to use a different PRNG, for instance a fast stream cipher, to speed up the key schedule (however, this in general loses some of the assumptions upon which the proofs of security of C rely).

The analysis framework is the one developed in [BV05] by Thomas Baignères and Serge Vaudenay to analyse the intrinsic security of the AES SPN.

One issue with C is that it requires random permutations, and using random functions would make the whole key scheduling process faster. This is solved in the design of KFC. KFC is a three round Feistel network, where the F-function is a SPN constructed similarly to C, but using random functions in place of random permutations – the resulting function is no longer injective, but this is not a problem in a Luby–Rackoff cipher. In order to make it difficult for an attacker to successfully exploit collisions, the first and last substitution layers are still constructed from random permutations.

Pierre-Alain Fouque and Pierre Karpman in [FK13a, FK13b] strengthen the FX construction (Section 1.6 on page 38) against MITM attacks (Section 2.4 on page 94) by using families of decorrelation modules (parametrised by keys) in place of simple key whitening.

1.11 Relations to Other Symmetric Constructions

One of the many aspects of block ciphers that make them very interesting is that they are closely related all other important symmetric primitives, such as stream ciphers and hash functions. In this section we recall these relations and mention some important theoretical constructions.

1.11.1 Block Ciphers and Stream Ciphers

The most obvious relation between block ciphers and stream ciphers is the use of a special mode of operation to turn an instance of the former into an instance of the latter.

The output feedback (OFB) mode of operation [Dwo01] turns a block cipher into a synchronous stream cipher by repeatedly encrypting an initialisation vector IV and XORing the results to the plaintext blocks $P_0, P_1, \ldots$:

$$C_i = P_i \oplus I_i \quad \text{where} \quad I_i = E_K(I_{i-1}) \quad \text{for} \quad i = 0, 1, 2, \ldots \quad \text{and} \quad I_{-1} = \text{IV} \ .$$

The same sequence of operations, with P_i and C_i swapped, is used to decrypt the ciphertext. Other modes of operation, such as cipher feedback mode (CFB), turn a block cipher into a self-synchronizing stream cipher. Counter mode (CTR) encrypts a text formed by concatenating a nonce and a increasing counter to obtain successive keystream blocks, which are then XORed

to the plaintext blocks to obtain the ciphertext blocks. More modes exist that turn a block cipher into a stream cipher, but we are not interested in giving a full treatment of such modes of operation.

The opposite transformation is possible as well. A computationally expensive construction was presented by Louis Granboulan and Thomas Pornin at FSE 2007 [GP07], which is mostly of theoretic interest. The idea consists in getting sufficient random data to completely determine a *Durstenfeld shuffle* [Dur64][2]. The paper uses a PRNG but a stream cipher can naturally be used instead for such a purpose. Using $O(\log b)$ space and $O((\log b)^3)$ PRNG invocations, the construction produces a random permutation over a set of m elements, uniformly selected among the $b!$ possibilities, where b is the block size. The key size is the size of the PRNG seed.

Some constructions explained later can also be used, such as BEAR, LION, LIONESS, and AARDVARK. These require, beside a stream cipher, also a hash function. The latter can be built from a stream cipher by using, for instance, the Toeplitz matrix construction due to Hugo Krawczyk [Kra94], using the stream cipher in place of the LFSR.

1.11.2 Block Ciphers and Hash Functions

Besides turning a block cipher into a stream cipher and then using the latter to construct a hash function, there are several methods that can be used to build a cryptographic hash function from a block cipher.

These usually take a *one-way compression function*, (i.e. a function that takes two fixed length inputs and produces a fixed length output of the same size as one of the inputs – and that is one-way [Gol98, §1.2.1]). A one-way compression function resembles a block cipher, and in fact it can often be easily constructed from a block cipher, usually by XORing one or both of its inputs to the output of the block cipher, in order to break its bijectivity. The one-way compression function is then used in an algorithm that resembles the common block cipher modes of operation used to encrypt arbitrarily long plaintexts: the compression function is called repeatedly with the output of the previous iteration as one of its inputs and the current message block as its second input. The message is suitably padded, then length of the message is usually appended to the padded message itself, and the output of the final application of the compression function is the tag.

There are several constructions that turn a block cipher into a compression function, for instance the Matyas–Meyer–Oseas [MMO85], the so-called Davies–Meyer scheme[3], Miyaguchi–Preneel [ISO91, MIO89, PGV93], and Hirose [Hir06]. These and many other schemes are analysed in [PGV93]: in fact a total of 64 different schemes are compared there, of which 12 are found to be secure.

Many well-known hash functions, including MD4 [Riv90], MD5 (see RFC 1321), SHA-1 and SHA-2 [NIS12], and WHIRLPOOL [BR11a] are built in this way starting from custom designed block-cipher-like components. The reason for the use of custom ciphers is that some features,

[2]This algorithm was popularised by Donald Knuth [Knu97, § 3.4.2, Algorithm P (Shuffling)] and therefore it is often known as *Knuth shuffle*. Knuth attributes it to Ronald Aylmer Fisher and Frank Yates [FY38] and claims that Richard Durstenfeld republished it. Derek O'Connor [O'C11b, O'C11a] argues that Durstenfeld's algorithm is not the same as the *Fisher–Yates shuffle*. Also, note that later editions of [FY38] replace the original algorithm with a different shuffling algorithm they attribute to Calyampudi Radhakrishna Rao.

[3]This scheme is attributed to D. Davies in [Win83, Win84], but Davies himself attributed it to C. Meyer in [DP84] and confirmed in a personal communication to the authors of [PGV93] that he did not develop the construction.

such as very large keys and blocks, and key agility, are not met by standard block ciphers such as the AES. Furthermore, in many hash function constructions it is important that the underlying block cipher does not have equivalent keys: whereas this is often just a minor weakness for the block cipher when used for encryption, it can be a catastrophic weakness for the derived hash function.

The cipher block chaining message authentication code (CBC-MAC) [ISO99], and its variation Cipher-based MAC (CMAC) [Dwo05], One-key MAC (OMAC) and Parallelizable MAC (PMAC, created by Phil Rogaway) are standardised methods to turn block ciphers into message authentication codes (MACs).

A hash function can be turned into a stream cipher if used in CTR mode. From this, since the hash function in CTR mode can be viewed as a PRNG, one could use the Granboulan–Pornin construction to turn it into a block cipher.

In [Sch96, § 14.11], Bruce Schneier shows how to use a hash function as a block cipher in CFB mode

$$C_i \longmapsto P_i \oplus H(K\|C_{i-1})$$
$$P_i \longmapsto C_i \oplus H(K\|C_{i-1})$$

together with other constructions.

A more direct approach follows the Luby-Rackhoff construction [LR86] (see also Section 1.3 on page 27): under the assumption that the has function is a cryptographically secure pseudorandom function, then three, resp. four rounds are are sufficient to create a block cipher that is pseudorandom permutation, resp. a "strong" pseudorandom permutation, and the block size is twice the size of the output of the hash function.

A remark by Schneier in [Sch96, § 14.11] applies to these last two constructions: "*While these constructions can be secure, they depend on the choice of the underlying one-way hash function. A good one-way hash function doesnot necessarily make a secure encryption algorithm. Cryptographic requirements are different. For example, linear cryptanalysis is not a viable attack against one-way hash functions, but works against encryption algorithms.*"

One way to both exploit the fact that hash functions accept inputs of arbitrary length and to provide better security, due to Ross Anderson and Eli Biham, will be described in the next subsection.

1.11.3 BEAR, LION, LIONESS, and AARDVARK

At FSE 1996, Ross Anderson and Eli Biham presented [AB96] three new block cipher designs, inspired by the results of Luby and Rachoff [LR86]. The three designs use a stream cipher S and a (keyed) hash function H to construct block ciphers. They are a variant of the unbalanced Feistel cipher design that does not repartition the output of a round - rather, they alternate source-heavy with target-heavy rounds. The resulting cipher is called an *alternating Feistel cipher* by Hoang and Rogaway in [HR10a, HR10b].

Let n be the block size (expressed in bits), which will typically be large - from 1 Kb to even several Mb. The keyed hash function uses a key K and compresses a message M of arbitrary length to a fixed size hash of k bits. The stream cipher takes a k-bit input and generates an arbitrarily long keystream.

Figure 1.8: BEAR

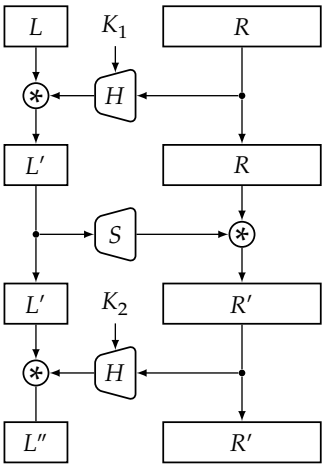


Figure 1.9: LION

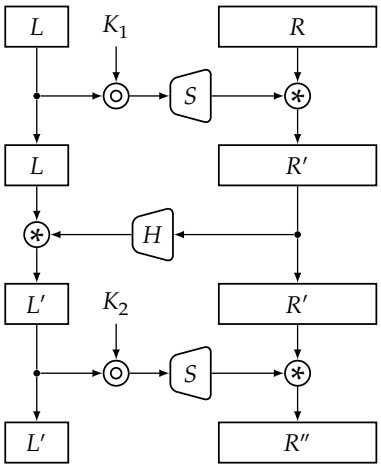
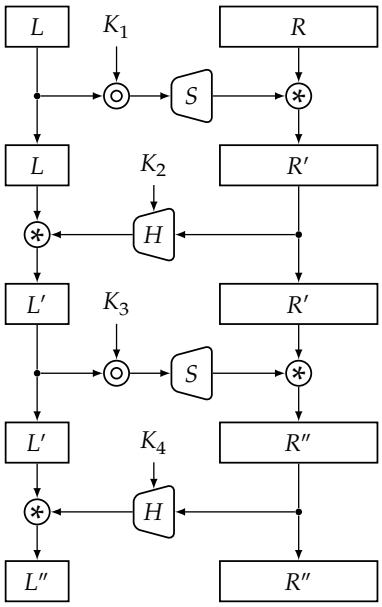


Figure 1.10: LIONESS



BEAR is defined as follows. The plaintext P is divided into two parts L and R whose sizes are $|L| = k$ and $|R| = n - k$. The key consists of two (independent) sub keys $K = (K_1, K_2)$, each of which have length greater than k. Encryption is done by

$$L' \leftarrow\!\shortmid L * H_{K_1}(R)$$
$$R' \leftarrow\!\shortmid R * S(L')$$
$$L'' \leftarrow\!\shortmid L' * H_{K_2}(R')$$

where $\circ$ is an invertible composition operation, such as bitwise XOR, word-wise or long integer addition, or even finite field multiplication à la IDEA (cf. Section 3.6 on page 142). The ciphertext is then the concatenation $L''\|R'$. BEAR encryption is represented graphically in Figure 1.8. Decryption is performed by executing the inverse steps.

It is assumed that the keyed hash function H:

(a) is one-way and collision-free, i.e. it is hard given Y to find X such that $H_K(X) = Y$, and to find unequal X and Y such that $H_K(X) = H_K(Y)$, and

(b) is pseudo-random, in that even given $H_K(X_i)$ for any set of inputs, it is hard to predict any bit of $H_K(Y)$ for a new input Y.

It is also assumed that the stream cipher $S(M)$:

(a) resists key recovery attacks, in that it is hard to find the seed X given $r = S(X)$;

(b) resists expansion attacks, in that it is hard to expand any partial stream of Y.

Under these assumptions, Anderson and Biham prove that breaking BEAR implies breaking both the stream cipher and the hash function – and thus they achieve better security than the

constructions exclusively based on the one-way function described in the previous subsection.

LION, depicted in Figure 1.9, is similar but it uses the stream cipher twice and the hash function (unkeyed) once:

$$R' \leftarrow\!\shortmid R * S(L \circ K_1)$$
$$L' \leftarrow\!\shortmid L * H(R')$$
$$R'' \leftarrow\!\shortmid R' * S(L' \circ K_2)$$

Also breaking LION implies breaking both the stream cipher and the hash function.

Similarly to the Luby-Rackoff construction [LR86], the three-round ciphers BEAR and LION are provably secure against an attacker that only has access to an encryption *or* a decryption oracle, but not to both, and four rounds are necessary to thwart an attacker that has access to both. For this reason Anderson and Biham also introduce the four round cipher LIONESS – represented in Figure 1.10 – that invokes both the stream cipher and the hash function twice. It uses four sub keys, which are used to mask the stream cipher seed and to key the hash function:

$$R' \leftarrow\!\shortmid R * S(L \circ K_1)$$
$$L' \leftarrow\!\shortmid L * H_{K_2}(R')$$
$$R'' \leftarrow\!\shortmid R' * S(L' \circ K_3)$$
$$L'' \leftarrow\!\shortmid L' * H_{K_4}(R')$$

Pat Morin showed [Mor96] that LION and BEAR are susceptible to meet-in-the-middle attacks, reducing the security margin by a constant factor, but not breaking the cipher. Morin also proposed another cipher, which he called AARDVARK, because *"all the exotic animal names were already being used"*. Let H be a unkeyed hash function, and H' a keyed hash function, both outputting k bits, and S a block cipher. AARDVARK encrypts the plaintext P into a cryptogram $C^* \| C'$ as follows:

$$C^* \leftarrow\!\shortmid H(P)$$
$$C' \leftarrow\!\shortmid P \oplus S(H'_K(C^*))$$

Decryption is then performed as $P = C' \oplus S(H'_K(C^*))$. This cipher has a significantly higher throughput than BEAR, LION, and LIONESS. In order to be secure, it requires H to be strongly collision free, H' to resist existential forgery, and S to resist expansion attacks. It is characterised by a constant ciphertext expansion.

Morin's attack have been discussed in [MPRS11a] (preprint on arXiv: [MPRS11b]). The observation is that *"the attack succeeds only because its brute force search on the round function contradicts the key-resistance of the hash function and of the stream function. So, whenever H or S remain key-resistant, both LION and BEAR are immune to such attacks."* The authors also show that these ciphers are actually immune to any efficient known-plaintext key-recovery attack that can use as input any number of plaintext-ciphertext pairs – even if the assumptions on the hash function and stream cipher are slightly weakened. Improvements on BEAR and LION, called BEAR2 and LION2, are presented, that key also the stream cipher by XORing a key to its input.

Chapter 2

Cryptanalysis

If you reveal your secrets to the wind, you should not blame the wind for revealing them to the trees.

Kahlil Gibran

In this chapter we describe mathematical attacks to block ciphers. We do not describe hardware attacks such as side-channel attacks and fault attacks.

2.1 Differential Cryptanalysis

Differential cryptanalysis is the first general cryptanalytic technique specifically introduced to break block ciphers. The main idea of differential cryptanalysis is to exploit properties of a cipher E like "if P and P' are two plaintext blocks such that $P^* = P \oplus \Delta P$, then it is likely that $E(P^*) = E(P) \oplus \Delta C$." Such correlations between *differences* in the inputs and outputs of a non-ideal block cipher are used to recover the key, usually through a chosen-plaintext attack.

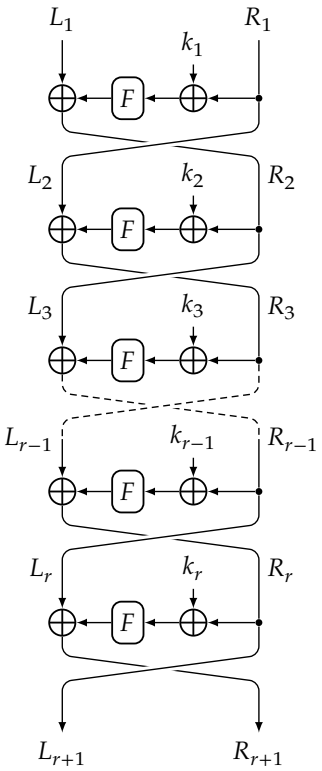
Differential cryptanalysis made its first appearance in the scientific literature in Biham and Adi Shamir's papers on DES-like ciphers [BS90, BS91a] and the first application to real ciphers was to FEAL [BS91c]. The technique did not receive much attention until it was further generalised and applied to DES [BS92]. DES shows a certain resistance to differential cryptanalysis which suggests that its designers knew about the technique a decade earlier. In fact, this was confirmed by Coppersmith [Cop94]:

After discussions with NSA, it was decided that disclosure of the design considerations would reveal the technique of differential cryptanalysis, a powerful technique that can be used against many ciphers. This in turn would weaken the competitive advantage the United States enjoy over other countries in the field of cryptography.

2.1.1 Fundamentals of Differential Cryptanalysis

The first observation that led to differential cryptanalysis is the following, that holds for most non linear mappings γ. Consider all pairs of inputs (x, x^*) to γ with a fixed non-zero difference $\Delta x = x^* - x$, and the corresponding output differences $\gamma(x^*) - \gamma(x)$. The output differences are in general not unique, but it is also true that these do not necessarily take all output values. In the particular case where the group operation $+$ is the XOR, each output difference occurs an even number of times. It turns out that the statistical distribution of these differences often presents some irregularities. Furthermore, if we know both the input and output difference to such a function, and the function is small enough to allow it to be tabulated, we can list all possible input pairs that generate the given output difference.

Figure 2.1: A Simplified Feistel Cipher (Notation for the Differential Cryptanalysis)



For each such non-linear function, such as a S-box, by enumerating all pairs with a given input difference, we obtain a distribution of output differences. So we choose an input difference such that a given output difference occurs with particularly high likelihood. Let us take the S-box S1 of DES, which is a 6×4 bit S-box, and thus it has 64 possible input differences. For the input difference `0x34` several output differences cannot occur (for instance `0x0` or `0x5`) but the output difference `0x2` occurs 16 times – and we know the exact input pairs that generate this difference. We can thus predict with probability 1/4=16/64 that if the input difference is `0x34`, then the output difference is `0x2`.

So what happens if we *do* observe an output difference of `0x2`? We know what actual input pairs could have generated it, and since one of these is equal to the chosen input pair XORed with the round key, we restricted the choice of the relevant round key bits to a fraction of the whole 6 bit input space, in this case 16 possible values out of 64.

This immediately leads to 0R Attacks on Feistel Ciphers, which we now describe.

2.1.1.1 Simple 0R Attacks on Feistel Ciphers

Consider a n-bit Feistel cipher E without expansion and compression in the F-Function, such as the one represented in Figure 2.1.

Suppose that we know the following: For a plaintext difference ΔP the ciphertext difference ΔC holds with probability $p > 2^{-n}$.

An important fact here is that while constructing the characteristic, we are computing the prob-

ability that $\Delta L_i \| R_i \rightarrow \Delta L_{i+1} \| R_{i+1}$ for $i = 1, 2, \ldots, r$ for differences $\Delta L_i \| R_i$ which are all known. The product of the probabilities $\mathscr{P}[\Delta L_i \| R_i \rightarrow \Delta L_{i+1} \| R_{i+1}]$ is a lower bound for the likelihood $p = \mathscr{P}[\Delta P \rightarrow \Delta C]$.

We start collecting $m = 2/p$ plaintext/ciphertext pairs, and we hope to find at least one pair of plaintexts P, $P^* = P + \Delta P$ such that for $C = E(P)$ and $C^* = E(P^*)$ the relation $C^* = C + \Delta C$ holds. The expected number of such pairs is $m(p + 2^{-n})$. Of these pairs, mp are *right* pairs, i.e. they result from the characteristic, and $m2^{-n}$ are *wrong* pairs, i.e. they occur by chance.

In particular, for a right pair, we do not only know ΔC, but also the difference $\Delta L_r \| R_r$.

If $p \gg 2^{-n}$ we can assume that all found pairs are right.

For any such pair, we know both the input difference to F, that is equal to $\Delta L_{r+1} = \Delta R_r$ – because adding the same round key to two values does not change their difference – and the output difference to F, that is equal to $\Delta L_r \oplus \Delta R_{r+1}$.

Now suppose for simplicity that F is a $\gamma\lambda$ structure, i.e. it is formed by an array of S-boxes followed by a linear transformation.

Inverting the linear transformation, we immediately determine the output differences at each S-box and since also the input difference is known, we have a list of possible candidate pairs for the input difference. Now, we *know* the actual inputs to the substitution layer: they are given by the left half of the ciphertext $R_r = L_{r+1}$ XORed with the r-th round key k_r. This immediately gives a list of candidates for the key bits that are associated to each S-box.

In general, the number of guesses for the key bits is smaller than all the possible choices by the same bits, and thus we can loop over these at each S-box and try all the possibilities for the remaining bits. We should be able to find the key in less than brute force time.

2.1.1.2 Constructing Differential Characteristics

Since finding correlated input and output differences for a complete cipher is in general a difficult problem, the differences of the internal states are studied as they evolve through the various operations of a product cipher. This is important also because, as we have just seen, even in a simple differential attack we need intermediate differences beside the initial and final ones. Differential trails for the whole cipher are thus obtained by chaining smaller ones.

Let the cipher be represented as the composition of several rounds

$$E = \rho_r \circ \cdots \circ \rho_2 \circ \rho_1 \tag{2.1}$$

where we can assume that the i-th round function ρ_i is keyed using round key $k_i = \psi_i(K)$. In general the rounds are equal and can be represented as a function of the round key and of the state, so $\rho_i = \rho[k_i]$, but we do not need this here. We shall also write

$$E_s = \rho_s \circ \rho_{s-1} \circ \cdots \circ \rho_1 \quad \text{for all } s \leqslant r$$

to denote truncated ciphers.

Let the plaintext be $P_1 = P$ and $P_{i+1} = \rho_i(P_i)$ for all i, we have that the ciphertext C is given by $C = P_{r+1} = \rho_r(P_r)$. Let P_i^* and C^* be defined similarly for a second plaintext P^* encrypted *with the same key*. For any quantity x we define $\Delta x = x^* - x$. The cryptanalyst then studies how ΔP_i

and ΔP_{i+1} are correlated. For brevity's sake we put

$$\delta_i := \Delta P_i \ , \ \Delta := \Delta P_1 \ , \text{ and } \ \nabla := \Delta P_{r+1} \ .$$

If p_i denotes the probability of the transition $\delta_i \rightarrow \delta_{i+1}$, i.e.

$$p_i = P_{\rho_i}(\delta_i, \delta_{i+1}) := \mathscr{P}[\rho_i(x + \delta_i) - \rho_i(x) = \delta_{i+1}] \ ,$$

which is computed over all the keys, then we have the differential characteristic (or trail)

$$\Delta = \delta_1 \xrightarrow{p_1} \delta_2 \xrightarrow{p_2} \delta_3 \dashrightarrow \delta_{r-1} \xrightarrow{p_{r-1}} \delta_r \xrightarrow{p_r} \delta_{r+1} = \nabla \ .$$

The nature of the operation used to compute P_{i+1}, resp. P^*_{i+1} from P_i, resp. P^*_i plays a fundamental role in the determination of the δ_i and of the transition probabilities.

Key-mixing by addition does not affect the differences, because $(s^* + k) - (s + k) = s^* - s$ for all s and k. Linear and affine operations do not affect the differences, or affect them in a predictable way: If λ is a linear operation, then $\lambda(s^* - s) = \lambda(s^*) - \lambda(s)$, therefore the difference is simply transformed by λ, and the constant term of an affine operation just disappears; bit-permutations of the state just reorder the bits of the differences in the same way.

In other words, only the non-linear components affect the differences, and the power of differential cryptanalysis is that it can study how these evolve. Hence, in the computation of the transition probabilitiy of ρ_i, only the non-linear components play a role, and the probability is obtained considering the values of ρ_i over all possible round key choices.

Also, note that if the function ρ_i is small, the value

$$p_i = P_{\rho_i}(\delta_i, \delta_{i+1}) := \mathscr{P}[\rho_i(x + \delta_i) - \rho_i(x) = \delta_{i+1}]$$

can be computed and tabulated explicitly. In other cases it can be computed as requested. For instance, if the round function is of type $\lambda \circ \gamma \circ \sigma[k_i]$ where $\sigma[k_i]$ is the addition of round key k_i, and λ, γ are linear and non-linear (cf. Section 1.4 on page 33). Then, σ is ininfluential for the computation of differences and λ can often be easily traversed. Suppose now that γ is a bricklayer construction from small S-boxes: then both input and output differences can be truncated to just the relevant bits and the transition probabilities through each S-box is first fetched, then the values are multiplied together to obtain the transition probability for the whole round.

The first long characteristics were iterative, i.e. they were a repetition of a single differential of form $\alpha \rightarrow \alpha$ [BS92], where α is a fixed difference of states of the cipher. This approach fails for modern primitives due to better diffusion properties.

Some remarks:

- Christian Rechberger [Rec09] provides a list of techniques used for the characteristic search, whose applicability strongly depends on the particular design.
- Differential cryptanalysis does not apply only to differences defined with the XOR operation. The difference can be defined also as the subtraction of integers – useful for cases where the native operation in the cipher, especially for key mixing, is addition – or as division modulo a prime – useful in ciphers where the native operation in the cipher is multiplication modulo

a prime, such as in IDEA (Section 3.6 on page 142). Sometimes tables of conversions of differences may be useful in case the S-box transforms one operation into another, such as logarithm or exponentiation S-boxes, as in SAFER (Section 3.8 on page 148). For the AES, the state is split into 8-bit words or bundles which are interpreted as elements of $\mathbb{F}_{2^8}$: this is compatible with bitwise XOR and at the basis of the square attack, that can be sees as a form of differential cryptanalysis.

- Differential cryptanalysis is a very versatile attack, and even though it was designed for iterated block ciphers it has many applications to stream ciphers and hash functions. Furthermore, there are several powerful generalizations and extensions of this technique. We shall desribe the most significant ones starting with Subsection 2.1.4 on page 80.

2.1.1.3 1R Attacks and Beyond

Once we have a differential over, say, s rounds, we may want to use it to break at least $s+1$ rounds of the cipher. Suppose, then, that we have a differential characteristic over s rounds

$$\Delta := \delta_1 \xrightarrow{p_1} \delta_2 \xrightarrow{p_2} \delta_3 \dashrightarrow \delta_{s-1} \xrightarrow{p_{s-1}} \delta_s \xrightarrow{p_s} \delta_{t+1} =: \nabla$$

where the transition probability from δ_i to δ_{i+1} is i and $p = \prod_{i=1}^{s} p_i$. Now we describe how to use the characteristic to attack $r = s+1$ rounds of the cipher:

1. We use the encryption oracle to compute the encryption of pairs of plaintexts P, P^* with $P^* = P + \Delta$, obtaining two ciphertexts C and C^*. The hope is that the encryptions, reduced by one round, will satisfy the expected characteristics, i.e. that $P_r^* - P_r^* = \nabla$.

2. In order to verify this, we decrypt through the last round – usually going through some of the S-boxes – using all possible values of the round key k_r that affect the traversed S-Boxes. If the difference between these partial decryptions agrees with the expected value δ_r, then we have a possible round key candidate.

   (Alternatively, we can turn the requirement that the difference be δ_r into a system of equations to be solved in the round key bits. See [AC09] and [WSMP11].)

   Optionally, we can further filter the candidate round keys by analysing the input and output differences of the second-to-last round exactly as we did in 0R attacks with the last round (for this we must use the difference δ_{r-1} as well). We obtain candidates for some of the key bits and at least one of these must agree with the currently guessed round key.

   For each value of the round key k_r we have a *counter*, initially set to zero, and each time a value passes the tests we have just described, we increase the corresponding counter.

   During this stage we will obtain some false positives from wrong pairs, but, as in the 0R case, if the probability of the characteristic is high enough, their impact will be negligible.

3. After enough plaintext pairs have been processed, we can test the candidate round keys with outstanding counters. If some bits of the master key are still undetermined, these are recovered by brute force.

The advantage of 1R attacks with respect to 0R attacks is that we can use shorter characteristics with higher probabilities to attack the same number of rounds, although the identification of

the right pairs is somewhat worse.

Depending on the structure of the cipher, it may be possible to extend attacks by more than one round. In the case of the DES, 2R and 3R round attacks are possible, because differences involving only one S-box in the last round can be traced back through a limited number of S-boxes for up to 3 rounds, and therefore the number of key bits to be guessed until we reach the last round of the differential characteristic is still significantly smaller than the full key space.

2.1.2 Markov Ciphers and the Wrong-Key Randomisation Hypothesis

Traditionally, in 1R, 2R, etc. attacks on a r rounds cipher, the attacker takes the encryption oracle for the target r rounds of the cipher and (partially) decrypts the last $r - s$ rounds with all possible (partial) round keys. A counter is increased for the current round key(s) guess if the computed difference fits the expected output difference of the first s rounds. Afterwards, these keys are ranked according to their counters, that is, the attacker first tries the key with the highest counter, then the one with the second highest counter, etc.

If the guesses for the round keys of the last $r - s$ rounds are correct, and sufficiently many plaintexts are collected, then we expect that the expected output difference will occur more often than other differences. On the other hand, the cryptanalyst's hope is that wrong key guesses will not contribute to just one or a few of the counters, creating a bias, but will be randomly distributed. There will be some noise, but the right guess will most likely correspond to one of the highest counters. Indeed, in [LMM91] it is stated that for a wrong key guess the corresponding counter is distributed as for a random permutation, i.e. uniformly – this means that for an incorrect key candidate the probability of observing the differential is $1/(2^n - 1)$, where n is the block size. In DBLP:journals/jmc/DaemenR07 Joan Daemen and Vincent Rijmen show that a more correct description of the probability distribution is a binomial distribution with parameter $1/(2^n - 1)$. In both cases, the probability of observing the differential is much lower than the probability of the characteristic: this is called the **Wrong-Key Randomisation Hypothesis (WKRH)** in the literature.

Let us now formalise this intuition.

An iterated cipher E is the composition of equal round functions ρ which are simply parametrised by the round keys $k_i = \psi_i(K)$, i.e. with respect to notation (2.1) we have $\rho_r = \rho[k_r]$:

$$E[K] = \rho[k_r] \circ \cdots \circ \rho[k_2] \circ \rho[k_1] \ .$$

Let now $s < r$, and

$$E_s[K] := \rho[k_s] \circ \rho[k_{s-1}] \circ \cdots \circ \rho[k_1]$$

be the cipher E reduced to s rounds. Further, let

$$T_s[K] := \rho[k_r] \circ \ldots \circ \rho[k_{s+1}]$$

denote the composition of the last $r - s$ rounds, so that $E = T_s \circ E_s$ (the letter of T is chosen being the initial of "tail").

The cryptanalyst considers the probabilities

$$P_{E_s}(\Delta,\nabla) := \mathscr{P}[E_s(x + \Delta) - E_s(x) = \nabla]$$

and she is interested in input/output difference pairs (Δ,∇) with large $P_{E_s[K]}(\Delta,\nabla)$. The context is that of a key recovery attack, where the attacker starts by collecting N (unordered) plaintext/ciphertext pairs encrypted with a fixed, unknown key K. If the number of pairs following the given differential, i.e. the respective *counter*, is denoted by

$$D^{(N)}_{E_s[K]}(\Delta,\nabla) \ , \tag{2.2}$$

then its expected value is $D^{(N)}_{E_s[K]}(\Delta,\nabla) = NP_{E_s[K]}(\Delta,\nabla)$. However, since we do not know K, we cannot consider $P_{E_s[K]}(\Delta,\nabla)$ directly: This hurdle is addressed by the concept of *Markov cipher* and by the *hypothesis of stochastic equivalence*.

The cipher E is said to be Markov if, for all choices of x, $\Delta \neq 0$ and $\nabla \neq 0$, the probability

$$\mathscr{P}[\rho[k](x+\Delta) - \rho[k](x) = \nabla]$$

depends only on Δ and ∇, and does *not* depend on x, *when the subkey k is uniformly random*.

If an iterated cipher is Markov and its round keys are independent, then the sequence of differences at each round output forms a Markov chain, i.e. the transition probabilities in one round are independent from those in the other rounds. Thus, we can omit the key in some notations, for instance we can write $P_\rho(\delta_i,\delta_{i+1})$ or even $P(\delta_i,\delta_{i+1})$ in place of $P_{\rho[k]}(\delta_i,\delta_{i+1})$.

If we consider *all* trails for E_s of the form $\Delta = \delta_1 \to \delta_2 \to \ldots \to \delta_s \to \delta_{s+1} = \nabla$ starting with Δ and ending with ∇, we can compute

$$\tilde{P}_{E_s}(\Delta,\nabla) := \frac{1}{\#K} P_{E_s[K]}(\Delta,\nabla) = \sum_{\delta_2,\delta_3,\ldots,\delta_s} P_\rho(\Delta,\delta_2)\, P_\rho(\delta_2,\delta_3) \cdots P_\rho(\delta_s,\nabla)$$

without having to know the actual intermediate values, but only considering the differences. Note that in differential cryptanalysis cryptanalists are often satisfied when they can find $\Delta \neq 0$ and $\nabla \neq 0$ such that for a single trail they can compute a lower bound for $\tilde{P}_{E_s}(\Delta,\nabla)$ that is sufficiently large enough than 2^{-n}.

The *hypothesis of stochastic equivalence* (see [LMM91]) states that for almost all keys we expect $P_{E_s[K]}(\Delta,\nabla) \approx \tilde{P}_{E_s}(\Delta,\nabla)$ which implies that $D^{(N)}(\Delta,\nabla) \approx N\tilde{P}_{E_s}(\Delta,\nabla)$ for almost all keys. This approximation has to be understood as expected value taken *over all expanded keys* (as opposed to "over the expansions of all master keys"). In other words the hypothesis assumes independent round keys, which is often very far from reality - still, it is an often reliable model.

Now, suppose an attacker has found a good characteristic for the first s rounds, has access to an encryption oracle $E = E_r$ for the first r rounds with $r > s$, that can only be taken as a whole, and knows the specification of the cipher, so she can implement it with chosen keys.

The attacker will guess round keys $k^*_{s+1}, k^*_{s+2}, \ldots, k^*_r$, for the last $s-r$ rounds. Let us write k^* for the vector formed by them, and define

$$T_s[k^*] := \rho[k^*_r] \circ \ldots \circ \rho[k^*_{s+1}] \ . \tag{2.3}$$

The attacker is computing a function

$$T_s^{-1}[k^*] \circ E[K] = T_s^{-1}[k^*] \circ T_s[K] \circ E_s[K] = (T_s^{-1}[k^*] \circ T_s[K]) \circ E_s[K] \tag{2.4}$$

composed of the encryption oracle and of a function she has defined.

If the round keys for the last $s - r$ rounds are correctly guessed and enough samples are taken, then the expected output difference will stand out, because in this case $T_s^{-1}[k^*] \circ E[K] = E_s[K]$. The WKRH describes instead what happens when the wrong round keys are guessed for the last $r - s$ rounds. It at least some $\psi_j(K) \neq k_j^*$ for $s < j \leqslant r$, we have that $T_s^{-1}[k^*] \circ E[K]$ is a different function from $E_s[K]$, because $T_s^{-1}[k^*]$ and $T_s[K]$ do not cancel each other. According to the WKRH the values of the output differences of the function defined in Equation (2.4) behave effectively like random values, i.e. the function is for the cryptanalyst (nearly) indistinguishable from a random function. The consequence is that the output difference ∇ will *not* stand out, and in fact the WKRH states that all output differences will occur with (roughly) the same likelihood. This is, essentially, the original attack by Eli Biham.

The WKRH will be generalised in Subsection 2.1.9 on page 85.

2.1.2.1 Remarks on the Wrong-Key Randomisation Hypothesis

1. The validity of the Wrong-Key Randomisation Hypothesis is proven under the assumption that the round keys are independent from each other, which is almost never the case in concrete designs. When the likelihood of the considered characteristic is large enough, this is often non relevant, but in border cases deviations from uniform behaviour may impact the heuristic evaluations of attack complexity.

2. The function considered in (2.6) is the composition of the (truncated) encryption function E_s with a function

$$\xi := \xi_s[K, k^*] := T_s^{-1}[k^*] \circ T_s[K] \tag{2.5}$$

   which in turn is the composition of a parametrised function *with its inverse*, where the parameters K and k^* are independent.

   Assuming that $\xi \circ E_s$ behaves like a random function, for a function E_s that is "understood" (i.e. some statistical properties are known) is not completely equivalent to assuming that ξ is a random function, but often close enough.

   If the parameters K and K^* are sufficiently close to each other it can be debated whether ξ indeed behaves like a random function (or even as a good "randomisation function" for the known statistical properties of E_s). In other words, if the distance between K and K^* is sufficiently small (with respect to the fundamental key mixing operation of the cipher), there is no guarantee that the quantities x and x' where

$$x' := \xi_s[K, k^*](x)$$

   are not correlated to each other. This may skew the distribution of the counters, which will no longer look like a homogeneous distribution, but may retain some of the characteritics expected when the right key is guessed. In particular, ξ may map the expected output difference with high probability to a different one, which could also stand out. If such a new output difference for the wrong key guess is found then some assumptions on the relation between the guessed round keys and the correct ones may be done.

   In fact, it is easy to envision a situation where this scenario is realised: $s = r + 1$, i.e. we are guessing just the last round; and the function ρ is simple, such as an S-box layer and a limited diffusion layer.

This observation implies that:

- A proper diffusion layer is necessary to achieve good security and make less rounds exploitable in first instance;
- We have the apparent paradox that a poor diffusion layer will create several false positives that may actually slow down the attacker...
- ... unless the attacker learns how to use false positives to further narrow down the round key guesses!

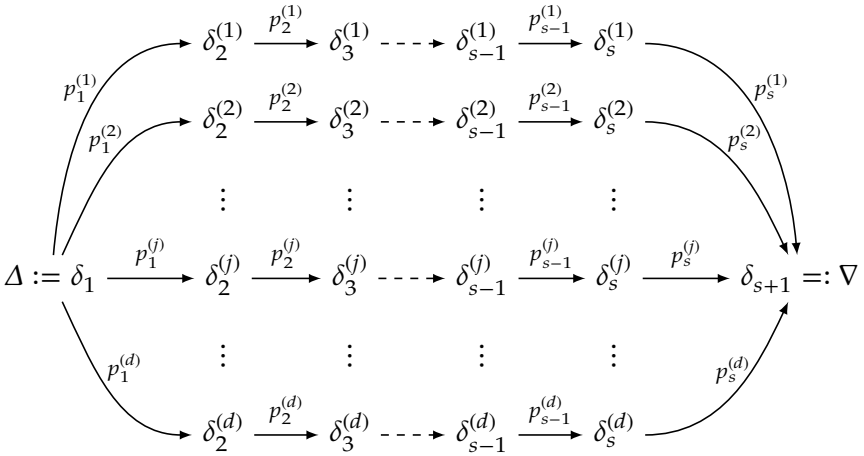
3. Regarding the WKRH, in [BGN12a], Céline Blondeau, Benoît Gérard and Kaisa Nyberg claim "*Assuming that this hypothesis does not hold would mean that $r+1$ rounds of the cipher are distinguishable and hence the attacker should be able to attack more rounds. As a consequence, this hypothesis is quite reasonable as soon as the attacker targets the largest number of rounds he is able to attack (which is typically the case).*"

   This remark is questionable since we are composing a cipher with the inverse of one rounds, which is a *different* function than one normal additional round – as we just observed in the previous paragraph – unless the cipher has an involutory structure. A near-guess for the round key may just confuse the analysis, because the added inverse round may indeed partially invert the last round of the oracle – and it is not yet clear how the analyst may be use these near matches to narrow the search for the correct round key.

4. A statistical model that takes into account per-wrong key biases is described by Andrey Bogdanov and Elmar Tischhauser in [BT13]. In [MC13a] James McLaughlin and John A. Clark show a concrete instance where this assumption does not hold. These results are applied to linear cryptanalysis, but the existence of biases may affect differential cryptanalysis as well.

2.1.3 Multiple Differentials

In the original presentation of differential cryptanalysis [BS90], a single differential is exploited, whereas in [BS91a] the use of several differentials having the same output difference is considered as well. For instance, one may be able to find several differential trails with the same initial and final differences, such as



where not only $\prod_{i=1}^{s} p_i^{(j)} \gg 2^{-n}$ holds for all probabilities ($1 \leqslant j \leqslant d$), but they are of similar magnitude, otherwise the trails with smallest probabilities would give a negligible contribution. This means that in 1R-type attacks, it can be verified, for each plaintext/ciphertext pair, *which* of the admissible differences actually gives non-empty sets of key candidates, and for these the counters are then updated.

Wrong guesses arising from the wrong trails (even for a right pair) will contribute to the wrong counters. However, by an argument similar to the wrong key randomisation hypothesis, their contributes will be essentially random. Therefore this approach should increase the effectiveness of differential cryptanalysis.

In [BG11b, BG11a] Céline Blondeau and Benoît Gérard study the use of multiple differentials and apply their analysis to construct attacks on 18 rounds reduced PRESENT, currently the best differential cryptanalysis of that cipher that does not make use of algebraic methods (see Section 3.29 on page 204).

2.1.4 Truncated Differentials

An important improvement – already mentioned in [BS91a] – is to ignore part of the state when considering the differentials, in other words not all the bits of the differentials are fixed. Lars Knudsen called these types of differentials *truncated differentials* [Knu94], where only a part of the difference in the ciphertexts (after a possibly reduced number of rounds) can be predicted. He used them to analyze various ciphers, such as SAFER [KB96] (Section 3.8 on page 148), and Skipjack [KRW99] (Section 3.14 on page 164).

An important type of truncated differential is the word-wise truncated differential, where the difference itself is not considered, but instead the differences are divided into two classes, for instance zero differences and nonzero differences. In these cases the data blocks are divided into *words* (for instance nibbles, bytes, or 16-bit words), and the analysis only considers whether the difference of a word is expected to be zero or not.

2.1.5 Higher-order Differential Cryptanalysis

Higher-order differential cryptanalysis, as defined by Xuejia Lai [Lai94] is a generalisation of differential cryptanalysis where higher order differences are considered in place of simple differences. (See also [Knu94].) Higher-order differentials are defined in an analogous way to higher order derivatives.

Let $f : S \to T$ be a cipher, where an addition operation is defined on S and T (this is usually the XOR, but it does not hurt to consider a more general setting). Recall that the differential of f at the point $a \in S$ is defined as

$$\Delta_a f(x) = f(x + a) - f(x) \ .$$

The i-th differential of f at the points $a_1, \ldots, a_i$ is defined as

$$\Delta^{(i)}_{a_1,\ldots,a_i} f(x) = \Delta_{a_i}\left(\Delta^{(i-1)}_{a_1,\ldots,a_{i-1}} f(x)\right) \ .$$

Applying this definition recursively, we obtain

$$\Delta^{(i)}_{a_1,\ldots,a_i} f(x) = (-1)^i \sum_{\emptyset \subseteq I \subseteq [1..i]} (-1)^{\#I} f(x + a_I) \quad \text{where} \quad a_I = \sum_{i \in I} a_i \ .$$

If we are considering functions between vector spaces S and T over $\mathbb{F}_2$), then it is easy to see that the points $a_1, \ldots, a_i$ must be linearly independent for the i-th differential to be non-zero. Denote the linear space generated by the points $a_1, \ldots, a_i$, by $\langle a_1, \ldots, a_i \rangle$, then

$$\Delta^{(i)}_{a_1,\ldots,a_i} f(x) = \sum_{\alpha \in \langle a_1,\ldots,a_i \rangle} f(x+\alpha) \ .$$

The important observation is that any derivation decreases the degree of the function. So, for any function $f : \mathbb{F}_2^n \to \mathbb{F}_2^m$ the n-th derivative is constant, and if the function is invertible, the $(n-1)$-th derivative is already constant.

But, a constant derivative is a constant differential, and if this can be found, it can be used exactly as a classic differential with probability 1 – and a lower degree differential will likely exhibit strong biases. As with first order differential cryptanalysis, successful search for higher order differentials with a significant bias will lead to relations involving the key (and ignoring low degree ciphertext dependent components), and thus speed up the search for key candidates.

So far, the direct application of higher-order differential cryptanalysis in the form just explained to block ciphers have let to very limited results.

On the other hand following developments need to be mentioned:

1. Integral attacks (square attack) are higher-order differential attack with truncated differentials (Subsubsection 2.1.5.1).

2. Cube attacks, at least in the boolean function case, are clearly a form of higher order differential attack (Subsubsection 2.1.5.2 on the next page).

3. At the core of the boomerang attack (Subsection 2.1.6 on page 83) there is, essentially, a construction of second order differentials, but, technically, the boomerang attack is not a second order differential attack.

4. Hidema Tanaka, Kazuyuki Hisamatsu, and Toshinobu Kaneka [THK99] break five rounds of MISTY1 with the "FL" function omitted (see Subsection 3.18.8 on page 174) using multiple seventh order differentials.

   Related to this, another recent exception is a 32nd order differential analysis of eight rounds of MISTY-2 without FL functions [IKE+13] (see Subsection 3.18.8 on page 174).

2.1.5.1 Integral Cryptanalysis (Saturation Attacks, Multiset attacks)

Integral cryptanalysis [KW02] was initially presented as a dual to differential cryptanalysis and it is the best known attack on AES (Section 3.20 on page 180).

Variants of this attack appeared earlier in the literature, under the names square attack [DKR97] and saturation attacks [Luc00]. These attacks are sometimes grouped under the name *Multiset attacks*.

In these attacks the attacker looks at a large, carefully chosen set of encryptions, in which parts of the inputs are the same, and other parts cycle through several, or even all, possibilities: For instance, the input may have all bytes equal to each other except for the least significant one, where all 256 possible values are taken.

Then, the outputs are combined, often just added together (whence the name integral as opposed to differential), and probabilities for particular values of this combination – possibly truncated – are considered.

For instance, suppose that all the possible values in a given plaintext byte are considered, the rest of the plaintext being fixed, and that the corresponding ciphertexts (possibly after a reduced number of rounds) are added together. This is clearly just a eighth order differential. If just a few bits of the ciphertext are considered in the estimation of the likelihood of the differential, will have a eighth truncated order differential. This is precisely the type of attack called *square* attack (because it was used to break SQUARE Section 3.11 on page 158)

Such attacks are among the most effective ones for reduced round Rijndael, due to the byte-wise structure of the cipher.

2.1.5.2 AIDA/Cube Attack

A cube attack is a cryptanalytic method used in order to retrieve secret values from a *tweakable polynomial*, i.e. a polynomial depending both on secret and public variables – typically the representation of a cryptographic algorithm where the public variables can be either initialisation vectors or known/chosen plaintexts. The attack is usually a key-recovery attack.

The attack was first presented by Michael Vielhaber as AIDA [Vie07] where it was applied to a simplified version of the stream cipher Trivium. A very similar attack was then presented by Itai Dinur and Adi Shamir [DS08b, Jou09] and then used by the same authors together with Jean-Philippe Aumasson and Willi Meier against reduced versions of Trivium and the hash function MD6 [ADMS09].

In the Boolean case, which is by far the most important case, the cryptanalyst computes higher order derivatives of the polynomial. This precomputation uses sums of values of the polynomial on subspaces of public variables - the so-called cube, that is in fact a hypercube, or the linear span of the set of base points for the computation of the higher order of the differential - to obtain a system of linear equations in the secret variables which can be solved using usual methods.

Cube attacks can be used together with other techniques, in particular when these can provide extra information on the secret bits, such as side channel attacks [DS09] or partially successful algebraic attacks.

For instance, Gregory Bard et al. [BCN+10] apply these attacks to the KATAN family of block ciphers (see Section 3.31 on page 207). They can break 60 rounds of KATAN-32 in time 2^{39}, 50 rounds of KATAN-48 in time 2^{49}, and 40 rounds of KATAN-64 in time 2^{35} using the cube attack alone. But, using a single bit of leaked state, all 254 rounds of KATAN-32 can be broken.

Usually a cube attack consists of two phases, an offline phase and an online phase:

1. Offline phase (precomputation)

   (a) Search for public variables (IV, plaintext) whose (higher order) derivative is a linear combination of key bits.

   (b) The linearity is detected via probabilistic testing.

(c) Then, the corresponding equations are reconstructed bit-per-bit.

2. Online phase

(a) Evaluate each linear equation detected during precomputation varying the public bits.

(b) Solve the linear system obtained.

2.1.6 Boomerang Attacks

David Wagner published the *boomerang attack* in 1999 [Wag99] and used it to break the COCONUT98 cipher. It is an adaptive chosen plaintext and ciphertext attack that makes a clever use of a second-order differential.

The attack considers the cipher E as the cascade of two subciphers, i.e. $E = E^2 \circ E^1$ (the decryption function is D and splits as $D = D^1 \circ D^2$). Then, two differentials are necessary:

- A good differential, say with input differential Δ and output differential Δ^*, for the encryption operation of E^1; and
- A good differential for the *decryption* operation of E^2, i.e. D^2.

Thus, the two differentials "meet" in the middle of the cipher E and together cover it completely. The attack, in its simplest form, proceeds as follows:

1. Choose a random plaintext P and calculate $P^* = P + \Delta$.
2. Request the encryptions $C = E(P)$ and $C^* = E(P^*)$ of P and P^*.
3. Calculate $\hat{C} = C + \nabla$ and $\hat{C}^* = C^* + \nabla$.
4. Decrypt $\hat{C}$ and $\hat{C}^*$, i.e. compute $\hat{P} = D(\hat{C})$ and $\hat{P}^* = D(\hat{C}^*)$.
5. Verify if $\hat{P}^* = \hat{P} + \Delta$, i.e. whether the differentials hold or not.

The four ciphertexts $P, P^*, \hat{P}, \hat{P}^*$ are called a *quartet*. If the two differentials have probabilities p and q respectively, the probability of a quartet is p^2q^2. When a quartet is found, the situation is depicted in Figure 2.2 on the following page, where the sought differentials for E^1 and D^2 are shown. At this point, a differential cryptanalysis attack can be mounted on the cipher.

Note that this is a order 2 differential of type $0 \rightarrow 0$. Indeed:

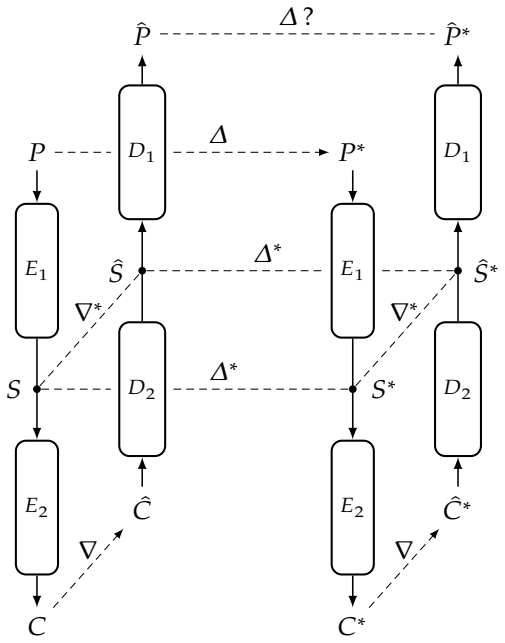
$$0 = \Delta - \Delta = (\hat{P}^* - \hat{P}) - (P^* - P) \rightarrow (\hat{C}^* - \hat{C}) - (C^* - C) = (\hat{C}^* - C^*) - (\hat{C} - C) = \nabla - \nabla = 0$$

but note that *higher order differentials of type* $0 \rightarrow 0$ *need not be trivial for order higher than 1.* This is a fundamental difference with respect to classical differential cryptanalysis.

Refinements on the boomerang attack have been published:

- The *amplified boomerang attack* [KKS00] is a known plaintext attack as opposed to adaptive. The attacks relies on the following idea: Encrypt many plaintext pairs, and hope that some

Figure 2.2: Construction of a Boomerang



quartet satisfy the conditions of the boomerang attack. The probability of a quartet to become a right quartet is $2^{-n-1}p^2q^2$ where n is the block size, hence it may seems that the attack is doomed to be considerably slower. However, the technique works well when many quartets are chosen by the birthday paradox. Also, it profits from the fact that it exploits several (unknown) differentials at the same time, so the number of possible XOR differences that can be found increases considerably. This principle is called the *boomerang amplifier*.

- The *rectangle attack* [BDK01b, BDK01c], that aims at improving the probabilities of the amplified boomerang attack by using multiple differentials in each sub-cipher.

The boomerang attack and its refinements have allowed new avenues of attack for many ciphers previously deemed safe from differential cryptanalysis, for instance reduced round SAFER++ (Section 3.8 on page 148), SQUARE (Section 3.11 on page 158), reduced round MARS (Section 3.16 on page 167), full round SHACAL-1 (Subsection 3.21.1 on page 187) under a related-key assumption, reduced round Camellia (Section 3.18 on page 170), and many more.

2.1.7 Impossible Differentials

Impossible differential cryptanalysis was introduced by Johan Borst, Lars Knudsen, and Vincent Rijmen in [BKR97]. One of the first notable uses of this attack was in Lars Knudsen's 1998 report [Knu98] where he introduced his AES candidate, DEAL (Subsection 3.19.1 on page 177). There, he also observed that Feistel ciphers with bijective round functions have 5-round impossible differentials (however, Knudsen did not use the terminology "impossible differentials").

The attack relies on finding an impossible (or extremely unlikely) event through a reduced part of a block cipher – an event that never happens if the key is wrongly guessed. The nature of this event is that of a differential, i.e. it is usually the condition on the output difference of a certain number of rounds given a certain input difference. Then, secret key bits are guessed, and

those that lead to this impossible event are discarded. An important difference with respect to classic differential cryptanalysis is that whereas in classic differential cryptanalysis the involved key bits are explicitly computed or restricted, in impossible differential cryptanalysis they are determined by exclusion. This attack is faster than brute force whenever only some specific key bits must be correctly guessed in order to verify the occurrence of the event and the remaining key bits are irrelevant. The attack is then completed by a brute force search on the remaining bits or by using other impossible differentials.

The name "impossible differential" was introduced by Eli Biham, Alex Biryukov and Adi Shamir at the CRYPTO '98 rump session, when they presented their attack on 31 out of 32 rounds of Skipjack and 4.5 out of 8.5 rounds of IDEA. The papers appeared the following year [BBS99a, BBS99b]. The journal version of [BBS99a] appeared in 2005 [BBS05]. The paper [BBS99b] describes a relatively efficient specialized method for finding impossible differentials called a *miss-in-the-middle* attack, which consists of finding "two events with probability one, whose conditions cannot be met together."

The technique has since been applied to many other ciphers, such as: Khufu and Khafre, E2, variants of Serpent, MARS, Twofish, KASUMI, Rijndael, CRYPTON, mCrypton, HIGHT, Zodiac, Hierocrypt-3, TEA, XTEA, Mini-AES, ARIA, Camellia, CLEFIA, and SHACAL-2.

2.1.8 Improbable Differentials

Improbable differential cryptanalysis introduced in [Tez10a, Tez10b] to analyze CLEFIA (Section 3.28 on page 201) uses a differential that is *less* probable when the correct key is used, but it is not an impossible differential. This differential then acts as a distinguisher, and repeated sampling with variable random plaintexts is used to determine whether the correct key (part) was guessed. The validity of the improbable differential cryptanalysis has been recently challenged [Blo13].

2.1.9 Unified Approaches to Differential Cryptanalysis

The fundamental observation is that if the round keys for the last $r - s$ rounds are correctly guessed, then (in the notation of Subsection 2.1.2 on page 76) the function $T_s^{-1}[k^*] \circ E[K]$ is equal to the s-round function $E_s[K]$ with a predictable behaviour, whereas if the round keys guesses are wrong, then $T_s^{-1}[k^*] \circ E[K]$ will be a r-round function whose characteristics are expected to be much closed to those of a random function than $E_s[K]$. This in fact applies to all forms of differential cryptanalysis, which then consist in applying a *distinguisher* to $T_s^{-1}[k^*] \circ E[K]$ to establish whether we have the "known" function or a different one. In simple differential cryptanalysis the distinguisher consists in verifying an outstanding characteristic, in impossible or improbable differential attacks the distinguisher is a low probability characteristic, in higher-order cryptanalysis it is a combination of more than two values, and so on.

Martin Albrecht and Gregor Leander in [AL12] provide a unified framework for these and other types of differential cryptanalysis. In order to do this, for a fixed input difference they analyse the vector of the counters for all possible output differences, and compare them to the expected counters for the correct and wrong round key guesses. They consider small ciphers because these allow to compute statistical distributions more precisely - however their results can be applied at least in theory to any Markov cipher.

For notation we refer to Subsection 2.1.2 on page 76, and we shall extend the arguments clos-

ing that section in order to provide the generalisation. The first generalisation with respect to those arguments is that instead of just picking a frequent counter we can consider very rare or vanishing counters (and we obtain improbable and impossible differential cryptanalysis) or take counters for whole sets of values (getting multi-set differential cryptanalysis).

Going further, we can look at the distribution of the values $D^{(N)}_{E_s[K]}(\Delta,\nabla)$ (cf. (2.2)). This was studied in [DR07] (see also [DR05]) and [BG10]. It turns out, considering $D^{(N)}_{E_s[K]}(\Delta,\nabla)$ as the results of N independent Bernoulli trials with success probability $\tilde{P}_E(\Delta,\nabla)$ leads to a precise model of the actual distribution. More precisely, according to Assumption 1 in [AL12], denoting by $\mathscr{B}(n,p)$ the Binomial distribution with n tries and success probability p, a reasonable approximation for the distribution of $D^{(N)}_{E[K]}(\Delta,\nabla)$ is given by the binomial distribution $\mathscr{B}\left(N,\tilde{P}_{E_s}(\Delta,\nabla)\right)$:

$$\mathscr{P}[D^{(N)}_{E_s[K]}(\Delta,\nabla)=c]=\binom{N}{c}\left(\tilde{P}_{E_s}(\Delta,\nabla)\right)^c\left(1-\tilde{P}_{E_s}(\Delta,\nabla)\right)^{N-c}$$

where the probability is taken over random keys K.

We can then rephrase the Wrong-Key Randomisation Hypotheses in this setting. As in Subsection 2.1.2 on page 76, we consider guesses for the round keys $k^*_{s+1},k^*_{s+2},\dots,k^*_r$ of the last $r-s$ rounds, write k^* for the vector formed by them, and define $T_s[k^*]$ as in (2.3). Then,

$$D^{(N)}_{T_s^{-1}[k^*]\circ E[K]}(\Delta,\nabla)\approx\mathscr{B}(N,2^{-n})\ , \tag{2.6}$$

where n is the bit size of the cipher's state. This definition slightly generalises the definition of WKRH as given in [AL12, Assumption 2].

The power of modelling $D^{(N)}_{E_s[K]}(\Delta,\nabla)$ as the results of N independent Bernoulli trials is that we can consider all possible output differences at once, for both the right key guess and the wrong key guesses. We compute the distribution for the right key guesses once for all, i.e. we compute the likelihoods of all possible output differentials for a fixed input differential, we also compute the distribution in case of a wrong key guess as in Equation (2.6).

For normal ciphers, it is impossible to compute the likelihoods of all possible output differentials, but Albrecht and Leander observe that it is possible for small ciphers. For instance, for a simple SPN cipher where a round is composed of an S-box layer and of a linear diffusion layer, we take A as the transformation matrix obtained by composing the block diagonal matrix with difference distribution matrices on the main diagonal and a matrix representing the linear layer. Then a probability vector representing the chosen initial difference is multiplied by this matrix as many times as the length of the characteristic(s) to be constructed, for instance:

$$\begin{array}{r} \Delta=0 \\ \Delta=1 \\ \vdots \\ \Delta \\ \vdots \\ \Delta=2^n-2 \\ \Delta=2^n-1 \end{array} : \begin{pmatrix} 0\\0\\\vdots\\0\\1\\0\\\vdots\\0\\0 \end{pmatrix} \xrightarrow{A} \begin{pmatrix} 0\\1/4\\\vdots\\1/8\\1/16\\0\\\vdots\\1/4\\0 \end{pmatrix} \xrightarrow{A} \begin{pmatrix} 0\\1/16\\\vdots\\7/256\\3/32\\1/4\\\vdots\\1/16\\1/4 \end{pmatrix} \xrightarrow{A} \begin{pmatrix} 0\\3/32\\\vdots\\19/1024\\3/128\\7/64\\\vdots\\1/128\\1/16 \end{pmatrix} \tag{2.7}$$

(This example is taken from Martin Albrecht's presentation of [AL12] at SAC 2012.)

The final vector is the probability distribution of the output differences. (That this can be done should not be a surprise: we are modelling a Markov cipher as a Markov process, i.e. according to its definition).

Now, for a given guess k^* of the last $r - s$ round keys, we *sample* the resulting counters distribution, and we denote by

$$\mathscr{D}^{(N)}_{T_s^{-1}[k^*]\circ E[K]}(\Delta) = \left[D^{(N)}_{T_s^{-1}[k^*]\circ E}(\Delta, i)\right]_{i=1,2,\ldots 2^n-1}$$

the vector of all counters corresponding to all output differences after encrypting N plaintext-pairs with difference Δ. It is here assumed that the corresponding distribution behaves like a multinomial distribution where each component is binomially distributed.

At this point, different techniques can be used to verify to which distribution the sampled distribution belongs (we shall return to this later) – and thus to verify whether the round key guesses were right or wring.

Using the whole counters table, not only classical differential cryptanalysis can be subsumed, but also impossible, improbable differential cryptanalysis, and considering several values at once also multi-set differential cryptanalysis (including integral cryptanalysis). Several input differences at once (for instance, for truncated integral cryptanalysis) can be considered by choosing an initial vector of weight different than one in (2.7) (the entries must add up to one, of course).

Céline Blondeau, Benoît Gérard and Kaisa Nyberg [BGN12a] use a different approach to address the problem of large counter vectors. A key components in their generalisation are is the *partition function*, i.e. a map π from the set of all possible differences to a smaller set of values V, and counters are gathered for each $v \in V$. The counters are then computed as

$$D^{(N)}_f(\Delta, v) = \#\{x \mid \pi\big(f(x+\Delta) - f(x)\big) = v, \text{ for } N \text{ random } x\} \tag{2.8}$$

for both expected counters – i.e. where $f = E_s[K]$) – and for the sampled ones – i.e. where $f = T_s^{-1}[k^*] \circ E[K]$.

A simple partition function is $\pi(x) = 1$ if $x = \nabla$ and $\pi(x) = 0$ otherwise. By mapping, for instance, all differences with some fixed bits to a single value, truncated differentials are included in the model. Multiple distinct differentials are considered by mapping each differential to a different index, i.e. for possible output differences ∇_i, $1 \le i \le m$ we can define $\pi(x) = i$ if $x = \nabla_i$ and $\pi(x) = 0$ if $x \notin \{\nabla_1, \ldots, \nabla_m\}$.

This has the advantage that the counters vector

$$\mathscr{D}^{(N)}_{T_s^{-1}[k^*]\circ E[K]}(\Delta) = \left[D^{(N)}_{T_s^{-1}[k^*]\circ E}(\Delta, v)\right]_{v\in V}$$

can be much shorter, and thus it can be computed and stored also for larger ciphers than in the Albrecht-Leander generalisation – but at the same time the mapping π destroys the original information.

The scoring function essentially maps the observed empirical distribution to scores for the various key candidates, allowing the analyst to try those with a better score first. Scoring functions

can be based on the LLR, as in the Albrecht-Leander generalisation, or χ^2 statistics. Their definitions are recalled next.

Definition of the LLR and χ^2 scoring functions. Let $= [\theta_v]_{v \in V}$ be a uniform distribution vector (the vector has to be *weighted* according to the cardinalities of the preimages of the $v \in V$ w.r.t. the partition function π), and let $\mathbf{p}$ be the expected probability distribution vector $\mathbf{p} = [p_v]_{v \in V}$. Let $\mathbf{q}$ be the observed distribution for N samples. Then the LLR scoring function is defined as

$$\mathtt{LLR}(\mathbf{q}, \mathbf{p},) = N \sum_{v \in V} q_v \log \left(\frac{p_v}{\theta_v} \right)$$

and the χ^2 scoring function as

$$\chi^2(\mathbf{q},) = N \sum_{v \in V} \frac{(q_v - \theta_v)^2}{\theta_v} \ .$$

Depending on the bias with respect to ideal distributions, according to [BGN12a] a χ^2 based scoring function proves often better than a LLR based one. Using the LLR method the statistics can be computed on-the-fly, without having to store all the counters, but this is not possible with the χ^2 method, resulting in an increased memory cost. For both scoring functions, estimated for the smallest number of samples that are necessary for a successful determination of the distribution type are given and proved in [BGN12a].

2.1.10 Countermeasures

We now see how to make a cipher resistant to the various types of differential cryptanalysis.

2.1.10.1 Classic Differentials

revise and complete

The major method to build resistance against (non-impossible, non-improbable) differential cryptanalysis is by bounding the probability of the best characteristic (or differential) to be very low.

This is done by using S-boxes of low differential uniformity, but this is per se not sufficient. It is important that each output difference to influence as many S-boxes as possible downstream, i.e. they are diffused as quickly and thoroughly as possible – in a bricklayer construction this is done by using diffusion layers of higher branch number.

S-boxes that transform the types of differences, such as SAFER's logarithm or exponentiation S-boxes, as well as mixing different types of operations, such as bitwise XOR, integer addition, arithmetic in Galois fields, help not only against linear cryptanalysis but also against differential cryptanalysis. The use of different operations for both key mixing and combination with F-Function output is fundamental in the design of CAST (Section 3.7 on page 146).

However, all above considerations do not directly translate into a formal evaluation of resistance. A designed desires to bound the probability p of the best characteristic (or differential) such that $1/p$ is larger than the required complexity, or even larger than the size of the plaintext space (in which case even choosing the whole plaintext space is not sufficient for mounting an attack).

These bounds were formalized into various theories of provable security against differential cryptanalysis. Results about designing DES-like ciphers which are resistant against differential cryptanalysis exist [NK95].

A specially interesting theory for provable security against differential cryptanalysis (and also linear cryptanalysis) is Serge Vaudenay's theory of decorrelation, which we described in Subsectio 1.10.3 on page 63.

2.1.10.2 Impossible or Improbable Differentials

Against impossible differential cryptanalysis (and improbable differential cryptanalysis) the above approaches will of course not work – in fact they may even facilitate such attacks. In fact, we would need lower bounds on the probability of a differential characteristic, and there are no good techniques to ensure that.

Ruilin Li, Bing Sun and Chao Li in [LSL11] give sufficient but not necessary conditions for the existence of impossible differentials in SPNs. They remark that the linear transformation should be carefully designed in order to protect the cipher against impossible differential cryptanalysis – however their suggestions are limited to a check list of conditions to avoid.

Similarly, Charles Bouillaguet, Orr Dunkelman, Pierre-Alain Fouque and Gaëtan Leurent describe in [BDFL11] new generic instances of impossible differentials on various classes of Generalized Feistel Networks, and these should be taken into account while designing ciphers.

2.2 Linear Cryptanalysis

Linear cryptanalysis uses linear approximations of block ciphers to perform key recovery. It is a known-plaintext attack. It is not the first general cryptanalytic technique introduced to break block ciphers – that privilege going to differential cryptanalysis (section 2.1 on page 71).

Mitsuru Matsui and Atsuhiro Yamagishi introduced the technique to break the cipher FEAL in 1992 [MY92] and Matsui applied it to DES the following year [Mat93]. Similar ideas have been published before by Adi Shamir in 1985 [Sha85] and by Anne Tardy-Corfdir and Henri Gilbert in 1991 [TCG91].

2.2.1 Matsui's Algorithm 1

In its simplest form this cryptanalysis works as follows. Let A be a vector of length t over $\mathbb{F}_2$ and $\mathcal{I}$ be a subset of $[0..t-1]$ (t can be the block size n or the key length ℓ). The notation $\langle A, \mathcal{I} \rangle$ denotes the sum (parity) of the bits in A at the positions indexed by $\mathcal{I}$, i.e. the parity bit of the corresponding choice of bits. Let P, C and K denote plaintext, ciphertext and key, respectively. The aim of linear cryptanalysis is to find sets of indexes $\mathcal{I}_P \subseteq [0..63]$, $\mathcal{I}_C \subseteq [0..63]$, and $\mathcal{I}_K \subseteq [0..55]$, such that an equation

$$\langle P, \mathcal{I}_P \rangle \oplus \langle C, \mathcal{I}_C \rangle = \langle K, \mathcal{I}_K \rangle \tag{2.9}$$

holds with probability $1/2+\varepsilon_i$ for some $\varepsilon_i \neq 0$. In fact, since no cipher is ideal, *any* such equation will hold with probability different from $1/2$, so the goal is to find equations of type (2.9) that hold with probability p significantly different from $1/2$. For instance, for DES Matsui found an approximation that held with probability $1/2 - 2^{-24}$. Then, a simple algorithm based on

maximum likelihood is used to find one parity bit of the key: Suppose we have N random known plaintexts and Z is the number of these plaintexts for which the l.h.s. of (2.9) is 0. Then if $(Z - N/2)$ and ε_i have the same sign we conclude that $\langle K, \mathcal{I}_K \rangle = 0$, otherwise $\langle K, \mathcal{I}_K \rangle = 1$. Matsui computed that $N = O(|\varepsilon_i|^{-2})$ plaintexts are necessary to determine the correct parity.

This procedure is repeated with other linear approximations, obtaining additional key bit parities, until the number of unknown bits is low enough that they can be recovered by brute force.

In order to find linear approximations for the whole cipher one usually starts with linear approximations for single rounds, and combines them. Care must be taken to correctly multiply the probabilities. To this purpose the **Piling-up Lemma** is used: *Let X_i for $i = 1, 2, ..., n$ be independent variables taking values in $0, 1$, each with bias ε_i (i.e. the difference of the probability the value is 0 or 1 from $1/2$). Then the bias of the sum $X = \oplus_{i=1}^{n} X_i$ is given by $\varepsilon = 2^{n-1} \prod_{i=1}^{n} \varepsilon_i$.*

Thus, the approximation for the whole cipher is written as a chain of connected linear approximations, each spanning a small part of the cipher. Such a chain is called a **linear characteristic**. Assuming that the biases of these partial approximations are statistically independent and easy to compute, the total bias can be computed using the piling-up lemma. In practice, the biases of the partial approximations are not completely independent, so the actual bias can be larger or smaller. In order to better evaluate attack complexities, Kaisa Nyberg described the *linear hull* effect in [Nyb94]. A complex linear approximation for several rounds can be obtained as the product of *different chains*. Estimates of the likelihood of linear and differential cryptanalysis attacks under these expanded considerations can be found in [Sel08].

2.2.2 Matsui's Algorithm 2

Matsui further improves his approach in his Algorithm 2 by using a linear approximation for $r - 1$ rounds to break an r round cipher. In the case of DES, a 15 round approximation is used, and in place of (2.9) an equation

$$\langle P, \mathcal{I}_P \rangle \oplus \langle C, \mathcal{I}_C \rangle \oplus \langle F(C, k_{16}), \mathcal{I}_F \rangle = \langle K, \mathcal{I}_K \rangle \tag{2.10}$$

where F is the F-Function of the cipher, that accepts (half of) the ciphertext C and a round key as inputs, and it is just 32-bits wide, so that $\mathcal{I}_F \subseteq [0..31]$. Note that, if the sets $\mathcal{I}_F$ and $\mathcal{I}_C$ are constructed such that the selected bits of C are all XORed with the corresponding bits of $F(C, k_{16})$, then $\langle C, \mathcal{I}_C \rangle \oplus \langle F(C, k_{16}), \mathcal{I}_F \rangle$ is indeed a linear function of bits of the output of the penultimate round.

For wrong guesses of the round key, under the *wrong key randomisation hypothesis* the bias will be strongly reduced, hence the right round key and the right parity are guessed for the largest difference $|Z - N/2|$. The wrong key randomisation hypothesis essentially states that when analyzing a block cipher, partially decrypting/encrypting with a wrong key up to the boundary of the linear approximation, the adversary faces a randomly drawn permutation instead of the expected cipher structure with rounds peeled off. This assumption has been partially called into question by Andrey Bogdanov and Elmar Tischhauser [BT13] and found that significant biases arising from wrong keys are unduly underestimated.

Expressions more complex than (2.10) allow to use a linear approximation of the last $r-t$ rounds of the cipher, which may be useful in case it is possible to compute backwards the required bits of the last t rounds of the cipher without using all the entropy of the secret key.

2.2.3 Chosen-Plaintext Linear Cryptanalysis

Lars Knudsen and John Erik Mathiassen show that the efficiency of Matsui's linear cryptanalysis can be improved by use of chosen plaintexts [KM00a]. With respect to more recent improvements, this variant has worse complexity but as François Koeune et al. [KRS+02] remark, it is better suited for hardware implementations, leading - in 2002 - to a successful key recovery attack on the DES that takes less than 15 hours, using hardware roughly worth $ 3500.

2.2.4 Multiple Linear Approximations

Burton Kaliski and Matt Robshaw use first several linear approximations simultaneously to obtain more information [KR94a, KR94b]. Their approximations are of the same linear combination of the key bits, and the various approximations must be linearly independent. With several approximations

$$\langle P, \mathcal{I}_P^{(i)} \rangle \oplus \langle C, \mathcal{I}_C^{(i)} \rangle = \langle K, \mathcal{I}_K \rangle$$

with bias ε_i, for $1 \leqslant i \leqslant m$ they need $1 / \sum_{i=1}^{m} \varepsilon_i^2$ ciphertexts to recover a key parity. Alex Biryukov, Christophe De Cannière and Michaël Quisquater in [BCQ04] and improve the complexity to $1 / 4 \sum_{i=1}^{m} \varepsilon_i^2$ where also linearly dependent relations can be taken into account – de facto improving the complexity of the attack up to m bits, however the growth is not proven and has been only verified experimentally by Baudoin Collard, François-Xavier Standaert and Jean-Jacques Quisquater [CSQ08].

2.2.5 Multidimensional Linear Cryptanalysis

Miia Hermelin, Joo Yeon Cho, and Kaisa Nyberg [HCN08, CHN08] further improve the attacks based on multiple linear approximations in [BCQ04] and formally proved complexity bound with fully relaxed requirement for linear independence of the relations. In *multidimensional linear cryptanalysis* the space of the linear approximations has dimension m and can be parameterized by the elements of a m-dimensional vector space V. The complexity of the attack is $m / 2 \sum_{a \in V}^{m} \varepsilon(a)^2$. In [NWWL10] Phuong Ha Nguyen, Lei Wei, Huaxiong Wang and San Ling improve (practically, not asymptotically) these attacks. Miia Hermelin and Kaisa Nyberg further develop the theory of multidimensional linear cryptanalysis in [HN11].

2.2.6 Zero-Correlation Cryptanalysis

Zero correlation linear cryptanalysis [BR11b, BW12, BLNW12] is the counterpart of impossible differential cryptanalysis in the domain of linear cryptanalysis. It is a key recovery attack. Instead of designing the attack around a linear hull that presents a significant bias, a linear hull (a linear approximation) for some middle rounds of the cipher that presents no bias is chosen.

Let the cipher $E = E_K$ be written as $E_K = F_K \circ M_K \circ I_K$ where I, M, and F denote the compositions of the initial, middle and final rounds, respectively, of the cipher E.

Then, for each possible key guess K, plaintexts are encrypted for the rounds that precede the linear hull and the corresponding ciphertexts are decrypted through the last rounds, i.e, $I_K(P)$ and $F_K^{-1}(C)$ are computed, where (P, C) is a plaintext/ciphertext pair. Then the bias of the chosen linear approximation of the values of $I_K(P)$, $F_K^{-1}(C)$ and K is evaluated. If this bias is zero, then the key guess K was most likely correct. To efficiently evaluate the bias, several algorithmic improvements exist, the most important being the use of Fast Fourier Transforms [BGW+13].

Some cipher types present zero bias linear approximations essentially by design:

- Theorem 2 of [BR11b] shows that Rijndael data obfuscation (Section 3.20 on page 180) has a set of zero correlation linear approximations over 4 rounds (to be more precise, 3 full rounds appended by 1 incomplete rounds with MixColumns omitted). This leads to successful attacks on 6 rounds of AES-192 and AES-256 with time complexity $2^{188.4}$.
- Theorem 1 of [BR11b] states that Type-2 generalized Feistel networks with 4 branches (Section 1.3 on page 27) have zero correlation linear approximations over 9 rounds, if the underlying F-functions of the Feistel construction are invertible. A notable example of this fact is given by CLEFIA (Section 3.28 on page 201).

Since this is just a particular type of linear analysis, it can be combined with other improvements. For instance, in [BW12] multiple approximations of correlation zero are used to attack TEA and XTEA, leading to the best attack on TEA to date (cf. Section 3.12 on page 159). We have mentioned that for Matsui-type linear cryptanalysis the use of multiple approximations can be generalised multidimensional linear cryptanalysis, and the same is true for zero correlation cryptanalysis. In [BLNW12] Andrey Bogdanov, Gregor Leander, Kaisa Nyberg and Meiqin Wang show propose multidimenstional distinguishers by which they can show that integrals cryptanalysis can be seen a special case of zero correlation cryptanalysis and these concepts are applies to the best attack to date to CAST-256 without weak key assumptions (cf. Subsection 3.7.2 on page 147).

2.2.7 Nonlinear Approximations

A generalization using nonlinear approximations has been introduced by Lars Knudsen and Matt Robshaw in [KR96c]. The technique was applied to LOKI91, a forerunner to AES contest submission LOKI91 (Subsection 3.19.5 on page 178), and improved both effectiveness and complexity with respect to previous attacks to LOKI97.

Juan Tapiador, John Clark, and Julio Hernández Castro observe [ETCC07] that a problem concerning non-linear functions is how to identify them efficiently, given that the search space is superexponential in the number of variables, and tackle this problem by using heuristic search techniques, in particular Simulated Annealing. This is still an active area of research.

In [MC13b] nonlinear approximations are applied to Serpent, resulting in a new attack on 11-round Serpent with better data complexity than any other known-plaintext or chosen-plaintext attack to date.

2.2.8 Countermeasures

We now see how to make a cipher resistant to linear cryptanalysis.

2.2.8.1 Matsui-Type Linear Cryptanalysis

We have described in Section 1.9 on page 55 the conditions an S-box must satisfy in order to harden a cipher against linear cryptanalysis. This, in principle, works against Matsui's Algorithm 1.

Against Algorithm 2 type linear cryptanalysis one should make sure that the information provided by the key is used completely in as few rounds as possible, both at the beginning and at

the end of the cipher, in order to make it pointless to use linear approximations of the almost complete cipher. This applies not only to Feistel ciphers, but also to block ciphers when the key size is larger than the block size.

2.2.8.2 Zero-Correlation Cryptanalysis

It is clear that making a cipher *too* resistant against Matsui-Type linear cryptanalysis can open up the possibility of a zero-correlation attack. The best defence in this case is to make the outer rounds as dependent on as much of the entropy provided by the key bits as much as possible, in order to make the key space search part of the attack equivalent to brute force.

Indeed, most instances of this type of attack are on ciphers with a key larger than the block size, which means that unavoidably only part of the key entropy is used in the outermost rounds. For such ciphers, therefore, the design of the key schedule is of the paramount importance.

This also means that the defences against zero correlation cryptanalysis are usually subsumed in the defences against other types of attacks, for instance MITM attacks and their variants (see Section 2.4 on the next page), and slide attacks (Section 2.6.2 on page 107).

2.3 Differential-Linear Cryptanalysis

Susan Langford and Martin Hellman proposed the differential-linear attack in 1994 [LH94]. It is a mix of linear and differential cryptanalyses. The block cipher is first partitioned into two parts, i.e. $E = G \circ F$. The first part, F, is covered by a (truncated) differential characteristic $\Delta \rightarrow \nabla$, and the second part G by a linear characteristic. Then, a differential with probability p for F is used to obtain a difference in the middle of the cipher, i.e. for the inputs to G.

Suppose that for a pair of plaintexts P and P^* with $P \oplus P^* = \Delta$, the truncated differential relation $\big(F(P) \oplus F(P^*)\big) \wedge X = \nabla$ holds with probability p, where X is a vector used to select just the output bits needed by the truncated differential. Let $\lambda \rightarrow \lambda'$ be the linear characteristic for G, where $\langle T, \lambda \rangle \oplus \langle G(T), \lambda' \rangle = a$ holds with likelihood ϵ for a fixed $a = 0$ or 1. We also assume that the support of the vector λ is contained in the support of X (i.e. $\lambda \wedge X = \lambda$ but $\lambda \wedge \neg X = 0$). In fact, w.l.o.g. we can assume that $X = \lambda$.

Since we know that $\big(F(P) \oplus F(P^*)\big) \wedge X = \nabla$ with probability p, we have that

$$\langle F(P), \lambda \rangle \oplus \langle F(P^*), \lambda \rangle = \langle \nabla, \lambda \rangle =: \Lambda$$

with probability at least p. We also have that

$$\begin{aligned} \langle E(P), \lambda' \rangle &= \langle F(P), \lambda \rangle \oplus a \\ \langle E(P^*), \lambda' \rangle &= \langle F(P^*), \lambda \rangle \oplus a \end{aligned} \tag{2.11}$$

hold each with probability $\frac{1}{2} + \epsilon$ for a non-negligible bias $\epsilon > 0$. Summing these relations we obtain

$$\langle E(P) \oplus E(P^*), \lambda' \rangle = \Lambda \ . \tag{2.12}$$

Now, (2.12) holds if and only if the two relations (2.11) are simultaneously either both satisfied or both *not* satisfied. Therefore, (2.12) holds with probability $\left(\frac{1}{2} + \epsilon\right)^2 + \left(\frac{1}{2} - \epsilon\right)^2 = \frac{1}{2} + 2\epsilon^2$,

and we thus obtain the bias

$$\mathscr{P}[\langle E(P) \oplus E(P+\Delta), \lambda' \rangle = \Lambda] - \frac{p}{2} \geqslant 2p\epsilon^2 \ .$$

Note that this bias holds for a single bit valued function, but it can still be successfully used in *R attacks to distinguish right from wrong key guesses.

Initially, the differential characteristic was chosen with probability $p = 1$. With this restriction, Langford and Hellman were able to break 8 rounds of DES, and their attack on 8 round DES is still the best such attack.

The technique was used in the subsequent years to analyse IDEA (Section 3.6 on page 142): a reduced version of IDEA was analyzed by means of a differential-linear attack by Alex Biryukov and Eyal Kushilevitz in [BK98a], and differential-linear weak keys of the full IDEA were found by Philip Hawkes in [Haw98], along with a related-key differential-linear attack on reduced IDEA. Johan Borst, Lars Knudsen and Vincent Rijmen showed also that the extension of differential and linear cryptanalysis to ciphertext-only attacks works in the context of differential-linear cryptanalysis as well [BKR97].

Eli Biham, Orr Dunkelman and Nathan Keller in [BDK02a] present an extension of the Langford-Hellman cryptanalysis in which the linear probability induced by the differential characteristic is smaller than 1. They apply their results to 9 and 10 rounds of DES and to full COCONUT98. In particular, they provide the best attack, to the date the paper was published, to a 9 rounds version of the cipher, requiring $2^{15.8}$ chosen plaintexts and time $2^{29.2}$ reduced encryptions.

Recently, Zheng Yuan, Xian Li and Haixia Liu [YLL13] presented an *impossible differential-linear attack* on reduced-round CLEFIA. In this attack a distinguisher is constructed using an impossible differential with probability 1 for F, and a linear approximation with non-zero bias for G. A 13-round impossible differential-linear distinguisher is constructed in [YLL13] and it is used to mount an attack on 16-round CLEFIA with whitening keys with data complexity of $2^{115.52}$, recovering 96 bits of the key. However this attack is at the moment of purely theoretical interest since the time complexity is 2^{171}.

Zhiqiang Liu, Dawu Gu, Jing Zhang and Wei Li [LGZL09] further improve the technique by concatenating multiple linear characteristics to a single differential characteristic. Using their improvement they can break 9 rounds of DES faster than the 2002 Biham-Dunkelman-Keller attack, requiring $2^{14.1}$ chosen plaintexts and time $2^{25.93}$ reduced encryptions.

Jiqiang Lu [Lu12] improves the probabilities in the original setting $p = 1$, and attacks 13 rounds of DES with $2^{52.1}$ chosen plaintexts and time $2^{42.2}$.

At FSE 2014 Céline Blondeau, Gregor Leander, and Kaisa Nyberg [BLN14] remarked that none of the papers on differential-linear cryptanalysis so far used the links between differential and linear cryptanalysis from [CV94] and subsequent papers (as cited in [BLN14]) in their analysis. They then use this theory to give better estimated of the bias of the differential-linear characteristics. They do not present practical applications.

2.4 Meet-in-the-middle Attacks and Bicliques

Meet-in-the-middle (MITM) attacks have been used in the early times of block cipher design to prove that some constructions were weaker than initially thought – for instance in the case

of multiple encryption. This class of attacks has been subsequently quickly overshadowed by differential and linear cryptanalysis – until recently, when various improvements allowed to successfully attack newer ciphers, especially lightweight ones. In this section we shall review MITM attacks, their variants, and recent developments.

2.4.1 Meet-in-the-middle Attacks

In their simplest form, MITM attacks apply to the following scenario. The cipher $E = E_K$ can be written as the composition of two subciphers $E = E^2 \circ E^1$ (with inverses D, D^1, and D^2), and two complementary key bit sets $\mathscr{K}_1$ and $\mathscr{K}_2$ (i.e. $\mathscr{K}_1 \cap \mathscr{K}_2 = \emptyset$ and $\mathscr{K}_1 \cap \mathscr{K}_2 = \mathscr{K}$, the set of all key bits) exist with the following properties: $E^1 = E^1_{k_1}$, where k_1 is dependent only on the key bits $\mathscr{K}_1$ and $E^2 = E^2_{k_2}$, where k_2 is dependent only on the key bits $\mathscr{K}_2$. Thus, it is possible to view k_1 and k_2 as subkeys of the key K.

If some plaintext/ciphertext pairs (P, C) are available, the attack proceeds as follows:

1. For a plaintext P compute and store $E^1_{k_1}(P)$ for all subkeys (or partial keys) k_1 in a table V.
2. For the corresponding ciphertext C compute $v = D^2_{k_2}(C)$ for each partial key k_2.
3. If v is in the table V, use k_1 and k_2 to construct a key candidate.
4. Verify the candidate using other plaintext/ciphertext pairs.

The time complexity is given by $2^{\#\mathscr{K}_1} + 2^{\#\mathscr{K}_2} + 2^{\#\mathscr{K}-b}$ computations where b is the block size. Storage is $2^{\min\{\#\mathscr{K}_1, \#\mathscr{K}_2\}}$ (because if $\#\mathscr{K}_1 > \#\mathscr{K}_2$ one can store the values $D^2_{k_2}(C)$ in a table instead and search all the $E^1_{k_1}(P)$ in it).

In this form the attack was used by Whitfield Diffie and Martin Hellman [DH77] to prove that the security of double DES encryption with two independent 56 bit keys is lower than a naïve expectation of 112 bits, and in fact it is still 56 bits. For this reason Triple DES (cf. Subsection 3.2.8 on page 132) was developed, with the aim to offer 112 bits of security using 112 bits of key material. Ralph Merkle and Martin Hellman [MH81] proved that it can still be defeated in time and space $O(2^{56})$ by a chosen plaintext attack, which we shall describe this attack shortly (Subsection 2.4.4 on page 100). David Chaum and Jan-Hendrik Evertse [CE85] mount MITM attacks to up to 6 initial rounds or 7 rounds starting with the second of DES (improved attacks can be found in [DSP07]).

Research focused then again on hash functions. Some milestones are: second preimages for hash functions based on iterated block ciphers [LM92]; preimages for MD4, 63-step MD5, step reduced SHA-2 [AS08, AGM+09]; and preimages for full Tiger [GLRW10].

The pendulum swung again and new results on block ciphers were published, starting with attacks on KTANTAN [BR10, WRG+11]. The two main ideas in [BR10] are: the key bit sets $\mathscr{K}_1$ and $\mathscr{K}_2$ are not necessarily disjoint; and a set of key candidates is kept to be sieved later. We shall describe the first of these two ideas in Subsection 2.4.3 on page 98.

After some further development on hash function cryptanalysis (for instance, the results on SHA-2 and Skein-512 in [KRS12]) a new technique was introduced, the use of bicliques, that can be roughly described as a clever way of using precomputations to speed up MITM attacks that otherwise would not be better than brute force. Bicliques led to the first crypt-

analysis of full round AES, by Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger [BKR11a, BKR11b], quickly followed by the first cryptanalysis of full round IDEA together with impressive speed-ups with respect to previous reduced round versions of the same cipher [KLR12]. They will be described in Subsection 2.4.5 on page 101.

2.4.2 Space-Memory Tradeoffs

There are not many generic ways to break a cipher (or to invert a one way function). One obvious way is of course brute force (i.e. exhaustive search), with a memory footprint of $O(1)$ and time N. Another method consists in precomputing all possible pairs $(k, E_k(P))$ for all possible keys k for a fixed P, store them in a sorted (or hash) table according to the ciphertexts and then retrieve candidate keys by a table lookup. This is called a dictionary attack, and requires as much memory as the key space, which is in general impractical.

Martin Hellman [Hel80] shows how to dramatically reduce storage requirements of dictionary attacks. by using chains of values.

Let us consider first the case of a function $f : D \to D$, with $N = \#D$, and that we want to compute the preimages of several elements y. So we start from a point x_1 and compute $x_{i+1} = f(x_i)$ for $1 \leqslant i < N$, and store each s-th point, there $s = \lfloor \sqrt{N} \rfloor$, i.e. we store $x_1, x_{s+1}, x_{2s+1}, \ldots$. This is a once time only precomputation stage, or offline stage. In the online stage, for each point y we compute $f^i(y)$ until a stored x_{ks+1} is encountered. Then retrieve $x_{(k-1)s+1}$ from the table and apply f to it until y is calculated – which means that the preimage has been found.

Precomputation cost is N, storage is $S = \sqrt{N}$ and online time is also $T = \sqrt{N}$. Different tradeoffs are obtained storing more or less points, which makes the online stage faster or slower.

However this works only if f includes a single cycle. It can also be adapted to the case where f is a permutation. But, in general, we have to consider the cases where the function f is random, so there may be collisions between chains $f^i(x)$ for different values of x.

Suppose then we have a random function $f : D \to D$. Important examples are obtained from a block cipher $E = E_k(P)$ with key size of ℓ, with $N = 2^\ell$, and a block size of n, and a fixed point P. If $n = \ell$, then $f(x) = E_x(P)$. If $n > \ell$, we take a *reduction* function R that *maps* a ciphertext to a key (in the case of ciphers like the DES it indeed reduces the values to smaller ones) and put $f(x) = R(E_x(P))$. If $n < \ell$, then we define $f(x)$ as the concatenation of enough $E_x(P_1)$, $E_x(P_2)$, ..., so that the total length of the texts is at least as big as ℓ, possibly *reduced* by a suitable reduction function if ℓ is not an exact multiple of n.

For the offline stage, for any *starting point* x_1 define a sequence as follows

$$x_{i+1} = R(E_{x_i}(P)) = f(x_i) = f^i(x_1) \ .$$

Now, pick a parameter t, and random starting points (i.e. key values) $x^{(1)}$, $x^{(2)}$, ... $x^{(t)}$. For each starting point $x^{(j)}$, compute the chain of length m starting at $x^{(j)}$, i.e. $x_1^{(j)}$, $x_2^{(j)}$, ..., $x_m^{(j)}$, *and store only the starting and ending points*, i.e. keep a table of the pairs $\left(SP_j, EP_j\right) := \left(x_1^{(j)}, x_t^{(j)}\right)$. Storage is then m (ignoring small constants) and time to create this structure is mt. The pairs are sorted (or hashed) on the final points.

In the online stage the attacker uses the target encryption oracle to compute $E_k(P)$ under an unknown key, then starts with $z_1 = R(E_k(P))$ and computes $z_{i+1} = f(z_i) = f^i(z_1)$, which is of

course $R(E_{z_i}(P))$, for at most t steps. Hopefully (i.e. for a high percentage of all the keys) at some point the attacker will find a value that is equal to one of the end points. Now the attacker will have to start from the corresponding *starting point* and repeat the computation, to see if $R(z_1)$ is actually in that chain, in which case the previous value in the chain is a candidate key.

In theory, in a perfect situation where no two chains merge for the given starting points, and with no false positives in the online phase, one could choose $m = t = N^{1/2}$ and have an attack that needs $N^{1/2}$ storage and $N^{1/2}$ online time. This is exactly what happens if f is a permutation. Also, this contradicts the behaviour of a random function.

In order to reduce false positives, r different *tweaks* f_i of the original function f are used, for instance different reduction functions are used, and the tables of starting end ending points are computed for each reduction function. Hence the memory usage becomes mr and the time to generate the offline structure mtr. The online time to find the key is then tr. If $m = t = r = N^{1/3}$, the storage for the offline table is $N^{2/3}$, and the online time is also $N^{2/3}$. In this situation it can be shown that the success probability is 0.55.

Rivest [Den82], page 100, suggested to use distinguished points (for instance key whose internal representation begins with a string of ten zero bits, say) and stopping chain computation at when those points have been found – for all chains, both in the offline and online stage.

Paul van Oorschot and Michael Wiener [vOW96] formulate MITM attacks as collision search problems. Given two functions f_0 and f_1 two values a and b are sought such that $f_0(a) = f_1(b)$. The domains are then joined and a function $f(a, i) : D \times \{0, 1\} \rightarrow R$ is created such that $f(a\|i) = f_i(a)$ for all $a \in D$ and $i \in \{0, 1\}$. Hence the collision which is sought is $f(a\|0) = f(b\|1)$. With likelihood $1/2$ a collision $f(a\|i) = f(b\|j)$ will have $i = j$, in the other half of cases $i \neq j$ and therefore a successful MITM candidate will have been identified. They also use distinguished points to further reduce the memory storage.

The tradeoff techniques just presented can be often combined with the methods described in the following subsections – however in each case a specific analysis has to be performed, and it cannot be assumed that they automatically lead to improvements.

2.4.2.1 Further Tradeoffs

A further important idea to reduce collisions among chains is the use of rainbow tables. A rainbow table is a table where a different reduction function is used for each column of the table, so a chain looks like as follow:

$$z_1 \xrightarrow{f_1} z_2 \xrightarrow{f_2} z_3 \xrightarrow{f_3} z_4 \xrightarrow{f_4} \dots \xrightarrow{f_{m-1}} z_m \ . \tag{2.13}$$

Thus two different chains can merge only if they have the same key at the same position of the chain. This makes it possible to generate much larger tables. Actually, a rainbow table acts almost as if each column of the table was a separate single classic table. Indeed, collisions within a classic table (or a column of a rainbow table) lead to merges whereas collisions between different classic tables (or different columns of a rainbow table) do not lead to a merge. This technique has been prove in password recovery, less so in other cryptanalitic contexts.

Amos Fiat and Moni Naor [FN99] provide rigorous time/space tradeoffs $TS^2 = N^3 q(f)$ for inverting any function f, where $q(f)$ is the probability that two random elements have the same

image under f. A more general tradeoff $TS^3 = N^3$ is also given in [FN99].

2.4.3 Partial Matching (Match- and Sieve-in-the-Middle)

Sometimes it may be impossible to partition the cipher in a useful way into chunks that cover it *completely* and depend only on disjoint sets of key bits.

To overcome these hurdles, in [BR10] overlapping key bit subsets are used, as well as a technique called partial matching, or match-in-the-middle.

The cipher is factored into three chunks $E = E^2 \circ M \circ E^1$ where M is a middle chunk that allows match m bits of information, with $0 < m \leqslant b$, b being the block size (whence the technique is also referred as "match-in-the-middle").

Instead of looking for state collisions in the middle of the cipher, both forward and backward computations, i.e. the computations through E_1 and D_2 stop at the beginning and end of the middle function M, and a (partial) collision if found by computing "through" M.

Usually M just allows only a few bits to be directly compared, or through m linear relations between input and output involving at least one input and not output bit, as in Takanori Isobe and Kyoji Shibutani's analysis of XTEA, LED and Piccolo [IS12] or in the analysis of full IDEA by Dmitry Khovratovich, Gaëtan Leurent and Christian Rechberger in [KLR12].

The chunks E^1, resp. E^2 depend only on the key bits $\mathscr{K}_1$, resp. $\mathscr{K}_2$, where $\mathscr{K}_1 \cup \mathscr{K}_2$ is the set $\mathscr{K}$ of all key bits, and we set $\mathscr{A}_0 := \mathscr{K}_1 \cap \mathscr{K}_2$. If $\mathscr{A}_0 \neq \emptyset$ then we are in the situation called the *3-Subset MITM attack* by Andrey Bogdanov and Christian Rechberger.

Anne Canteaut, María Naya-Plasencia and Bastien Vayssière describe in [CNPV13a] a generalisation of the Match-in-the-Middle and 3-Subset MITM attacks, which they name *sieve-in-the-middle*. In some cases, it allows to extend MITM attacks to a higher number of rounds than previously possible.

As in Match-in-the-Middle attacks, forward and backward computations stop at the beginning and end of the middle function M. Instead of computing "through" M, in [CNPV13a] a particular middle function S is computed instead of M. The function is denoted by the letter S because it is often an S-box, but it can also be a superBox (as in the case of the AES) or an arbitrary component of the cipher. This function can of course be M itself, but one of the reasons it can be restricted is that it can be used to pre-filter the possible key candidates, which are then verified later. In other words, S serves to *efficiently* discard – i.e. sieve – all key candidates which do not correspond to a valid transition through it. Then, a merging step collects the keys that are not discarded into a single list, and these are then tested until the correct key is found.

The target cipher is then, as already mentioned, written as $E = E^2 \circ M \circ E^1$ where for simplicity E, E_1, M, E_2 are represented as functions $\mathbb{F}_2^n \to \mathbb{F}_2^n$.

Also, there are functions ϕ_f (the lower case f stands for "forward") and ϕ_b (the lower case b stands for "backward") and S, together with subsets I and J of $[0..n-1]$ of bit positions, with following properties:

- $u = \phi_f(x)$ is obtained by restricting $E_1(x)$ to just the bits in the positions I;
  The function ϕ_f depends only on a subset K_1 of the bits of the secret key;
- $v = \phi_b(y)$ is obtained by restricting $D_2(y)$ to just the bits in the positions J;

The function ϕ_b depends only on a subset K_2 of the bits of the secret key;

- The function S maps u to v, i.e. the bits in positions I of $E_1(x)$ to the bits in positions J of $D_2(y)$, for a valid $y = E(x)$.

For instance, in a SPN the function ϕ_f can be taken as the function that maps the plaintext to the input to a specific S-box S in the i-th round, and ϕ_b as the function that partially decrypts the ciphertext until the output of the same S-box.

Put $\mathscr{A}_0 := \mathscr{K}_1 \cap \mathscr{K}_2$. It can be $\mathscr{A}_0 = \emptyset$ or $\neq \emptyset$, the latter case being called the 3-Subset Attack. and $\mathscr{A}_i := \mathscr{K}_i \setminus \mathscr{A}_0$ for $i = 1, 2$.

Let (P, C) be a plaintext/ciphertext pair.

The following steps are performed for all partial keys K_0 defined over $\mathscr{A}_0$ (if $\mathscr{A}_0 = \emptyset$ there is no need to loop over the keys K_0):

1. Start with two empty lists $\mathscr{L}_f$ and $\mathscr{L}_f$.
2. For all partial keys K_1 defined on $\mathscr{A}_1$ compute $u = \phi_f(P)$ and add (u, K_1) to $\mathscr{L}_f$.
3. For all partial keys K_2 defined on $\mathscr{A}_2$ compute $v = \phi_b(C)$ and add (v, K_2) to $\mathscr{L}_b$.
4. Merge the lists $\mathscr{L}_f$ and $\mathscr{L}_b$ to obtain a list $\mathscr{L}_{sol}$ formed by candidate partial key pairs (K_1, K_2) for which an $x \in \mathbb{F}_2^n$ exists such that $u = (x_i, i \in I)$ and $v = (S(x)_j, j \in J)$.

The lists $\mathscr{L}_{sol}$ for each partial key K_0 are joined together and the keys contained therein are tested. This step can also be interleaved with the merging of $\mathscr{L}_f$ and $\mathscr{L}_b$.

Obviously, the whole secret key can be recovered only if the key length does not exceed the blocksize. Otherwise, $2^{\ell-n}$ possible keys will be returned in average where ℓ is the key length – and this will be a lower bound for the memory usage.

We say that any pair (I, J) of sets of size (m, p) with $m + p > n$ is a sieve for S with sieving probability $\pi_{I,J} \leqslant 2^{n-(m+p)}$. The running time is

$$2^{\#\mathscr{A}_0}\left(2^{\#\mathscr{A}_1}c_f + 2^{\#\mathscr{A}_2}c_b + C_{merge}\right) + \pi_{I,J}2^{\ell}C_E$$

where c_f and c_b are the costs of evaluating ϕ_f and ϕ_b, C_{merge} is that of merging the lists $\mathscr{L}_f$ and $\mathscr{L}_b$, and C_E is that of one encryption. This running time is thus very similar to that of match-in-the-middle attacks, but the strategy can win if a suitable choice of S and a good merging strategy can be found. Also, it is obviously crucial that verifying whether S permits a valid transition be significantly faster than direct candidate key verification.

Merging can be done in several different ways, as shown in [CNPV13a]. One of the most powerful tools is to partition one of the lists $\mathscr{L}_f$, $\mathscr{L}_b$ into smaller sublists and use precomputations to speed up the matching. Matching can also be done using a recursive approach (what Canteaut et al. call *gradual matching*) and in parallel with a negligible memory footprint adapting techniques from [DDKS12] and [NP11].

The sieve-in-the-middle method can be combined in a natural way with some of the techniques presented later, such as Splice-and-Cut and Bicliques.

2.4.4 Splice-and-Cut: The Merkle-Hellman Attack

We describe now Ralph Merkle and Martin Hellman's key recovery attack [MH81]. It is a chosen plaintext MITM attack. It applies to any cipher $E = E_{k_1,k_2}$ constructed from a cipher S in the same way as the Triple DES, i.e. $E_{k_1,k_2} = S_{k_1} \circ S_{k_2}^{-1} \circ S_{k_1}$ where k_1 and k_2 are two subkeys.

However, it does not hurt to consider the following more general situation, that covers also the case of the 2-key Triple DES construction: the cipher E can be written as the functional composition of three *chunks* $E_{k_1,k_2} = E^3_{k_1} \circ E^2_{k_2} \circ E^1_{k_1}$ where the key k_1 of E_1 and E_3 only depends on key bits $\mathscr{K}_1$ and the key k_2 of E^2 depends on key bits $\mathscr{K}_2$. Let D, resp. D^i $(i = 1, 2, 3)$ denote the decryption functions of E, E^i. Also let n be the block size of the cipher.

Then, for any plaintext/ciphertext pair (P, C) there are two intermediate values u, v such that

$$P \xrightarrow[E^1]{k_1} u \xrightarrow[E^2]{k_2} v \xrightarrow[E^3]{k_1} C \quad . \qquad (2.14)$$

$$P \xrightarrow{\mathscr{O}} C$$

The attack assumes the adversary has access to an encryption oracle $\mathscr{O}$ for E, for unknown, fixed keys k_1 and k_2. It works as follows:

1. Fix a value $\bar{u}$, for instance $\bar{u} = 0$.
2. Compute and store a table V of the values $v_j = E^2(\bar{u})$ for all keys j on the bits $\mathscr{K}_2$.
3. For each key i on the bits $\mathscr{K}_1$, perform the following steps:
   (a) compute $P_i = D^1_i(\bar{u})$, $C_i = \mathscr{O}(P_i)$, and $v'_i = D^3(C_i)$;
   (b) determine if the value v'_i is in the table V, i.e. if there exists a key j such that $v'_i = v_j$;
   (c) if a match is found, then (i, j) is a candidate for (k_1, k_2);
   verify it on one or more additional plaintext/ciphertext pairs.

To understand why (i, j) with $v'_i = v_j$ is a candidate for (k_1, k_2) observe that the collision

$$\bar{u} \xrightarrow[D^1]{i} P_i \xrightarrow[\mathscr{O}]{} C_i \xrightarrow[D^3]{i} \boxed{v'_i = v_j} \xleftarrow[E^2]{j} \bar{u}$$

implies that the following diagram

$$P_i \xrightarrow[E^1]{i} \bar{u} \xrightarrow[E^2]{j} v_j = v'_i \xrightarrow[E^3]{i} C_i \qquad P_i \xrightarrow{\mathscr{O}} C_i$$

commutes, i.e. at least for the pair (P_i, C_i) the cipher $E_{i,j}$ behaves like E_{k_1,k_2}.

Storage is required only for the table V, i.e. $2^{\#\mathscr{K}_2}$ entries. The running time is $O(2^{\#\mathscr{K}_1} + 2^{\#\mathscr{K}_2})$. The exact complexity depends on the likelihood that a match at step 3(c) is actually a match

for the keys. Generally speaking a match should be confirmed on at least one more plaintext/ciphertext pair than $\lfloor \#\mathcal{K}_1/n \rfloor$. If $\mathcal{K}_2 > \mathcal{K}_1$ the storage requirement can be reduced by storing the values v'_i instead and matching the v_j against them.

The first such MITM attack to a block cipher is found in [WRG+11].

This attack was first called "splice-and-cut" (as applied to the compression function of a hash function) by Kazumaro Aoki and Yu Sasaki in [AS08], where it is described thusly:

> *We consider the first and last steps of the attack target as consecutive steps. Then, we divide the attack target into two chunks of steps so that each chunk includes at least one message word that is independent from the other chunk.*

The last and first step of the target need to be "spliced" together, i.e. the output of the last step must be transformed into a useful input to the first step. In the context of Davies–Meyer hash functions this is often done by choosing the input message to influence the feedback addition. In the context of block ciphers we have seen that the encryption oracle serves this purpose.

Initial splice-and-cut attacks on hash functions and block ciphers were all chosen message (ciphertext) attacks. Paul van Oorschot and Michael Wiener introduced in [vOW90] a variant of the attack against the 2-key Triple DES construction that relies on a table of known plaintext/ciphertext pairs (P, C) and waits until a collision between the computed $P_i = D_i^1(\bar{u})$ and a known plaintext P is found. If no match occurs, a new $\bar{u}$ is chosen: how often $\bar{u}$ is expected to be changed depends on the size of the available plaintext/ciphertext pairs pool. This permits a tradeoff between memory and running time.

An example of a cipher that can be easily broken by splice-and-cut (both CPA and KPA) attacks is MAGENTA: (cf. Subsection 3.19.6 on page 179). In fact, the attacks described in [BBF+99] are nothing but splice-and-cut attacks *ante litteram*, as they predate the Aoki and Sasaki description in the case of single encryption.

Recently Itai Dinur et al. [DDKS13a] showed that splice-and-cut techniques can be performed as known plaintext attacks also in the single encryption scenario. They demonstrated this on step reduced LED-64 – their attack (the first known plaintext attack on LED-64) is a *differential* MITM attack, a fact that is noteworthy on its own.

Partial matching and overlapping key bit sets (Subsection 2.4.3 on page 98) can be combined with the splice-and-cut technique in a straightforward way.

2.4.5 Bicliques

Bicliques are a recent cryptanalytic tool [KRS12]. They can be used when there is a part of the cipher that cannot be separated in a useful way into two sections that depend only on distinct sets of key bits. They are a formalisation of the "initial structure" technique introduced by Kazumaro Aoki and Yu Sasaki in [AS09].

2.4.5.1 Definition

A biclique is a pair of sets of internal states, which are (usually) constructed either in the first or in the last rounds of a cipher, and mapped between each other by specifically chosen keys.

Let E be a cipher and f be a subcipher of E, i.e. there are two ciphers E^1 and E^2 such that

$E = E^2 \circ f \circ E^1$. A *biclique* is given by a pair of sets $\{S_i\}$ and $\{T_j\}$, where the $\{S_i\}$ are inputs to f and $\{T_j\}$ outputs, and a set of keys $\{N[i,j]\}$ such that

$$S_i \xmapsto[f]{N[i,j]} T_j$$

and for each pair (i,j) there is exactly one key $N[i,j]$ that maps S_i to T_j. Thus, if we consider the $\{S_i\}$ and $\{T_j\}$ as the vertices of a graph, and the $N[i,j]$ as edges connecting S_i to T_j, we obtain a complete bipartite graph or biclique.

If both $\{S_i\}$ and $\{T_j\}$ contain 2^d states, the biclique is said to have dimension d.

2.4.5.2 Using Bicliques

Suppose we can write the cipher E as the functional composition of four *chunks*

$$E_{k_1,k_2} = E^3_{k_1} \circ f_{k_1,k_2} \circ E^2_{k_2} \circ E^1_{k_1}$$

where the key k_1 of E_1 and E_3 only depends on the bits in $\mathcal{K}_1$ and the key k_2 of E^2 on the bits in $\mathcal{K}_2$. For any plaintext/ciphertext pair (P, C) intermediate values S, T, and v exist such that

$$P \xrightarrow[E^1]{k_1} S \xrightarrow[f]{k_1,k_2} T \xrightarrow[E^2]{k_2} v \xrightarrow[E^3]{k_1} C \quad . \qquad \mathcal{O} \tag{2.15}$$

The goal is to find a collision at v. The idea here is to treat the "$S \xrightarrow[f]{k_1,k_2} T$" part of (2.15) like the "u" in (2.14). Hence, we do not start with just any value S or T, but with the elements in the biclique. In other words we compare

$$v'_i = D^3_i(\mathcal{O}(D^1_i(S_i))) \quad \text{and} \quad v_j = E^2_j(T_j)$$

where S_i and T_j are biclique vertices connected by $N[i,j]$.

Now, to construct a biclique for *all* subkeys i and j on key bit sets $\mathcal{K}_1$ and $\mathcal{K}_2$ respectively would require memory comparable to the whole key space, so this is not a viable attack. Therefore the context is necessarily that of relatively small key spaces $\mathcal{K}_1$, $\mathcal{K}_2$, say of dimension d, where the other key bits are fixed for the whole cipher, say, $\#\mathcal{K}_3 = \#\mathcal{K} - 2d$ bits where $\mathcal{K}$ is the set of all key bits and $\mathcal{K}_3 := \mathcal{K} \setminus (\mathcal{K}_1 \cup \mathcal{K}_2)$. A biclique has to be created for each new choice of the bits in $\mathcal{K}_3$ and the cost of the attack is $2^{\#\mathcal{K}-2d}(c \cdot 2^d + C_{\text{biclique}})$. Since c is about twice the cost of a single encryption, it is easy to see that for small values of d and without a clever way of building bicliques, the attack cannot be faster than brute force, without even taking into account the cost of multiple verification of candidates or partial matching.

A remark about the existence of bicliques is due. The whole cipher can be seen as a biclique of dimension $\#\mathcal{K}/2$. Also, simple bicliques of dimension zero (i.e. mapping a single state to another state under a fixed key) always exist. One may be naïvely tempted to believe that bicliques for any dimension always exist. This is sadly not the case: proving the existence of bicliques of useful dimension for all choices of a fixed set of key bits, and *actually constructing them*, is

a non trivial task. In the following subsubsection we shall describe two biclique construction methods.

2.4.5.3 Biclique Construction

There are several approaches to the construction of bicliques. An obvious one can be applied where some key bit subsets are used in an alternating fashion in a cipher. For instance suppose that the function f itself can be written as $f = f_2 \circ f_1$ where f_1, resp. f_2 is *not* affected by some of the bits in $\mathscr{K}_1$, resp. $\mathscr{K}_2$.

Then the situation can be roughly described by the following diagram

$$P \xrightarrow[E^1]{k_1} S \xrightarrow[f^1]{k_2'} w \xrightarrow[f^2]{k_1'} T \xrightarrow[E^2]{k_2} v \xrightarrow[E^3]{k_1} C \ ,. \qquad \mathcal{O} \tag{2.16}$$

where k_2', resp. k_1' is *not* contained in $\mathscr{K}_1$, resp. $\mathscr{K}_2$, and therefore it can be assumed w.l.o.g. that k_i' is defined on (a subset of) $\mathscr{K}_i$, for $i = 1, 2$. Then a biclique can be constructed with keys defined over these key bits.

By adopting a differential cryptanalysis approach, bicliques can also be constructed from two sets of *related key* differential trails

$$\left\{ 0 \xmapsto[f]{\Delta_i^k} \Delta_i \mid \forall i \right\} \qquad \text{and} \qquad \left\{ \nabla_j \xmapsto[f]{\nabla_j^k} 0 \mid \forall j \right\} .$$

A differential in the first set maps input difference 0 to an output difference Δ_i under key difference Δ_i^k, whereas a differential in the second set maps input difference ∇_j to an output difference 0 under key difference ∇_j^k. Now, suppose that the trails of Δ_i-differentials do not share active nonlinear components (such as active S-boxes in AES) with the trails of ∇_j-differentials – then if we combine a differential from the first set with a differential of the second set the only effects that are combined are combined *linearly*, hence

$$\nabla_j \xmapsto[f]{\Delta_i^k \oplus \nabla_j^k} \Delta_i$$

holds for all i, j. This is the same principle used in the boomerang attack [Wag99] (cf. Subsection 2.1.6 on page 83), at the root of the concept of the S-box switch [BK09] and sandwich attack [DKS10b].

Finally, if we have two states S_0 and T_0 with a key $N[0,0]$ that maps S_0 to T_0, we see that following states and keys

$$\begin{aligned} S_i &:= S_0 + \nabla_i \\ T_j &:= T_0 + \Delta_j \\ N[i,j] &:= N[0,0] \oplus \Delta_i^k \oplus \nabla_j^k \end{aligned}$$

satisfy the biclique property

$$S_i \xrightarrow[f]{N[i,j]} T_j \ .$$

Essentially all techniques that can be used to construct differential trails can be used to construct bicliques, for instance exploiting neutral bits (or neutral variables) [BC04], as used by Dmitry Khovratovich, Gaëtan Leurent, and Christian Rechberger [KLR12] to efficiently construct bicliques for IDEA.

2.4.6 The Parallel-Cut MITM Attack

Ivica Nikolic, Lei Wang and Shuang Wu [NWW13b] present an interesting twist on the MITM attack idea. Consider an n bit cipher

$$E = \rho_r \circ \rho_{r-1} \circ \cdots \circ \rho_2 \circ \rho_1$$

where, $P = P_1$ is the plaintext, $P_{i+1} = \rho_i(P_i)$ and P_{r+1} is the ciphertext, and the index i denoted dependence on the i-th expanded key $k_i = \psi_i(K)$, as usual. Then, each state P_i is split (possibly upon relabeling of the bit indexes) into left and right halves of $n/2$ bit each as $P_i = L_i \| R_i$ and the cipher itself is also split in *parallel* with its encryption path as two subciphers

$$E^R = \rho_r^R \circ \rho_{r-1}^R \circ \cdots \circ \rho_2^R \circ \rho_1^R$$
$$E^L = \rho_r^L \circ \rho_{r-1}^L \circ \cdots \circ \rho_2^L \circ \rho_1^L$$

that depend on each other as follows:

$$L_{i+1} \| R_{i+1} = \rho_i^L(L_i, R_i) \| \rho_i^R(L_i, R_i) \ .$$

The key observation here is that if the cipher is such that $\rho_i^L(L_i, R_i)$ depends on only a very few bits of R_i and $\rho_i^R(L_i, R_i)$ on only a few bits of L_i, then we can just *guess* these few bits, just as if they were unknown bits, similarly to the key bits, but in addition to them. Then, for each choice of these guessed bits we mount two separate MITM attacks on the ciphers E^R and E^L and we intersect the resulting candidate key sets. Note that the two attacks can both be accelerated by checking if some of the first steps of the two half encryptions match.

The attack works best when also the bits of the master key are at least partially partitioned between the two cipher halves, in other words we can also write $K = K^L \| K^R$ and $k_i = k_i^L \| k_i^R$ where the left halves k_i^L depend on K^L and just a few bits of K^R, and the right halves k_i^R depend on K^R and just a few bits of K^L.

KLEIN is such a cipher and Nikolic et al. have been able to successfully attack reduced round versions of the cipher (cf. Subsection 3.37.1 on page 224 for information about the cipher and the attacks).

Several refinements are described in [NWW13b] as well, such as the use of partial matching and time-memory tradeoffs. The main advantage of the attack is the data requirement - it is in many cases absolutely minimal, even one or two plaintexts/ciphertext pairs can suffice against KLEIN.

2.4.7 Countermeasures

To prevent MITM attacks and their variants a good strategy is to make sure that all key bits are used "everywhere" in the cipher. Especially for ciphers where the key size is larger than the block size, or for Feistel designs, this calls for at least a non trivial key schedule. Note that an alternating key schedule may not be sufficient: see Subsection 2.4.5.3 on page 103, and in particular the situation depicted in formula (2.16).

Furthermore, it is clear that good resistance against differential cryptanalysis additionally makes biclique construction difficult "for free."

2.5 Weak or Equivalent Keys

A weak key for a specific cipher is a key which makes the cipher weaker when used. Weak keys usually represent a very small fraction of the whole key space, hence the likelihood to use one at random is often negligible and it is unlikely that this causes problems.

However, it is a desirable design goal to have no weak keys. In some contexts weak keys are a problem when the adversary has some control over what keys are used, such as when a block cipher is used in a mode of operation intended to construct a secure cryptographic hash function (e.g. Davies-Meyer).

It is known that DES has a few weak keys. The design of MARS (Section 3.16 on page 167) had a problematic key schedule, with several checks added to avoid weak keys, slowing down the cipher considerably in a side channel resistant implementation.

There is no general rule for designing a cipher without weak keys. The key schedule plays here an important role.

A similar consideration applies to equivalent keys. Some ciphers, such as TEA (Section 3.12 on page 159) have sets of equivalent keys for each key. This usually only reduces the security minimally when the cipher is used for encryption (in the case of DES the sets of equivalent keys are called semi-weak keys), but this becomes a matter for concern if the block cipher is used to construct a hash function in a (modified) Davies-Meyer mode, as simple bit changes can produce different messages with the same hash.

We note that also certain types of weakness against related-key attacks may make a block cipher unsuitable to construct a hash function.

2.6 Related Key and Key Related Attacks

A related-key attack is any form of cryptanalysis where the attacker can observe the operation of a cipher under several different keys whose values are initially unknown, but where some mathematical relationship connecting the keys is known to the attacker. For example, the attacker might know that the last 80 bits of the keys are always the same, even though he doesn't know, at first, what the bits are. This is not an unrealistic model – for instance, the DES complementation property (Subsection 3.2.2 on page 130)

$$E_K(P) = C \iff E_{\overline{K}}(\overline{P}) = \overline{C}$$

(where $\overline{x}$ is the bitwise complement of x) is a related-key weakness that can speed-up brute force attacks by a factor of 2, and NewDES (Subsection 3.3.2 on page 134) possesses a similar property that can accelerate attacks by a factor of 2^8.

Also, glitches in random number protocols, or weaknesses in protocols can cause related keys to happen. On the other hand, in some protocols, the conditions for the desired key relations cannot occur – at least not without hardware glitching. The KASUMI block cipher, for instance, was broken in 2010 by Orr Dunkelman, Nathan Keller, and Adi Shamir, under a related key assumption [DKS10b], but the attack is not effective in the context of 3G protocols.

The situation is the opposite in the case of the Wired Equivalent Privacy (WEP) protocols for WiFi wireless networks. The WEP protocol has a fixed WEP key and its concatenation with a 24-bit initialization vector is used as the key for the RC4 encryption algorithm (a stream cipher). It is essential that the same key never be used twice with a stream cipher, however, the 24-bit space for the IV only allows 2^{24} possibilities and by the birthday paradox, it is highly likely that for every 4096 packets, two will share the same IV and hence the same RC4 key – at least these two packets can thus be attacked. This is not sufficient to recovery the key, but since RC4 has vast classes of weak keys, if one of these is found then the RC4 key can be recovered – hence also the WEP key itself. Such an attack was publicly shown in 2005 by agents from the U.S. Federal Bureau of Investigation.

2.6.1 Biham's Sliding Related Key Attack

Another important class of related key attacks has been discovered by Eli Biham in [Bih93, Bih94b], and it is related to the slide attacks described in the next subsection.

Assume that the cipher E is an iterative product cipher, based on a keyed round function $f = f_k$, which is susceptible to known plaintext attacks. Suppose also that for a key K we can force the use of a key K' with the property that the key schedule of K, i.e. $(k_i)_{1 \leqslant i \leqslant r}$ significantly overlaps with the key schedule of $\hat{K}$, i.e. $(\hat{k}_i)_{1 \leqslant i \leqslant r}$. For simplicity let us consider the case where $k_{i+1} = \hat{k}_i$ for $2 \leqslant i \leqslant r-1$. The situation is depicted in Figure 2.3.

Figure 2.3: Biham's "Sliding" Related Key Attack

$$\begin{array}{ccccccccccccc}
P_0 = P & \xrightarrow[f]{k_1} & P_1 & \xrightarrow[f]{k_2} & P_2 & \xrightarrow[f]{k_3} & \cdots & \xrightarrow[f]{k_{r-1}} & P_{r-1} & \xrightarrow[f]{k_r} & P_r = C \\
\| & & \| & & \| & & & & \| & & \| \\
\hat{P}_0 = \hat{P} & \xrightarrow[f]{\hat{k}_1 = k_2} & \hat{P}_1 & \xrightarrow[f]{\hat{k}_2 = k_3} & & & \cdots & \xrightarrow[f]{\hat{k}_{r-2} = k_{r-1}} & \hat{P}_{r-2} & \xrightarrow[f]{\hat{k}_{r-1} = k_r} & \hat{P}_{r-1} & \xrightarrow[f]{\hat{k}_r} & \hat{P}_r = \hat{C}
\end{array}$$

Let $\hat{P} = f_{k_1}(P)$. Due to the equality between the functions, P and $\hat{P}$ share $r-1$ rounds of the encryption. Thus, $\hat{C} = f_{\hat{k}_r}(C)$.

If we have such a pair of plaintext/ciphertext pairs (P, C) and $(\hat{P}, \hat{C})$, we can exploit the vulnerability of the round function f to extract some (partial) information about k_1 and $\hat{k}_r$.

If the cipher has n bit blocks and keys, we proceed as follows:

1. Ask for the encryption of $2^{n/2}$ plaintexts $P^{(i)}$ under K. Let $C^{(i)} = E_K(P^{(i)})$.
2. Ask for the encryption of $2^{n/2}$ plaintexts $\hat{P}^{(j)}$ under $\hat{K}$. Let $\hat{C}^{(j)} = E_{\hat{K}}(\hat{P}^{(j)})$.

3. By the birthday paradox there is an index pair (i, j) such that $P^{(i)}$ is encrypted under one round to $\hat{P}^{(j)}$, i.e. $\hat{P}^{(j)} = f_{k_1}(P^{(i)})$. From this point forward, the two encryptions evolve together.

4. Hence, we consider all pairs $(P^{(i)}, C^{(i)})$ and $(\hat{P}^{(j)}, \hat{C}^{(j)})$ for all i, j and try to infer k_1 from the assumption that $\hat{P}^{(j)} = f_{k_1}(P^{(i)})$ and $\hat{k}_r$ from the assumption that $\hat{C}^{(j)} = f_{\hat{k}_r}(C^{(i)})$. By the related-key assumption, there will be some some known relation between k_1 and $\hat{k}_r$: if this is not satisfied we try further.

The attack requires in general 2^n comparisons, but it shall be noted that these involve recovering information about the key from the function f, twice each time, and not full evaluation of the cipher, so these comparisons may be much faster than brute force, albeit by just a small factor.

If E is a two branch balanced Feistel cipher, the right half of P is equal to the left half of $\hat{P}$ and the right half of C is equal to the left half of $\hat{C}$. This dramatically reduces the attack complexity: we need only $2^{n/4}$ plaintexts $P^{(i)} = (L_0^{(i)}, A)$ and $2^{n/4}$ plaintexts $\hat{P}^{(j)} = (A, \hat{R}_0^{(j)})$ for a fixed A. Also, the relation between the ciphertext accelerates the fourth step, since we only have to look at the pairs (i, j) for which the right half of $C^{(i)}$ is equal to the left half of $\hat{C}^{(j)}$.

2.6.1.1 Countermeasures

To counter related-key attacks minimal differences in the key must also be subject to diffusion in the cipher. This does not necessarily imply a complex key schedule – but confusion and diffusion of the key bits themselves must be ensured by the design. As a rule of thumb, ciphers that do not mix the key too often should perform non-trivial key scheduling, but there are exceptions, such as LED (Subsection 3.37.4), that always mix the same value, and only every four rounds, and have still withstood related-key cryptanalysis.

Whether to consider related-key attacks depends also on the intended application.

2.6.2 Slide Attacks

David Wagner and Alex Biryukov presented the slide attack in 1999 [BW99], and extended it one year later [BW00]. The idea of the slide attack has roots in an IBM Tech Report by Edna Grossman and Bryant Tuckerman in 1977 [GT77], later published as [GT78], where a block cipher named New Data Seal (NDS) was successfully attacked. The attack exploits weaknesses in the key schedule, especially periodicity. For instance, the NDS has identical subkeys in each round, i.e. a cyclic key schedule with a cycle length of one.

The attack also proves that just increasing the number of rounds (which makes sense to counter differential cryptanalysis) does per se not make a design stronger if one just repeats an already weak key schedule: The number of rounds in such a cipher is irrelevant.

Suppose that the target cipher can be broken down into multiple rounds of an identical function $f = f_k$ that is using the same key material k. The only additional requirements for the attack (in its simplest form) to work is that f is vulnerable to a known-plaintext attack – in other words that from $y = f_k(x)$ it is possible to recover some information about k. The crucial observation is that if $E(x) = f^r(x)$, then

$$\hat{P} = f(P) \qquad \text{implies} \qquad \hat{C} = E(\hat{P}) = f^r(f(P)) = f(f^r(P)) = f(C) \ .$$

We call a pair of plaintext/ciphertext pairs (P, C) and $(\hat{P}, \hat{C})$ where $\hat{P} = f(P)$ and $\hat{C} = f(C)$ a *slid pair*. If we represent the process of encrypting P and $\hat{P}$, where $P_i = f^i(P)$ and $\hat{P}_i = f^i(\hat{P})$, as in Figure 2.4, we see that the encryption of $\hat{P}$ can be paired with the encryption of P, *slid* by one place, resulting in the same values at the same positions.

Figure 2.4: A Slid Pair

$$\begin{array}{ccccccccccccc}
P_0 = P & \xrightarrow[f]{k} & P_1 & \xrightarrow[f]{k} & P_2 & \xrightarrow[f]{k} & P_3 & \xrightarrow[f]{k} & \cdots & \xrightarrow[f]{k} & P_{r-1} & \xrightarrow[f]{k} & P_r = C \\
\| & & \| & & \| & & \| & & & & \| & & \| \\
\hat{P}_0 = \hat{P} & \xrightarrow[f]{k} & \hat{P}_1 & \xrightarrow[f]{k} & \hat{P}_2 & \xrightarrow[f]{k} & \hat{P}_3 & \xrightarrow[f]{k} & \cdots & \xrightarrow[f]{k} & \hat{P}_{r-2} & \xrightarrow[f]{k} & \hat{P}_{r-1} \xrightarrow[f]{k} \hat{P}_r = \hat{C}
\end{array}$$

Now we can describe the attack. We start collecting candidate slid pairs. For any such candidate pair (P, C) and $(\hat{P}, \hat{C})$ we try to recover the key exploiting the weakness of f, using *both* equations $\hat{P} = f_k(P)$ and $\hat{C} = f_k(C)$. We recover two (partial) keys: if these do not agree, then the considered pair is discarded, otherwise we have a slid pair with high likelihood, and if some key bits are still to be determined a brute force attack will complete the attack.

With $2^{n/2}$ plaintext-ciphertext pairs where n is the block and key size, one slid pair is expected by the birthday paradox. The number of comparisons is however 2^n, which is the same as the number of *encryptions* in a brute force attack, but, as in Biham's related key attack, these comparisons are usually faster than encryption operations. Therefore, even though this strategy does not automatically lead to attacks significantly better than brute force, it is not uncommon that it can show that the actual security level is smaller by at least a small number of bits.

The probability that the wrong key will correctly encipher two or more messages is very low for a good cipher. False positives can be eliminated by using additional plaintext/ciphertext pairs.

On the other hand, if the cipher admits equivalent keys, the attack could found one such instead.

2.6.2.1 Slide Attacks on Feistel Ciphers

Sometimes the structure of the cipher can greatly reduce the complexity of the attack. The clearest example of this is given by the Feistel cipher using a single round key. The simplifications and complexity reduction are the same as in Biham's "sliding" related key attack.

Let the block size of the target balanced Feistel be n bits. Each of the two branches is $n/2$ bits wide. Set up the slide attacks as depicted in Figure 2.5 on the next page. The plaintexts of the two slid encryptions are of the form $L\|A$ and $A\|R'$. The attacker chooses $2^{n/4}$ different values for L, encrypting the plaintexts $L\|A$, and $2^{n/4}$ different values for R', asking for the encryption of the plaintexts $A\|R'$. Each ciphertext $E(L\|A) = M\|N$ is compared to each $E(A\|R') = M'\|N'$, and by the slide property, if we found a slid pair we must have $N = M'$, therefore pairs that cannot be candidates are quickly discarded. Now, the computation of key candidates is very efficient: from the first round of the first encryption in the slid pair we have a candidate $k \stackrel{?}{=} F^{-1}(L \oplus R') \oplus R$, which is compared with the value $F^{-1}(M \oplus N') \oplus N$ obtained from the last round of the second encryption.

If the cipher is DES-like, as in Figure 2.6 on the facing page the round function is different: key

Figure 2.5: Slide Attack on a Feistel Cipher with a Single Round Key

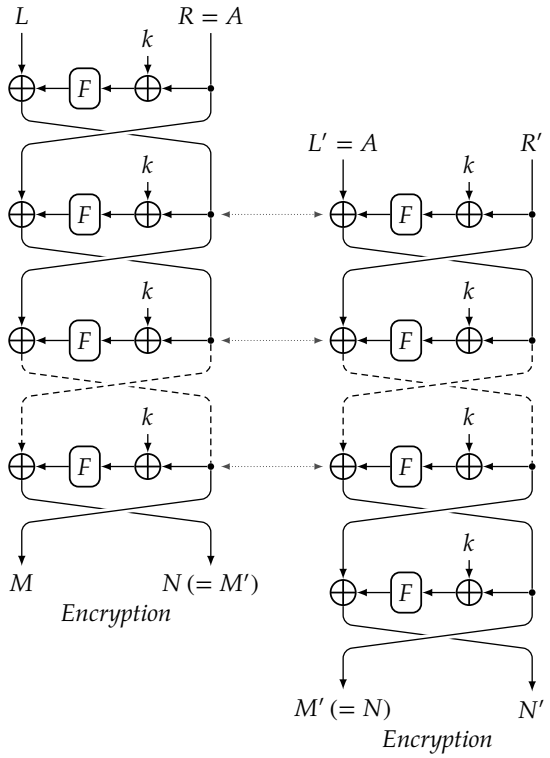
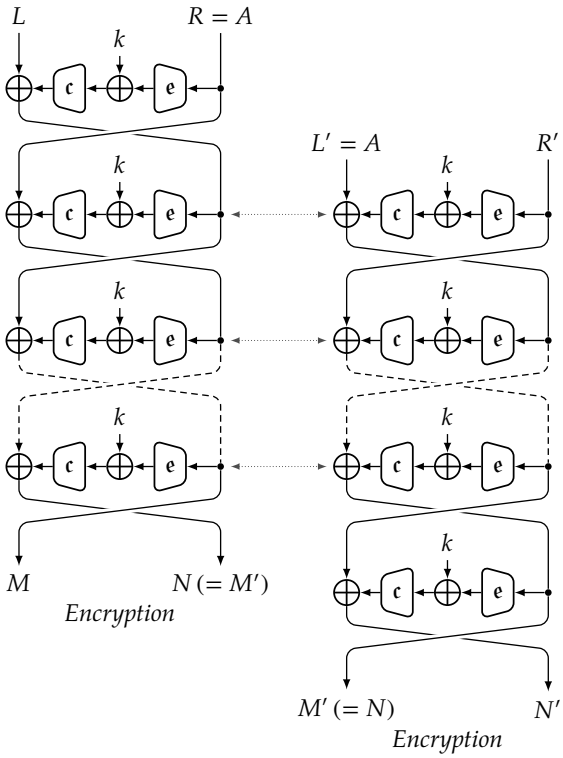


Figure 2.6: Slide Attack on a DES-like Cipher with a Single Round Key



mixing occurs after an expansion $\mathfrak{e}$ from 32 to 48 bits, and is followed by an SP layer, which we denote here by $\mathfrak{c}$ because it *compresses* its input from 48 bits to 32 bits (cf. Section 3.2 on page 127 and in particular Figure 3.3). Therefore there are 2^{16} candidate preimages for, say, $\mathfrak{c}^{-1}(L \oplus R')$, and thus 2^{16} candidate keys $\mathfrak{c}^{-1}(L \oplus R') \oplus \mathfrak{e}(R)$. These are intersected with the set obtained from the last round of the second encryption, namely $\mathfrak{c}^{-1}(M \oplus N') \oplus \mathfrak{e}(N)$, and then the remaining values are tested.

2.6.2.2 Countermeasures

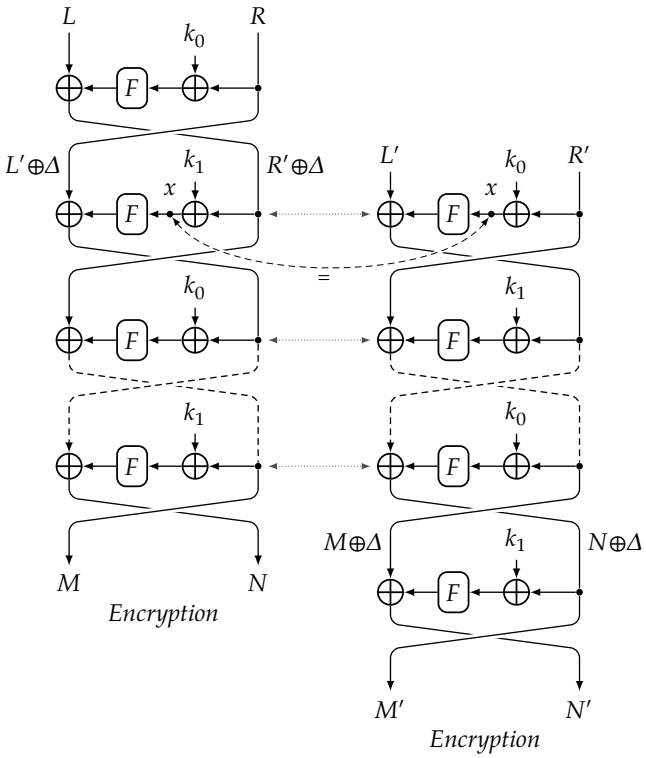
There are essentially three classes of countermeasures to thwart slide attacks (there variably effective against attacks on Even-Mansour constructions as well).

1. The first countermeasure is to use an irregular key schedule. This is crucial if the function f is always the same. Some ciphers include non-linear components in round key derivation, in other cases XORing the key (as in PRINCE, cf. Section 3.35 on page 215) or the state (as in LED, cf. Subsection 3.37.4 on page 227) with a different constant at each round may suffice.

2. Design the cipher with heterogeneous rounds. PRINCE, for instance uses three different types of rounds (two of are the inverse of each other, and there is a third type in the middle).

3. If a cyclic key schedule is used, the iterated block f should not be vulnerable to known plaintext attacks. In Feistel networks, non surjective F-functions only slow down the attacker by offering more choices for a key. A better approach consists in using bijective multi-round

Figure 2.7: Complementation Slide Attack on a Two Keys Feistel Cipher



SP Networks with additional keying between the SP layers (we note that such F-Functions do more than just making the cipher resistant against slide attacks, cf. Subsection 1.8.4 on page 53).

2.6.3 Advanced Slide Attacks

2.6.3.1 Complementation Slide Attack

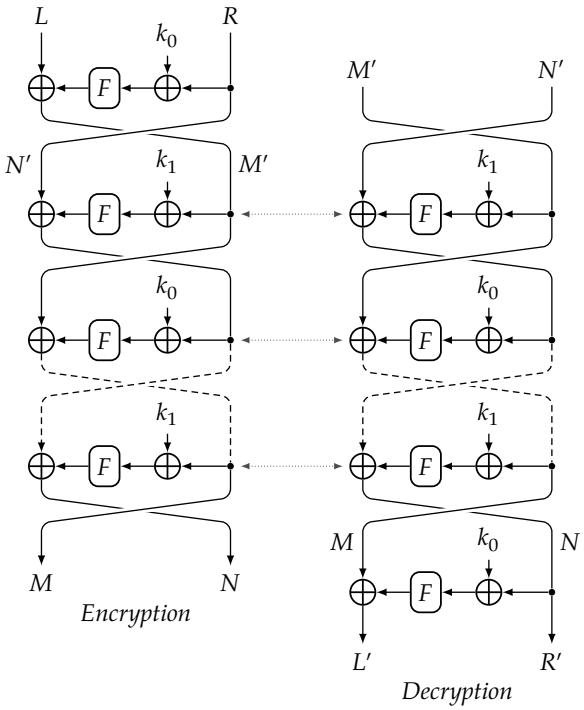
A Feistel-like cipher with two alternating round keys k_0 and k_1 (2K-DES) can be broken by a simple slide attack where each two rounds are paired in "super-rounds" that use all the key material. For this setting, Wagner and Biryukov [BW00] introduce a more efficient slide attack, called the *complementation slide attack*, represented in Figure 2.7.

Put $\Delta = k_0 \oplus k_1$, and consider two plaintexts $L\|R$ and $L'\|R'$ with $L' = R \oplus \Delta$ and $R' = L \oplus F(k_0 \oplus R) \oplus \Delta$. It is easy to prove by induction that these two plaintexts will form a slid pair with respective ciphertexts $E(L\|R) = M\|N$ and $E(L'\|R') = M'\|N'$, where $M' = N \oplus \Delta$ and $N' = (M \oplus \Delta) \oplus F(k_1 \oplus N \oplus \Delta) = M \oplus F(k_0 \oplus N) \oplus \Delta$.

Indeed, suppose that the i-th state of the encryption of $L\|R$ is equal to the $(i-1)$-th state of the encryption of $L'\|R'$ XORed with $\Delta\|\Delta$. Then, by the definition of Δ, the input and outputs of the F-function at the $(i+1)$-th round of the encryption of $L\|R$ are equal to those at the i-th round of the encryption of $L'\|R'$. This implies that $(i+1)$-th state of the encryption of $L\|R$ is equal to the i-th state of the encryption of $L'\|R'$ XORed with $\Delta\|\Delta$.

As a result, $L' \oplus M' = R \oplus N$ is half-state wide condition on a slid pair, which helps greatly

Figure 2.8: Slide Attack with a Twist on a Two Keys Feistel Cipher



reduce the amount of computations. From $M' = N \oplus \Delta$ we obtain a candidate for Δ. If the round function F is weak enough, we will now be able to derive key candidates k_0 and k_1. The work factor of the attack is thus in general very low.

If applied to a DES variant, the attack must be modified, since the difference in the data words is 32 bits, and the subkeys are 48-bit words. In this case $\Delta = k_0 \oplus k_1$ must be of the form $e(\delta)$, where $e(\cdot)$ is the DES expansion function and δ is the actual difference in the half states.

2.6.3.2 Slide Attack with a Twist

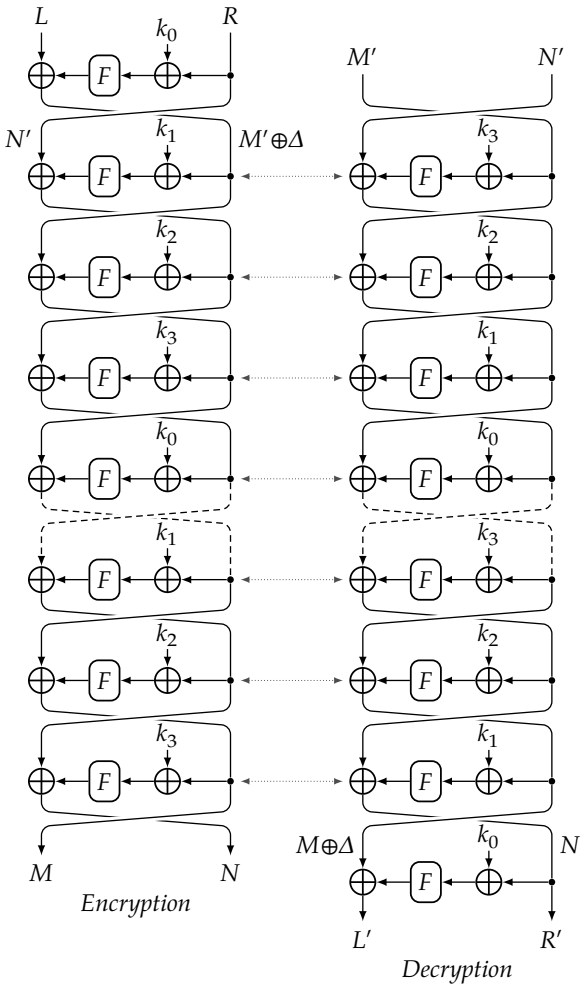
Another type of slide attack shown in [BW00] is the *slide with a twist attack*. Consider encryption and decryption in a two keys DES like Feistel block cipher as in Figure 2.8: the decryption is slid by one round with respect to encryption, so that the round keys are aligned.

Two plaintext/ciphertext pairs are considered as before, i.e. plaintexts $L\|R$ and $L'\|R'$ with respective ciphertexts $E(L\|R) = M\|N$ and $E(L'\|R') = M'\|N'$, i.e. $D(M'\|N') = L'\|R'$ (where, as usual, D denotes decryption). However we now relate the plaintext of one pair with the ciphertext of the other. If $R = N'$ and $M' = L \oplus F(R \oplus k_0)$, then one sees that $N = R'$ and $L' = M \oplus F(N \oplus k_0)$.

Given 2^{32} known plaintexts, it is possible to find a twisted slid pair and recover a round key. Since this is a Feistel cipher one can work with the encryption of 2^{16} chosen plaintexts $L\|A$ and the decryption of 2^{16} chosen ciphertexts $M'\|A$. Then, in a slid pair, the right part of $E(L\|A)$ will be equal to the right part of $D(M'\|A)$.

One can also combine the two techniques in order to analyze a *four round keys* Feistel cipher. The situation is depicted in Figure 2.9 on the next page. Note that two of the round keys, namely

Figure 2.9: Complementation Slide With a Twist Attack on a Four Keys Feistel Cipher



k_0 and k_2, appear at the same places in both members of the slid encryption/decryption pair, whereas the other two, namely k_1 and k_3, always appear facing each other, and the corresponding rounds have the constant difference $\Delta = k_1 \oplus k_3$ in the round subkeys. Therefore, the complementation slide technique can be applied, using texts with a slid difference of $0\|\Delta$.

2.6.4 Saarinen's Chosen Key Recovery of Secret GOST S-boxes

One distinctive feature of the GOST block cipher (Section 3.4 on page 138) is that the S-boxes can be kept secret and be varied from application to application. The cipher uses eight 4-bit S-boxes, which can be chosen freely but must be the same for each round. They they contain about 354 ($\log_2((16!)^8)$) bits of secret information, so one may be tempted to believe that the effective key size could, in theory, be increased to 610 bits.

In 1998 Markku-Juhani Saarinen [Saa98b] showed how to recover the contents of the S-boxes in approximately 2^{32} encryptions with chosen keys. Saarinen's attack is interesting since it is probably the first instances of a slide attack (cf. Subsection 2.6.2 on page 107) after the Grossman-Tuckerman 1977 attack on NDS [GT77, GT78]. It predates even David Wagner and Alex Biryukov's 1999 and 2000 papers [BW99, BW00].

Let $F(R,k)$ be the GOST F-Function, as depicted in Figure 3.7 on page 139. It is a 32-bit to 32-bit function, and it consists of an addition modulo 2^{32} with a 32-bit round key, the substitution layer consisting of the aforementioned eight 4-bit S-boxes, and a cyclic left rotation by 11 bits.

Saarinen's attack is based on the slid property of Feistel ciphers with constant key schedule, so the F-Function is $F(x) = F(x,k)$ with k fixed. If encrypting $P_1 = y\|x$ produces a ciphertext $C_1 = b\|a$ if $F(x) = y$, then encrypting $P_2 = x\|0$ will produce a ciphertext of the form $C_2 = a\|c$. i.e. the right half of C_1 is the same as the left half of C_2. Saarinen calls this the "lifting" property, and it is in fact a slid pair property where the second encryption is slid "up" one round.

The attack then proceeds in two steps. The first step searches for a 32-bit "zero vector" $z = F(0)$, and it requires at most 2^{32} encryptions. The second step examines one S-box at the time and extracts the contents of that S-box, at the cost of at most 2^{11} encryptions.

In the first step the attacks sets the key to 0. As a result the round keys are all 0. Let a be the right half the encryption of the zero block $0\|0$. Then, loop over the plaintexts $z\|0$ until a ciphertext is found whose left half is equal to a. There is a high probability that this z is the zero vector $z = F(0)$. We can now move to the second step with this value of z: If that step fails, we will resume the search for the zero vector starting with $z+1$; the zero vector is always found within 2^{32} encryptions.

In the second step we start by encrypting $a\|0$. Let x be *right* half of the ciphertext. If $F(a) = b$, then the *left* half of the encryption of $b\|a$ will be x. To test whether $S_i(u) = v$, one can thus set

$$\begin{aligned} a &= u \ll 4i \\ b &= \left(z \wedge \overline{1111_2 \lll 4i + 11}\right) \vee (v \lll 4i + 11) \end{aligned}$$

and verify whether the left half of the encryption of $b\|a$ is indeed x. We see that a is constructed by shifting the 4 bits of the value u so that they are fed into the i^{th} box S_i, and b is obtained from z by replacing the guessed $u = S_i(v)$ in the corresponding *output* bits. Then, assuming $F(0) = z$, it is $F(a) = b$. Since there are eight 4-bit S-boxes, a naive trial and error algorithm will determine the contents of all S-boxes in no more than $8 \times 2^4 \times 2^4 = 2^{11}$ encryptions – if z is correct. If the resulting S-boxes are not permutations of $[0..15]$ or are otherwise wrong (one can just check on some known plaintext/ciphertext pairs), the search for the right z resumes.

2.6.5 Generalised Even-Mansour Schemes

Generalisations of the Even-Mansour scheme with more key mixing rounds alternated with arbitrary permutation layers, as depicted in Figure 2.10 on the next page, have been extensively analysed. Even though Even-Mansour schemes per se are not used to design a cipher, mostly for performance reasons, they are very useful as tools in cryptanalysis and as tools for block cipher designers.

They can be used to approximate the design of traditional iterated or product ciphers, isolating and abstracting some of their aspects and allowing the cryptanalyst to draw design principles, mostly in the form of choices to avoid. Dozens of papers devoted to the cryptanalysis of EM schemes have been published since 2011 at important cryptography conferences.

The attacks on EM schemes that we shall present are related to slide attacks, the simplest case being that of a multi-round EM scheme with a single key.

Figure 2.10: General Even-Mansour Scheme

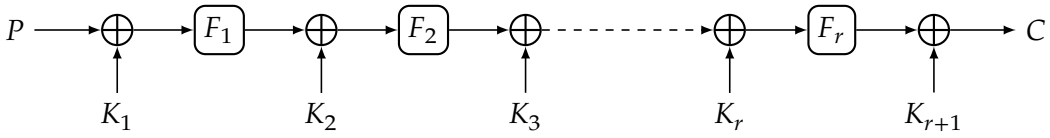
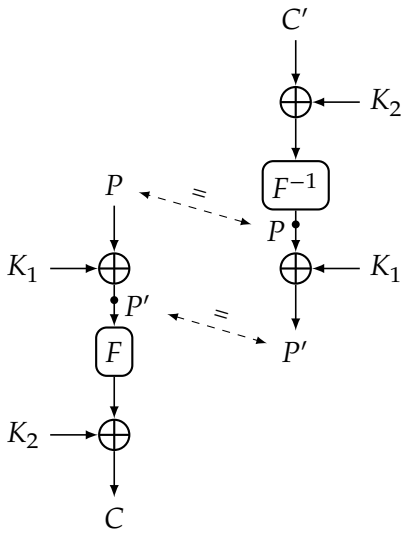


Figure 2.11: Slide With a Twist Attack on an Even-Mansour Scheme



In this subsection, n is the bit size of the plaintext, state and keys, and $N = 2^n$. T and S denote the time complexity and storage of the attacks.

2.6.5.1 Sliding with a Twist on Even-Mansour Schemes

One may be tempted think that key whitening (i.e. the FX construction) would effectively function as a "barrier" against slide attacks, similarly to its function against linear and differential cryptanalysis. However, a crucial observation in [BW00] is that the slide with a twist technique can be used to "penetrate" one level of key whitening and therefore mount attacks, say, on DESX, that work more efficiently than differential cryptanalysis. Their example is a slide attack with a twist on DESX (Subsection 3.2.9 on page 132), that is in fact more general, and we shall first describe it in the context of Even-Mansour schemes.

Let $F(x)$ denote any random permutation. We define an Even-Mansour encryption scheme under the key $K = \langle K_1, K_2 \rangle$ as $P \mapsto EM^F_{K_1,K_2}(P) = K_2 \oplus F(P \oplus K_1)$. Now consider an EM-encryption and a EM-decryption, with the latter slid up as depicted in Figure 2.11.

We say that the two known plaintext pairs (P, C) and (P', C') form a (twisted) slid pair if they satisfy $P \oplus P' = K_1$. Consequently, for any slid pair, we will have

$$C' = K_2 \oplus F(P' \oplus K_1) = K_2 \oplus E(P) \quad \text{as well as} \quad C = K_2 \oplus F(P') \ .$$

Combining these two relations yields $K_2 = C \oplus F(P) = C' \oplus F(P)$, and we see that slid pairs

must satisfy the *sliding relation*

$$C \oplus F(P) = C' \oplus F(P') \tag{2.17}$$

(or, equivalently, $C \oplus C' = F(P) \oplus F(P')$). To find a single slid pair, we then collect enough known plaintext/ciphertext pairs (P_i, C_i) in a table until the birthday paradox conditions are satisfied, and search for two of these that satisfy the sliding relation. This is optimised by creating a table of pairs $[C_i \oplus F(P_i), i]$ that is hashed or sorted sorted or hashed according to the first value and where i is the index of the corresponding plaintext/ciphertext pair. Once a collision between the first entries is found, the corresponding plaintext/ciphertext pairs (P_i, C_i) are immediately retrieved. Each candidate slid pair will immediately suggest values for K_1 and K_2 to be tested on a few additional texts.

In general, for n bits states and keys, the attack requires $S = 2^{n/2}$ known plaintexts and $T = 2^{n/2}$ additional queries to the encryption function to find a slid pair (here S and T mean space and time as in the discussion about space-time tradeoffs, cf. Subsection 2.4.2 on page 96).

If the function $F(x)$ is instead a keyed permutation $F(x) = E_K(x)$ with unknown key K, then the the plaintext/ciphertext pairs are still collected once, but the $[C_i \oplus E_K(P_i), i]$ must be created for each K. Applied to DESX, this results in an attack using $2^{32.5}$ known plaintext/ciphertext pairs, $2^{32.5}$ additional space, and time $2^{87.5}$ DES decryptions (not $2^{88.5}$ because of the DES complementation property). This attack can be modified into a ciphertext only attack with complexity 2^{95} (for instance, if we know enough statistical properties of the plaintext).

Alex Biryukov and David Wagner remark that "*the attack may be described without reference to sliding, but the sliding with a twist methodology made it possible to find the attack in the first place.*" The attack is nowadays usually depicted in the form given in Figure 2.12 on the next page, to better compare it to more recent improvements.

2.6.5.2 The SlideX Attack

The slide with a twist attack to an EM scheme does not lend itself to space-memory tradeoffs. But, sometimes we can only collect $S < 2^{n/2}$ texts. This case is addressed by the SlideX attack, introduced by Orr Dunkelman, Nathan Keller and Adi Shamir in [DKS12]. Consider the situation depicted in Figure 2.13 on the following page, and let P and P' be two distinct plaintexts with the property that

$$P' = P \oplus K_1 \oplus \Delta \ . \tag{2.18}$$

Then

$$\begin{aligned} C' &= EM^F_{K_1,K_2}(P) = F(P \oplus K_1) \oplus K_2 = F(P' \oplus \Delta) \oplus K_2 \quad \text{and} \\ C &= EM^F_{K_1,K_2}(P') = F(P' \oplus K_1) \oplus K_2 = F(P \oplus \Delta) \oplus K_2 \ . \end{aligned} \tag{2.19}$$

Hence,

$$C \oplus F(P \oplus \Delta) = C' \oplus F(P' \oplus \Delta) \ . \tag{2.20}$$

In light of this, we define a SlideX pair as a pair which satisfies the *SlideX relation* (2.18), and thus also (2.20).

To check for SlideX pairs, we take S plaintext/ciphertext pairs (P_i, C_i), and for each guess of Δ, we construct a table of all values $C_i \oplus F(P_i \oplus \Delta)$. Then $O(S^2)$ pairs are checked for the SlideX relation – for each match, candidate keys K_1 and K_2 are obtained at once from (2.18) and (2.19)

Figure 2.12: Slide With a Twist Attack on an Even-Mansour Scheme, Refactored

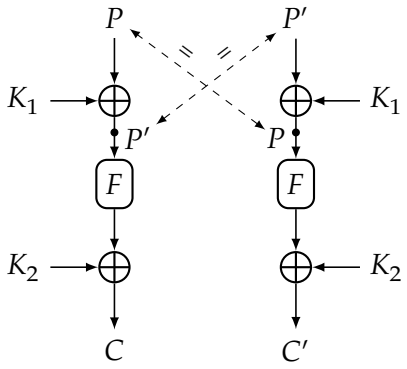
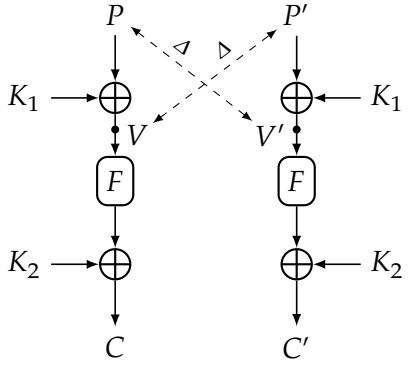


Figure 2.13: The SlideX Attack on an Even-Mansour Scheme



respectively. The construction of the table is repeated $O(2^n/S^2)$ times, each time with S calls to F, whence $T = O(2^n/S)$. This matches the information-theoretic security lower bound under the assumption that F is a random permutation.

2.6.5.3 Further Analysis of Even-Mansour Constructions

Multi-round Even-Mansour constructions have attracted considerable interest since the development of ciphers such as LED (Subsection 3.37.4 on page 227) that consist of relatively simple key scheduling (constant or alternating) and heterogeneous functions (usually derived from the same function but with different constants mixed in at each step). This suggests to study the security of such a cipher modeled as an iterated EM scheme, taking into account only the key schedule of the cipher and not the properties of the specific internal permutations.

2.6.5.3.1 One-Round Single-Key Even-Mansour Constructions

One-round, single-key EM schemes, as in Figure 2.14 on the facing page, can be attacked by a simplified variation of the SlideX attack: if $P_i + K = X_i$ and $C_i + K = Y_i$, then $P_i + C_i = X_i + Y_i$. For each known text (P_i, C_i) we calculate $P_i + C_i$ and store it in a table along with P_i. Then, for random values X_j we calculate $Y_j = F(X_j)$ and search $X_j + Y_j$ in the table. For each match, we test the candidate key $K = P_i + X_j$. If we collect S texts, a match is expected in time $T = 2^n/S$. Assuming $S \leqslant 2^{n/2}$, the time complexity of the attack is $max(T, S) = T$.

2.6.5.3.2 Two-Round Single-Key Even-Mansour Constructions

At FSE 2013 Ivica Nikolic, Lei Wang and Shuang Wu [NWW13a] proved that even 2-Round EM constructions with one key do not provide n-bit security. We describe here a variant due to Itai Dinur, Orr Dunkelman, Nathan Keller and Adi Shamir [DDKS13b, DDKS13c]. The setting of the attack is depicted in Figure 2.15 on the next page.

The preprocessing phase consists in evaluating F_1 on $S = \lambda 2^n$ arbitrary inputs X, where $\lambda \leqslant 1$, and find t-way collisions on $F_1'(X) := X \oplus F_1(X)$. Such a t-way collision is a t-uple of values $X_1, X_2, \ldots, X_t$ such that $X_i \oplus F_1(X_i)$ are all equal to a single value by Δ. Note that $P_i \oplus V_i = X_i \oplus Y_i$, and thus for an arbitrary P_i, we have $P_i \oplus V_i = \Delta$ with probability $t/2^n > 1/2^n$.

In the online stage, for each (P_i, C_i) assume that $V_i = P_i \oplus \Delta$ and compute $W_i = F_2(V_i)$. From

Figure 2.14: Single Round Even-Mansour Scheme With One Key

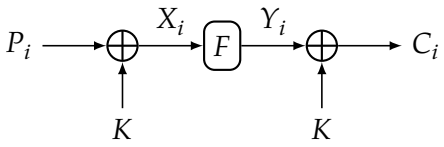
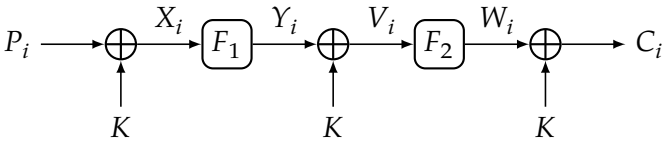


Figure 2.15: Two Rounds Even-Mansour Scheme With One Key



this we obtain a suggestion $K = W_i \oplus C_i$, which we test.

The preprocessing time complexity is $\lambda 2^n$ – and this is also the amount of precomputed data. The online time complexity is also $2^n/t$. The total time complexity is therefore $\lambda 2^n + 2^n/t$, but we do not have determined λ and t yet. To calculate the optimal time complexity, we need to understand the tradeoff between these two parameters, i.e. establish how t-way collisions behave when evaluating a fraction λ of inputs for F_1'.

Assuming F_1 is a random permutation, we can assume that F_1' is very close to a random function mapping n bits to n bits. The in-degree of a vertex (i.e. the number of preimages) in the range of F_1' can therefore be assumed to follow a Poisson distribution, i.e. there are

$$\frac{2^n \lambda^t e^{-\lambda}}{t!} \tag{2.21}$$

values which have in-degree t. The tradeoff between λ and t is subject to the additional condition $(2^n \lambda^t e^{-\lambda})/(t!) \geqslant 1$ and taking $\lambda \approx 1/n$ gives $t \approx 1/\lambda \approx n$ and this minimizes $T \approx 2^n/n$. This is faster than exhaustive search by a factor of about n.

For $n = 64$ we have that T and S are $\approx 2^{60}$. However, equation (2.21) shows that the number of t-way collision increases sharply when t decreases. For instance, if $n = 64$, with 2^{60} inputs the number of expected 10-way collisions is 4, that of 9-way collisions is 95, and that of 8-way collisions exceeds $100,000$. Using 8-way collisions the time total complexity grows marginally but the *online* data processing decreases to 2^{45}. This is the complexity of an attack to two round LED-64 as well.

2.6.5.3.3 More Even-Mansour Constructions

Three round EM does not provide n-bit security as well. Further developing the ideas of the attack against the two round version, in [DDKS13b, DDKS13c] Itai Dinur, Orr Dunkelman, Nathan Keller and Adi Shamir show how to obtain an attack that is faster than brute force by a factor n as in the two round case. In the case of three round reduced LED-64 this leads to an attack that requires total time 2^{60}, storage for precomputations of 2^{60} values, working on 2^{49} known texts.

Now consider a eight round two alternating key Even-Mansour construction. If we guess the first key K_1, this fixes also three internal mixings of this key besides the two outer ones. Therefore we are left with a 3 round single key (K_2) Even-Mansour construction that is attacked as just mentioned.

Eight round LED-128 is then amenable to an attack that requires total time 2^{124}, storage for 2^{60} precomputations, processing 2^{49} known texts.

Four-round EM constructions with two alternating keys are analysed as well, the result being that using S memory they can be defeated with total time $T = 2^{2n}/S$ as long as $T \geqslant 2^n$, hence the effective security level is 2^n. Hence, four round LED-128 offers only 64 bits of security.

The most remarkable lesson is that an alternating key schedule does not necessarily offer security better than n-bits unless the number of rounds exceeds 8, but the storage and time requirements still make the attacks unfeasible.

In [DKS12] additional variants of the SlideX attack on EM constructions are considered. Modular addition is considered as an alternative to XOR, and essentially all attacks carry over. Also, the influence of involutory rounds is considered. A single round EM scheme with two keys K_1 and K_2 where the round is an involution can be attacked by a variant of SlideX called the *mirror slide* revealing $K_1 \oplus K_2$ with $2^{n/2}$ queries to the encryption oracle and no queries to the involutory function.

In [DDKS13d] (the full version of [DDKS13b, DDKS13c]) serious consequences of the use of involutions in EM schemes are presented. A random involution F on a n-bits space has an expected number of approximately $2^{n/2}$ fixed points[1] and therefore the vertex 0 in the graph of the function $F'(x) = x \oplus F(x)$ has an expected in-degree of $2^{n/2}$. This is much larger than the $t = O(n)$ in-degree for a random permutation and applied to two, resp. three round EM schemes it permits to drop the time complexity from $T = 2^n/t$ to $T = 2^{n/2}$, resp. $T = 2^{3n/4}$, if just one (any one) of the permutations is an involution. This leads to the paradox that if we add an involution to a two round EM construction the cipher becomes *weaker*.

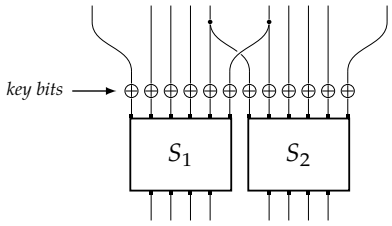
2.7 Statistical Analysis

Statistical analysis is one of the oldest cryptanalytic techniques, used to break many simple substitution ciphers. In the context of modern block ciphers statistical analysis is a generalisation of linear analysis – no inhomogeneity should be present in the output of a block cipher regardless of statistical properties of the plaintext – the output should always look like a pseudorandom sequence. Even recently proposed ciphers – such as Madryga (Subsection 3.3.1 on page 133) – succumbed to ciphertext only attacks based on statistical assumptions on the plaintext.

Pascal Junod extensively studied this line of research in his PhD research [Jun01a, Jun01b, Jun03a, Jun03b], and applied it to the design of FOX [JV04a]. FOX is also one of the very few ciphers that follows the Lai-Massey design (cf. Section 1.5 on page 36).

[1] Let a_m be the number of involutions over $[m]$. Fix an element $x \in [m]$. For any one of the a_m involutions on $[m]$, x is a fixed point if and only the permutation it induces on $[m] \setminus \{x\}$, i.e. a permutation on $m-1$ elements, is involutory itself, and there are a_{m-1} such. This holds for each x, thus the expected number of fixed points of an involution over $[m]$ is $\approx m \cdot a_{m-1}/a_m$. From [FS09, Example VIII.9, page 583] we know that $a_m \approx \sqrt{m!}e^{\sqrt{m}}(8\pi em)^{-1/4}$, from which we get that the desired number of fixed points is asymptotically $\sqrt{m} - 1/2 + o(1)$. Now put $m = 2^n$. An exact formula is given by Lemma 2.6 in [YTH96], which confirms that the asymptotic formula gives good results already for $n = 8$.

Figure 2.16: Relation Between Adjacent S-boxes of the DES (as in Figure 3.3 on page 129)



Any component in a new cipher design should be tested against any deviations from perfect homogeneity: The whole arsenal of statistical methods used to analyze PRNGs (pseudo-random number generators) can be deployed.

2.7.1 The Davies-Murphy Attack

The Davies and Murphy Attack – sometimes known just as Davies' attack [sic] – is a statistical cryptanalysis technique originally devised to attack the DES (Section 3.2 on page 127). It is known-plaintext attack. It was originally created in 1987 by Donald Davies and published eight years later in a joint paper with Sean Murphy [DM95]. The technique was improved in 1994 by Eli Biham and Alex Biryukov [BB97]. Davies' attack, at least in theory, can be adapted to other Feistel ciphers besides DES.

The S-boxes of the DES are balanced, i.e. when the input to any DES S-box is uniformly distributed, then the output is uniformly distributed as well. However, the expansion $e(\cdot)$ in the F-function of DES generates correlations between the output bits of adjacent S-boxes, when a fixed key is used. In fact, two adjacent S-boxes have an input size of 12 bits; 12 bits of the round key but only 10 bits of the expanded state, say α, are combined to be used as the input to the two S-boxes. Two bits of α are duplicated and the two instances of each of these bits are XORed with two different key bits, and then go into the two S-boxes. This is what causes non-uniformity in the output of the two S-boxes (and this applies to triples of S-boxes as well) and also makes the correlation dependent on the key. The situation is depicted in Figure 2.16.

However, non-uniformities must be observed at a higher level in the cipher, since the outputs of individual S-boxes cannot be observed when all we have is an encryption oracle.

The attacks this also makes use of the following observation. With respect to Figure 3.2 on page 128, we have $F(R_i, k_i) = L_i \oplus R_{i+1}$ Hence, if $i < r$, we have:

$$F(R_i, k_i) = L_i \oplus L_{i+2} \ . \tag{2.22}$$

This remark is true for each round except the last one (where $i + 2$ has no sense).

Now, ignoring the initial and final permutation, let $L\|R = L_1\|R_1$ and $L'\|R' = R_{r+1}\|L_{r+1}$ be the plaintext and ciphertext, respectively. If we take the XOR of the equations (2.22) for all even i, we obtain the following relation:

$$R \oplus L' = \bigoplus_{j=1}^{r/2} F(R_{2j}, L_{2j}) \ .$$

If we XOR the odd rounds, we obtain instead:

$$R' \oplus L = \bigoplus_{i=1}^{r/2} F(R_{2i-1}, L_{2i-1}) \ .$$

Each plaintext/ciphertext pair thus gives the XOR of the outputs of the F-Functions of the even rounds, as well as the corresponding sum for the odd rounds. That is what allows the attacker to observe a non-uniform distribution of the outputs of the F-Functions (induced by the non-uniformity of S-box pairs) by analysing a large quantity of plaintext/ciphertext pairs.

The attack starts by calculating the empirical distribution of certain characteristics based on many known plaintext/ciphertext pairs. Bits of the key can thus be deduced given sufficiently many known plaintexts by correlating the outputs with the inferred distributions. Davies' original attack allowed to find 2 parity bits requiring $2^{56.6}$ known plaintexts, and finding 16 parity bits requires $2^{85.6}$ known plaintexts. Thus, in his original form, the attack is slower than exhaustive search.

Eli Biham and Alex Biryukov's improvements permit to mount attacks faster than brute force. These consist in: considering different pairs of S-boxes than Davies; splitting the attack in a known plaintext collection phase and an online analysis phase; and describing tradeoffs between the number of plaintexts, the success rate and the time of analysis. One tradeoff requires 2^{50} known plaintexts and 2^{50} steps of analysis with a 51% success rate to break the full DES. A different attack can find 24 bits of the key with 2^{52} known plaintexts with 53% success rate, using only 2^{17} DES encryptions and 2^{31} further elementary machine instructions in the analysis phase. Suggestions how to design S-boxes immune to these attacks are also given in [BB97].

In 1998, Thomas Pornin [Por98] developed techniques for analyzing and maximizing a cipher's resistance to this kind of cryptanalysis.

Sébastien Kunz-Jacques and Frédéric Muller [KM05] further reduced the complexity of the Biham-Biryukov variant to 2^{45} *chosen* plaintexts and study the relation of the Davies-Murphy attack to linear cryptanalysis.

2.8 Algebraic Methods

2.8.1 Interpolation Attacks

Interpolation attacks can are the first algebraic attacks on block ciphers. The underlying intuition of this attack is that the relationship between plaintext and ciphertext can always be expressed as a set of polynomial expressions. If these have sufficiently low degrees, an attacker can reconstruct them from known (or chosen) plaintexts and the corresponding ciphertexts. Then, he can encrypt any plaintext of his choice without knowing the secret key.

Thomas Jakobsen and Lars Knudsen first presented this attack in [JK97], where it was shown that S-boxes represented by functions of low degree, even if provably secure against linear and differential attacks, could fall to interpolation attacks. The name of the attack comes from the fact that the Lagrange interpolation formula is used to determine the interpolating polynomials once a sufficient number of plaintext/ciphertext pairs have been computed.

To make matters worse, shortly thereafter Amr Youssef and Guang Gong [YG00] showed that in-

terpolation attacks are possible even when using monomial S-boxes (i.e. S-boxes representable by a power function over a finite field) regardless of their degree. This observation is useful for attacking ciphers using simple algebraic functions (in particular quadratic functions) as S-boxes. Also, ciphers of low non-linear order are vulnerable to higher order differential attacks.

Jakobsen and Knudsen give upper bounds on the number of required pairs for known-plaintext interpolation attacks to succeed for selected examples. This number is exponential in the degree of the polynomial function describing the S-box(es), the number of rounds and the size of internal state. This means that in general interpolation attacks are difficult to mount, because the ciphers must be quite special, and computing the polynomials may become prohibitively expensive.

2.8.2 Algebraic Attacks

Algebraic attacks break a cipher by representing it a set of polynomial functions of its inputs, then substituting in known data for some of the variables and solving the resulting multivariate system of polynomial equations for the key.

Nicolas Courtois, Alexander Klimov, Jacques Patarin, Adi Shamir presented the framework of these attacks at Eurocrypt 2000 [CKPS00]. The "XL algorithm" (eXtended Linearisation) – an improvement on Aviad Kipnis and Adi Shamir's relinearisation method [KS99] – is their method for solving these systems.

Courtois applied the XL algorithm initially to stream ciphers [Cou03, CM03]. Later, he and Josef Pieprzyk [CP02a] observed that Rijndael (and, partially, also Serpent) could be expressed as a system of quadratic equations, and thus adapted XL to this setting. This attack has been described also in the case of the DES [CB06, CB07], leading to a 6 round key recovery attack and a 12 round plaintext recovery attack.

There is more than one method can be used for solving multivariate systems of polynomial equations. Gröbner bases algorithms can be used, and the XL algorithm is sometimes proposed as an alternative to, but in fact XL is just an inefficient Gröbner basis algorithm in disguise [AFI+04]. Other methods include resultant-based methods, SAT solvers (for instance [ES]) and a method by Håvard Raddum and Igor Semaev [RS06].

The underlying algebraic structure can be chosen by the attacker: against one cipher he might treat the text as a vector of bits and use Boolean algebra while for another he might choose to treat it as a vector of bytes and use arithmetic modulo 2^8, and for yet another cipher nibbles may be interpreted as elements of $\mathbb{F}_{2^4}$, or even as non-zero elements of $\mathbb{F}_{17}$.

The actual effectiveness of algebraic attacks is debated, often with heated tones. The following quote is one of the polite responses:

> *I believe that the Courtois-Pieprzyk work is flawed. They overcount the number of linearly independent equations. The result is that they do not in fact have enough linear equations to solve the system, and the method does not break Rijndael.*
>
> *Don Coppersmith, 2002*

At the AES 4 Conference, Bonn 2004, one of the inventors of Rijndael, Vincent Rijmen, commented, *"The XSL attack is not an attack. It is a dream."* Promptly Courtois answered, *"It will*

become your nightmare." The attack has not become a nightmare yet, but it has caused some experts to express greater unease at the algebraic simplicity of the current AES.

However, its possibility can not be excluded a priori during the design of a new cipher.

The most significant successful algebraic attack so far is Nicolas Courtois, Gregory Bard and David Wagner's cryptanalysis of the block cipher KeeLoq, an NLFSR-based block cipher used in the automotive industry with a 32-bit state and 64-bit keys [CBW08]. Using a combination of slide attacks and SAT (boolean satisfiability problem) solvers, they were able to successfully attack this cipher.

2.8.3 Gröbner Basis Attacks

Johannes Buchmann, Andrei Pyshkin and Ralf-Philipp Weinmann [BPW06] use Gröbner bases to implement algebraic attacks leading to key recovery. They first step in their attack is to write down polynomials $\{p_i\}$ that fully describe the cipher – these are Boolean functions on the key and plaintext bits. Then, a plaintext/ciphertext pair are used to create additional linear equations $\{g_i\}$ and the variety defined by the union set $\{p_i\} \cup \{g_i\}$ (called the key recovery ideal) is computed by Gröbner basis computations. This set of points represents, in fact, a list of key candidates, which is then sieved using further plaintext/ciphertext pairs, as usual.

The bulk of the complexity is in the Gröbner basis computations, in particular the computation of the variety (that amounts to solving the system) and in the creation of the Gröbner bases themselves and their conversion. For this purposes, the FLGM algorithm is used [FGLM93] as well as the Gröbner walk [CKM97].

The attack has been demonstrated on toy ciphers Flurry and Curry – these ciphers are however not trivial, since they show proven good resistance against differential and linear attacks.

2.8.4 Countermeasures

Despite the fact that no algebraic attack has successfully broken ciphers that have withstood intense cryptanalysis of other types – for instance the 2^{101} time complexity attacks on AES-256 are not taken seriously by part of the cryptographic community – it would be unwise to ignore the possibility and do not take at least some simple precautions. For instance, if the S-boxes do not have high degree, the cipher may be easily defeated by algebraic attacks as it happened for some toy ciphers.

Indeed, all forms of algebraic attacks can be made impractical when the system of equations and their degrees can be made sufficiently large. Cipher designers therefore strive to make their ciphers highly nonlinear. This is achieved not only by making the non-linear components, such as S-boxes, of as high degree as possible.

To this purpose the algebraic normal form (ANF) of the boolean functions involved in a cipher are considered and functions of as high degree and high algebraic immunity as possible should be used.

2.9 A Remark on The Rebound Attack

We do not discuss here the rebound attack. It is a useful tool in the cryptanalysis of cryptographic hash functions designed around AES-like compression functions, but it is not a specific

method for block ciphers.

It was first published in 2009 by Florian Mendel, Christian Rechberger, Martin Schläffer and Søren Thomsen [MRST09] to analyse the hash functions Whirlpool and Grøstl; It was later applied to Keccak, JH, Skein and other designs [LMR+09, LMR+10, MRST10, NP11, NPTV11, DGPW12].

Still, it is clear that methods to study the security of hash functions based on block ciphers are important when designing a block cipher if the use in a hash function is one of the intended use cases of the design. Therefore rebound attacks have been taken into account in the study of block ciphers [NPSS10]. LED is a block cipher designed taking rebound attacks into account for the case where it is used to construct a hash function [GPPR12] (Subsection 3.37.4 on page 227).

Chapter 3

Block Ciphers

I don't want to do architecture that's dry and dull.

Frank Gehry

We describe here an ample selection of block ciphers. The designs are often very different from each other and are a testimony to the creativity of the cryptographic community.

This chapter makes no attempt to be exhaustive: several ciphers are only mentioned in passing or are just omitted. The purpose is to exemplify the main design types and their variants, their strengths and weaknesses, especially with respect to algorithmic security and performance – while at the same time presenting all the most significant block ciphers designed in the last four decades.

We also do not aim at providing complete descriptions of all ciphers with full details – these can be found in the specifications, which are always referenced. We attempt to focus on the salient aspects of each design, such as the overall topology and the relations to other designs. In general we prioritise giving details about the main encryption routine over the key schedule.

The ordering of the presented ciphers is, for the most part, chronological according to the date of algorithm disclosure. There are a few exceptions, for instance when cipher families are treated together (for instance in the cases of DES, SAFER and Camellia) or when some ciphers have been grouped for historical reasons or extreme similarity.

There are countless block ciphers, so it is nearly impossible – and perhaps pointless – to provide a complete description of all designs together with all their variants. More than 70 ciphers are here described, but had to omit even more ciphers for several reasons, such as similarity to other designs, lack of complete specifications, and, ultimately, space and time. Some of the ciphers we did not include are: 3-Way, AIDA, Akelarre, ARIA, Armadillo, BaseKing, BassOmatic, BATON, CIKS-1, CIPHERUNICORN-A, CMEA, Cobra, COCONUT98, Crab, Cryptomeria/C2, CS-Cipher, DONUT, EPCBC, FEA-M, Grand Cru, Hummingbird, Hummingbird-2, ICE, Intel Cascade Cipher, Iraqi, Khufu and Khafre, KN-Cipher, Libelle, M6, M8, MacGuffin, Mercy, MMB, MULTI2, MultiSwap, New Data Seal, Nimbus, NUSH, PEANUT, Puffin, Puffin-2, Q, REDOC-II, REDOC-III, Red Pike, S-1, SAVILLE, Seed, SMS4, SPEED, Spectr-H64, SXAL/MBAL, Treyfer, UES, Vitamin-B, WALNUT, Xenon, xmx, and Zodiac.

Our exposition starts with Lucifer, one of the very first block ciphers, if not the first. There are indeed precursors that can be considered block ciphers as well, such as polygraphic substitution ciphers.

One of these is the Playfair cipher, invented in 1854 by Charles Wheatstone, and used until the second world war by Commonwealth forces and also by the German Army. Playfair operates on digraphs instead of single letters to make statistical analysis more difficult. A description of

the cipher and its cryptanalysis are given in Chapter 7 of the U.S. Army Training and Doctrine Command (TRADOC) *Basic Cryptanalysis* Field Manual [TRA90].

Another example is the Hill cipher. Invented by Lester S. Hill in 1929, it is a polygraphic substitution cipher based on linear algebra, the first such cipher that made it practical to operate on more than three symbols at once, and it could be implemented by a mechanical machine, described in US Patent 1,845,947.

However, Lucifer is the first mathematically non trivial family of block ciphers intended to be implemented on a general purpose computer. Hence, it will be our starting point.

3.1 Lucifer

In the late 60s Horst Feistel, and later Walt Tuchman, led an IBM research program to develop secure ciphers, called Lucifer. This project culminated in 1971 with the development of a series of ciphers called Lucifer as well. Some of these ciphers are pure SPNs, others are Feistel Networks that use a SPN in their round functions.

The most unifying aspect of this family of ciphers is the use of two S-boxes which are selected by key bits. We are aware of following members of the Lucifer family:

1. A SPN, described in U.S. Patent 3,798,359 that uses a 48-bit key and operates on 48-bit blocks. It uses two 4-Bit S-boxes and key bits select which of the S-boxes acts on each nibble of the state. The same S-box selection mechanism is used also in U.S. Patent 3,798,360 on a "step code ciphering system."

2. Another variant, described in U.S. Patent 3,796,830, uses a 64-bit key operating on a 32-bit block, using one addition mod 4 for key mixing followed by the application of an S-box - also chosen among two S-boxes depending on key values.

3. A stronger variant, described in [Fei73], uses a 128-bit key and operates on 128-bit blocks. The cipher is an SPN and uses two 4-bit S-boxes. The key selects which S-box is used. However, the paper [Fei73] is very scarce on details, which means that several cryptanalitic papers had to make often quite arbitrary assumptions in order to study "Lucifer-like" ciphers.

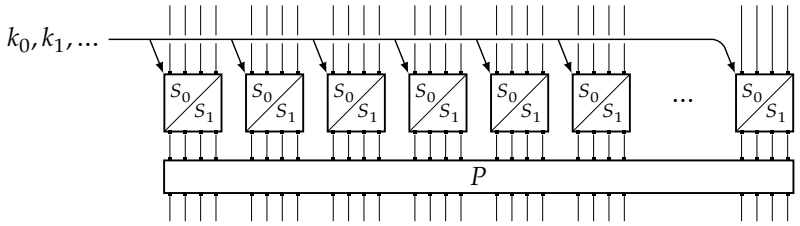
4. Later, a 16-round Feistel network version of Lucifer was described in [Sor84]. This is the first known examples of a practical cipher designed around a SPN, and it is the version we are going to describe in the following.

The first three versions require distinct implementation of encryption and decryption. The fourth version uses a single code path for encryption and decryption, with only the key schedule having to be reversed in order to perform either operation.

The fundamental building block of these ciphers is depicted in Figure 3.1 on the next page. The key bits $k_0, k_1, \ldots$ determine whether the S-box S_0 or S_1 is chosen.

The particular representation refers to the versions of the cipher with nibble-wide branches, i.e. the first three. The Feistel Network version of Lucifer considers the state byte-wise, and a single bit determines how the two 4-bit S-boxes transform the two nibbles in a byte, as we shall explain shortly.

Figure 3.1: The Lucifer Basic Building Block



The Feistel Network version of Lucifer has a 128-bit key and block length. It is a 16 rounds, balanced 2-branch Feistel construction (as in Figure 1.3 on page 29 but without IP and FP blocks) with a one round SPN as the round function. The latter consists of:

1. A layer of eight *pairs* of 4-Bit S-boxes. Only two S-boxes are used throughout, denoted by S_0 and S_1. Each byte of the state is transformed by two S-boxes acting in parallel on the two nibbles. The least significant byte of the current round key is used to choose which nibble is fed to which S-box: if the value of the i-th bit of the round key byte is one, the two nibbles of the state byte are swapped; then S_0 acts on the least significant nibble while S_1 acts on the most significant nibble – the S-box selection system is described in U.S. Patent 3,798,360.
2. Then the round key is XORed to the state.
3. Then each byte of the state is passed through a fixed permutation of its bits.
4. Finally, a simple convolution is computed on the 8 half-state bytes to guarantee diffusion.

The key schedule is very simple: the key is rotated by 7 bytes after each round. The round key then consists of the 8 least significant bytes of the (rotated) key. The least significant byte of the round key is used twice, as it also parametrises the S-boxes, as we have seen.

This version of Lucifer is susceptible to differential cryptanalysis; Preliminary results were published in [BS91b], using S-box values borrowed from DES. A complete analysis shows for about half the keys, the cipher can be broken with 2^{36} chosen plaintexts and 2^{36} time complexity [BAB93, BAB96].

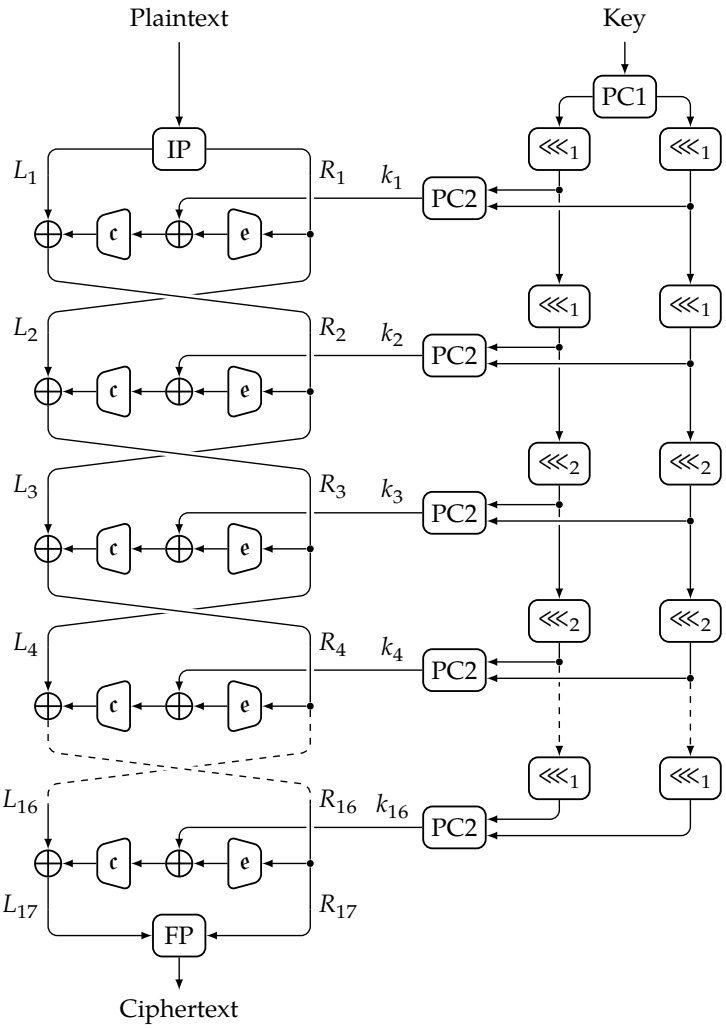
3.1.1 Intellectual Property

The Lucifer designs were covered by a few patents, some of which we already mentioned, that have since expired.

3.2 DES

A design borrowing ideas from Lucifer and with a key size reduced to 64 bits was submitted in 1976 by IBM to the National Bureau of Standards (NBS). The NSA then persuaded IBM to reduce the effective key size from 64 to 56 bits by including a parity bit in each byte of the key as the most significant bit, and also changed the S-boxes. This drew immediately considerable criticism, but it was later revealed that this made the S-boxes better resistant to differential cryptanalysis. The revised cipher was selected as an official Federal Information Processing Standard (FIPS) for the United States as the Data Encryption Standard (DES) in 1977.

Figure 3.2: The Data Encryption Standard



The design, represented in Figure 3.2, is now known as a *Feistel network*. DES has 16 rounds. The Initial Permutation (IP) and Final Permutation (FP) are the inverse of each other.

The cipher is still deployed despite being considered nowadays insecure.

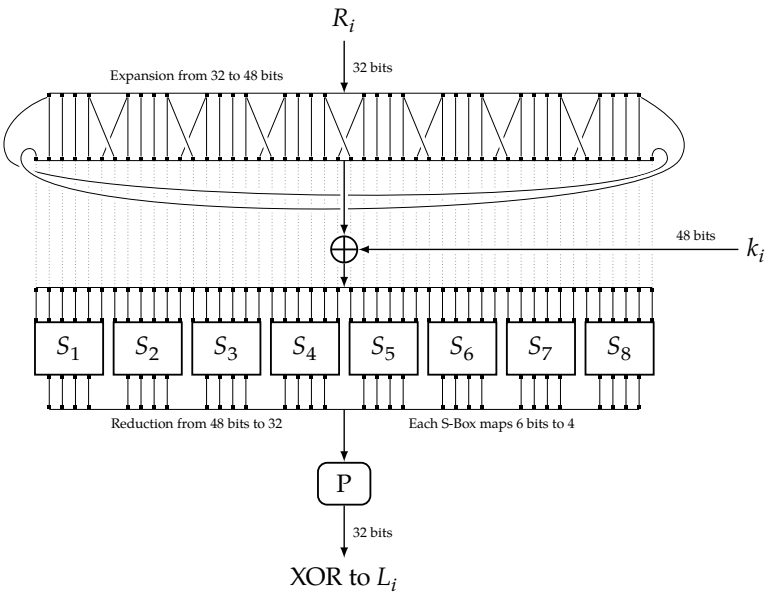
3.2.1 Description

3.2.1.1 The F-function

The round function, represented in Figure 3.3 on the facing page, operates on a 32-bit branch at a time and consists of four stages:

1. `Expansion`: the 32-bit half-block is expanded to 48 bits using the expansion permutation, by duplicating half of the bits. This function is represented by the letter **e** in Figure 3.2. The output consists of eight 6-bit chunks, each containing a copy of 4 corresponding input bits, plus a copy of the immediately adjacent bit from each of the input pieces to either side. Each S-box is such that for each fixed choice of the most significant bit and of the least significant

Figure 3.3: The DES Round Function



bit of the input, the output is a permutation of the values represented by the middle four input bits.

2. `Key mixing`: the result is XORed with a round key. Sixteen 48-bit round keys – one for each round – are derived from the main key using the key schedule.

3. `Substitution`: The eight 6-bit chunks of the state are transformed non-linearly by eight different S-boxes. The output of each S-box is just 4 bits long. Therefore the output of the substitution layer is just 32 bits wide. This step is merged with the next one and represented by the letter ɕ (because it *compresses* its input from 48 to 32 bits) in Figure 3.2.

4. `Permutation`: Also called the P-box, this is a fixed permutation of the output of the substitution layer. This guarantees diffusion.

The purpose of the expansion in the F-function is to allow the mixing of more bits in each round – in other words to improve confusion. We see here the heritage of Lucifer: Each 6-by-4-bit S-box can be viewed as four 4-bit S-boxes, selected by the first and the last significant bits of the input – and these are influenced by the round key. For instance, the first S-box is defined as follows, the input being given as $x\|Y\|z$, where x and z are single bits and Y is a four-bit value:

S_1:

(x,z) \ Y =	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
(0,0)	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7
(0,1)	0	F	7	4	E	2	D	1	A	6	C	B	9	5	3	8
(1,0)	4	1	E	8	D	6	2	B	F	C	9	7	3	A	5	0
(1,1)	F	C	8	2	4	9	1	7	5	B	3	E	A	0	6	D

3.2.1.2 The Key Schedule

The key schedule the splits the 56 effective bits of the key into two 28-bit halves. The function that partitions the bits is called Permuted Choice 1 (PC1).

These two halves are each rotated cyclically by fixed amounts at each round, either one or two bits depending on the round. The sequence of rotation amounts is irregular: in rounds 1, 2, 9 and 16 the rotation amount is one bit, in all other rounds it is two bits.

From the two rotated 28-bit halves, 48 bits are selected, 24 bits per each half, using a fixed function, called Permuted Choice 2 (PC2), to form the round key.

3.2.2 Other Properties

DES exhibits the *complementation property*, namely that

$$E_K(P) = C \iff E_{\overline{K}}(\overline{P}) = \overline{C}$$

where $\overline{x}$ is the bitwise complement of x. E_K denotes encryption with key K. P and C denote plaintext and ciphertext blocks respectively. This property implies that the work for a brute force CP attack could be reduced by a factor of 2 (or a single bit).

There are four *weak keys* (same encryption), and six pairs of *semi-weak keys* (encryption with a key in a pair is equivalent to decryption with the other key in the same pair). These keys can be easily avoided by checking against them.

It has been proved that DES has a maximum security level of 64-bits even if all 16 48-bit round keys were chosen independently (yielding a key space of 768 bits).

3.2.3 Cryptanalysis

Mathematically speaking, DES was broken by linear cryptanalysis in 1993 by Mitsuru Matsui [Mat93]. Linear cryptanalysis of DES requires time equivalent to 2^{39} to 2^{43} DES evaluations, as estimated in 2001 by Pascal Junod [Jun01a, Jun01b].

Davies' specific attack for DES, as improved by Biham and Biryukov [BB97], requires 2^{50} known plaintexts and has a computational complexity of 2^{50} with a 51% success rate (cf. Subsection 2.7.1 on page 119).

In June 1997, the DESCHALL Project (DESCHALL is short for DES Challenge), led by Rocke Verser, Matt Curtin, and Justin Dolske, successfully decrypted a DES encrypted message in 90 days [CD98]. This attack was a distributed brute force attack. In July 1998, the EFF's DES cracker (Deep Crack) broke a DES key in 56 hours.

This prompted the NBS to introduce Triple-DES (Subsection 3.2.8 below)) in 1999, start the AES specification process (finished in 2001, see Section 3.19 on page 175) and finally, in 2005, deprecate the use of DES altogether. Since then, there has been further progress, for instance the 2002 chosen-plaintext attack by François Koeune et al. [KRS+02] that could break the DES in less than 15 hours, using $3500 worth of hardware. As of today it is possible to break a DES key in a few hours using commercially available array of FPGAs such as COPACOBANA and RIVYERA, or cloud services like CloudCracker for just 17 dollars.

3.2.4 Advantages

Since DES is a Feistel network, encryption and decryption are similar. Decryption is in fact just encryption where the order of the round keys is reversed, so it requires very little additional SW or HW – mostly for flow control.

3.2.5 Disadvantages

The key size is too short to provide a reasonable level of security. It is practically broken by linear cryptanalysis and brute force. For today's standard, the throughput per security level is on the low side.

3.2.6 Intellectual Property

When the DES was introduced, U.S. Patent 3,962,539 covered aspects of its design, such as round functions that use expansion functions based such as $\mathbf{e}$. This patent expired on June 1, 1993. However, International Business Machines Corporation (IBM) granted nonexclusive, royalty-free licenses under the patents to make, use and sell apparatus which complied with the standard. In the meantime any patent pertinent to the original design has expired. This also explains why further cipher designs using state expansion in the round functions were not proposed for many years, the first notable new application of the idea being PICARO (Subsection 3.38.1 on page 228) about 35 years later.

3.2.7 The Bit-Slicing Implementation Technique

Not only did the DES stimulate an amazing amount of creativity in its attackers, it also spurred the creation of new implementation techniques.

For instance, DES can efficiently implemented by using the *bit-slicing* technique, as described first by Eli Biham in [Bih97b].

On a machine with w-bit registers, these are used as SIMD vectors of single bit elements.

Each bit of the state of the cipher is stored in a fixed bit of a different machine word, and the states of up to w different instances of the cipher can be thus stored in parallel, interleaved bit by bit. Each logical operation on these machine words performs the same operation in parallel on the corresponding bits of all the instances of the cipher.

Each component of the cipher must be implemented using logic operators: even S-boxes, that in traditional implementations are often performed by table lookup must be represented mathematically and described by short straight line programs.

This implies that the latency of a single encryption increases with respect to more traditional implementations. On the other hand, w mutually independent encryptions can be performed simultaneously. Thus, this approach proves useful in some contexts where latency is less of a concern, but combined high throughput of independent encryptions is desirable.

Bit-slicing has become then a cipher design criterion. I.e. ciphers have been designed where the internals of a single encryption can be parallelised and implemented efficiently in software, for instance when several parallel instances of a single S-box applied to the state and/or with clever bit wirings. Serpent (Section 3.17 on page 168) is one such example. More recent ones include Robin and Fantomas (Subsection 3.38.3 on page 230).

3.2.8 Triple-DES

To counter the recent successful attacks on DES, in 1998 ANSI X9.52 and FIPS 46-3 introduced the Triple-DES encryption algorithm. As the name suggest, it is a triple encryption process based on DES. It uses a "key bundle" which comprises three 56-bits DES keys K_1, K_2, and K_3. The encryption algorithm is:

$$\text{ciphertext} = E_{K_3}(D_{K_2}(E_{K_1}(\text{plaintext})))$$

I.e., DES encrypt with K_1, DES "decrypt" with K_2, then DES encrypt with K_3. The block size is unchanged at 64 bits.

The standards define three keying options:

1. All three keys are independent.
2. K_1 and K_2 are independent, and $K_1 = K_3$.
3. All three keys are identical, i.e. $K_1 = K_2 = K_3$.

In each case the middle operation is the reverse of the first and last. This improves the strength of the algorithm when using keying option 2, and provides backward compatibility with DES with keying option 3.

Keying option 1 is the strongest, with $3 \times 56 = 168$ independent key bits.

Keying option 2 provides less security, with $2 \times 56 = 112$ key bits, but is stronger than just simply DES encrypting twice, e.g. because it protects against meet-in-the-middle attacks. However, it can still be defeated in time and space $O(2^{56})$ by a chosen plaintext attack [MH81].

Keying option 3 is equivalent to DES, with only 56 key bits, it is now deprecated and was only provided to guarantee backward compatibility with DES.

3.2.9 DES-X

DES-X (sometimes also denoted DESX) is a variant of DES intended to increase the complexity of a brute force attack using key whitening. The idea was suggester in 1984 by Ron Rivest and consists in augmenting DES by XOR-ing an extra 64 bits of key k_0 to the plaintext before applying DES, and then XORing another 64 bits of key k_2 after the encryption:

$$\text{DES-X}_{K_0,K_1,K_2}(M) = \text{DES}_{K_1}(M \oplus K_0) \oplus K_2 \ .$$

The key size is thereby increased to $56 + (2 \times 64) = 184$ bits. This construction is also known as the FX Construction (see Section 1.6 on page 38) and was analyzed by Joe Kilian and Phillip Rogaway in [KR96b, KR01a]. The effective key size is then 118 bits (cf. Section 1.6 on page 38).

Advanced slide attacks break DES-X [BW00] with just $2^{32.5}$ data in time $2^{87.5}$ (known plaintext) or 2^{95} (ciphertext only).

Eli Biham estimates [Bih94a] (see also Burton Kaliski and Matthew Robshaw [KR96a]) that differential cryptanalysis would require 2^{61} chosen plaintexts (as opposed to 2^{47} for DES), while linear cryptanalysis would require 2^{60} known plaintexts (vs. 2^{43} for DES.)

3.2.10 GDES

In 1981 Ingrid Schaumüller-Bichl [SB81] proposed a DES variant called GDES (Generalized DES) with the purpose of improving both security and throughput of DES.

The most noticeable aspect of this cipher is that is is an *unbalanced* Feistel network. In other words, the "right" and "left" sides have variable lengths.

The block size is variable and a multiple of 32. In each round, the DES round function is applied to the rightmost 32-bit subblock, and the result is XORed with all the other parts. Then the block is rotated 32 bits to the right.

In 1990, Eli Biham and Adi Shamir showed that GDES is vulnerable to differential cryptanalysis, and that any GDES variant faster than DES is also less secure than DES [BS90, BS91a].

3.2.11 DESL and DESXL

DESL and DESXL are lightweight DES variants (the "L" stands for "lightweight") proposed by Gregor Leander, Christof Paar, Axel Poschmann, and Kai Schramm at FSE 2007 [LPPS07].

Unlike DES, DESL uses a single S-box repeated eight times. The S-box is designed taking into account cryptanalitic results on Feistel-network ciphers, so that DESL is resistant against certain types of the most common attacks, i.e., linear and differential cryptanalyses, and the Davies-Murphy attack.

DESXL is the variant of DESL that uses key whitening, similarly to DES-X.

3.3 Some Early Post-DES Developments

A flurry of new ciphers were introduced in the wake of the standardisation of the DES. Many of them have been broken quickly, however, some also introduced novel design ideas which have resurfaced many times since. We present here just four such ciphers: Madryga, NewDES, KeeLoq and FEAL.

3.3.1 Madryga

Madryga was designed by William Madryga [FD84]. We briefly discuss it because some of its design choices that have proven influential later.

The algorithm works with arbitrary key and block lengths. Key schedule consists in first XORing the key with a magic constant (that must be defined for each key size) and then rotating the key to the left by three bits before each round.

The state is processed three bytes at a time, cyclically. Each round the "window" of processing is moved by a byte along the state. So if the cipher were working on bytes 2, 3 and 4, at the following round it would process bytes 3, 4 and 5.

The operations in each round are very simple: The rightmost byte in the window is XORed with the rightmost byte of the (current) key, It XORs a key byte with the rightmost byte, and rotates the other two as one block. The rotation depends on the output of the XOR, hence it is both state and key dependent.

The cipher makes at least 8 cyclic passes over the whole state - the state at the end of the last

cycle is the ciphertext.

Decryption is easily implemented, since the rotation amount can be obtained before undoing the XOR.

3.3.1.1 Remarks

The cyclic processing of arbitrarily long states is one of the distinctive features of Block TEA (Subsection 3.12.3 on page 160) and XXTEA (Subsection 3.12.5 on page 162).

The byte-wise processing of the state without complex bit permutation layers makes the cipher easy to implement and very efficient in software. This is a deign principle that has been adopted (to varying degrees) in several other ciphers, such as IDEA (with 16 bit words, see Section 3.6 on page 142), the SAFER family of ciphers (Section 3.8 on page 148), and most of the ciphers designed according to the wide-trails strategy.

A few subsequent ciphers also use variable rotations, such as RC5 (Section 3.10 on page 156), and RC6 (Section 3.15 on page 165).

3.3.1.2 Cryptanalysis

All of Madryga's operations are linear - there is no source of nonlinearity such as the DES S-boxes or other logical or arithmetic operations.

Madryga's worst flaw is that it does not exhibit the desired avalanche effect [GDC90] – in general changes in one byte induce changes in the previous byte of the state and in at most the next two. Therefore, if the block is too large, changes may not propagate sufficiently.

Also, Eli Biham (in a personal communication to Bruce Schneier, cf. [Sch96], Section 13.2) noticed that *"the parity of all the bits of the plaintext and the cipher text is a constant, depending only on the key. So, if you have one plaintext and its corresponding ciphertext, you can predict the parity of the ciphertext for any plaintext."*

Ken Shirrif [Shi95] proved that Madryga is susceptible to differential cryptanalysis. The key can be determined with as little as 5,000 chosen plaintexts, and 10,000 on average.

Alex Biryukov and Eyal Kushilevitz in [BK98a] improved this attack twice. Firstly, with a differential attack requiring only 16 chosen-plaintext pairs. Secondly, they turned it into a ciphertext only attack, with only reasonable assumptions on the statistical distribution of the bytes in the plaintext. The attack requires only 2^{12} ciphertexts.

3.3.1.3 Intellectual Property

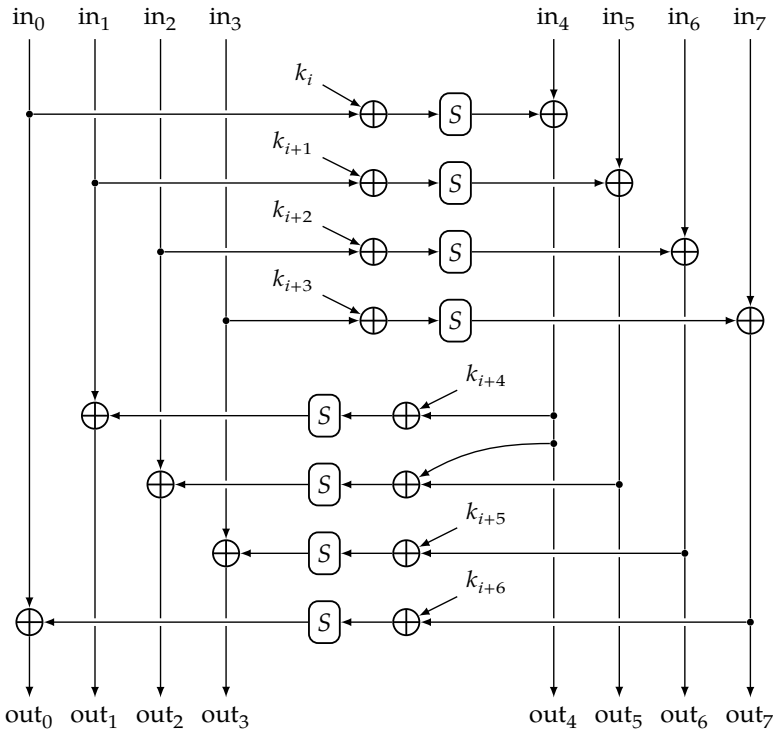
We are not aware of patents on Madryga. However, since the cipher was disclosed in 1984, any patent would have already expired.

3.3.2 NewDES

Robert Scott designed NewDES in 1985 as a possible replacement for DES [Sco85]. It operates on 64-bit blocks of plaintext with a 120-bit key. The state is divided into 8 one-byte blocks and the key into 15 one-byte subkeys.

The cipher has 17 equal rounds and no initial or final permutation. The subkeys are considered

Figure 3.4: A Cycle (Two Rounds) of NewDES



cyclically and each round uses seven consecutive subkeys.

A cycle, i.e. two rounds of NewDES are depicted in Figure 3.4: The 17th round consists in just the upper half of the cycle represented in the Figure. The cipher can be viewed as a Feistel network on two 32-bit branches with two alternating, different F-functions.

It has a very simple structure. The eight blocks of the state are divided into two sets of four. During a round one set of four blocks is transformed non-linearly and XORed to the other set of four blocks – then the roles are reversed and the second set of blocks is used to transform the values of the first set (but the two halves are not perfectly symmetrical). Hence, there are eight non-linear transformations per cycle, four per round. The non linear transformation consists in XORing the input with a subkey (in seven cases) or with another block of the state (in one case) and feeding this value into an 8-bit S-box. The values of the 8-bit S-box have been derived form the Declaration of Independence – and therefore it's a set of "nothing up my sleeve numbers," not a cryptographically designed S-box.

The first 8 cycles use each 7 subkeys alternatively, and the last round uses 4 subkeys. Thus, each byte of the 120-bit key is used exactly 4 times. A similar key schedule is used in the Bielorussian cipher Bel-T (Section 3.33 on page 212).

The designer showed that NewDES exhibits the full avalanche effect after seven rounds: every ciphertext bit depends on every plaintext bit and key bit. NewDES has the same complementation property that DES has: namely, that if $E_K(P) = C$, then $E_{\overline{K}}(\overline{P}) = \overline{C}$, where $\overline{x}$ is the bitwise complement of x.

The cipher can be made quite fast in software, because of its byte-oriented structure. It can be

implemented either compactly or efficiently in hardware as well, because it uses just a small amount of different operations in a fairly regular pattern, but at the same time up to four state block transformations can be easily parallelised. The same parallelism can be exploited in SW on 16- or 32-bit architectures.

3.3.2.1 Cryptanalysis

Only a small amount of cryptanalysis has been published on NewDES. The complementation property makes brute force attacks faster by a factor of 2.

Bruce Schneier reports in [Sch96], Section 13.3, a few observations by Eli Biham: changing a full byte in all the key and data bytes leads to another complementation property, reducing the complexity of bruce force attacks by a factor 2^8; a related-key attack can break NewDES with 2^{33} chosen-key chosen plaintexts.

John Kelsey, Bruce Schneier, and David Wagner's related-key cryptanalysis [KSW97] breaks NewDES with 2^{32} known plaintexts and one related key.

When informed of this attack, Scott modified the NewDES key schedule to resist rotational related key cryptanalysis. The new key schedule starts with the 15 bytes of the key $k_0, k_1, \ldots, k_{14}$, and then instead of repeating them, it first XORs all the bytes of the sequence first with k_7, then k_8, and finally k_9. The resulting cipher is called NewDES-1996. However, in [KSW97] it is shown that NewDES-1996 can be completely broken with 24 related-key probes and 530 chosen plaintext/ciphertext queries.

3.3.2.2 Remarks

The design is interesting, however it has some obvious weaknesses:

- The S-box displays poor linear and differential properties; and
- Diffusion is slow as, in the current form, full diffusion is only achieved after 7 rounds.

3.3.3 KeeLoq

KeeLoq is a proprietary block cipher designed by Gideon Kuhn for south african company Nanoteq Pty Ltd in the mid 80's It was sold to Microchip Technology Inc in 1995. It is specially designed for compact implementation in hardware. Based on Kuhn's work on self-synchronising stream ciphers [Kuh88], Keeloq is an unbalanced Feistel cipher based on a NLFSR (non-linear feedback shift register). KeeLoq was meant for lightweight HW implementations [KBS90] and is still used in many remote keyless entry systems by several car manufacturers.

KeeLoq accepts 64-bit keys and encrypts 32-bit blocks by executing its single-bit NLFSR for 528 rounds. An important component in the feedback function is the non-linear function $\mathfrak{n}$ (often named after the hexadecimal value $\texttt{3A5C742E}_x$) that is given as

$$\mathfrak{n}(a,b,c,d,e) = d \oplus e \oplus ac \oplus ae \oplus bc \oplus be \oplus cd \oplus de \oplus abc \oplus abd \oplus ace \oplus ade$$

where a, b, c, d and e are bits number 1, 9, 20, 26 and 31 of the NLFSR state during encryption and bits number 0, 8, 19, 25 and 30 during decryption. This function is specified by a table in

the original specifications. The actual NLFSR feedback function is given as

$$F(a,b,c,d,e,x,y,z) = \mathfrak{n}(a,b,c,d,e) \oplus (x \oplus y \oplus z)$$

where x and y are bits 0 and 16 of the NLFSR on encryption and bits 31 and 15 on decryption, and z is a key bit (bit 0 of the key state on encryption and bit 15 of the key state on decryption. The key schedule is simple: the key is copied into a 64-bit register that is then rotated one bit to the left each round.

Andrey Bogdanov [Bog07a] points at three fundamental weaknesses of the cipher:

- The key schedule is periodic, allowing the use of sliding techniques;
- The blocks are just 32 bits; and
- Efficient linear approximations of $\mathfrak{n}$ exist.

This allows him to present various attacks, culminating in an attack that needs only time 2^{37} and using 2^{32} 32-bit known plaintexts [Bog07b]. Soon hereafter, the complexity of the attacks was brought down to just 2^{28} for about 30% of the key space (also using 2^{32} known plaintexts) by Nicolas Courtois, Gregory V. Bard and David Wagner in [CBW08]

The cipher can also be broken using side channel analysis with only ten power traces. The attack described in [EKM+08] allows efficient recovery of both the secret key of a remote transmitter and the manufacturer key stored in a receiver in just a few minutes, permitting practical the cloning of remote controls.

The KeeLoq system is also susceptible to replay attacks.

3.3.3.1 Intellectual Property

Implementations of the KeeLoq cipher and several systems using it are heavily protected by patents, starting with the U.S. Patent 5,517,187, protecting its implementation in a microchip, which is the IP sold by Nanoteq to Microchip Technology Inc in 1995.

3.3.4 FEAL

FEAL (the Fast data Encipherment ALgorithm) is a 64-bit block cipher first published in 1987 by Akihiro Shimizu and Shoji Miyaguchi from NTT [SM87]. FEAL is a Feistel design. The initial version, now called FEAL-4, was a four-round cipher and used a 64-bit key size. There are options for including parity bits in the key. The cipher was quickly broken: FEAL was first extended to 8 rounds, then to arbitrarily many rounds, and a 128-bit key option was added. Eli Biham and Adi Shamir show in [BS91c] that variants with fewer than 31 rounds can be broken. NTT is still using the cipher: The current version is called FEAL-NX where the even integer $N \geqslant 32$ is the number of rounds, and X means that the 128-bit key contains no parity bits.

As a Feistel network, the encryption path of FEAL is quite standard, but the cipher has a few interesting historical aspects:

- Apart from DES-X (Subsection 3.2.9 on page 132), it is the oldest cipher we are aware of to use key whitening to improve its strength.

Figure 3.5: The FEAL F-function

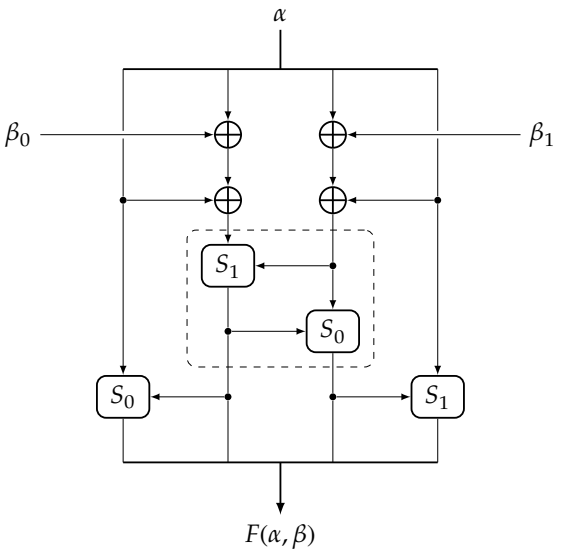
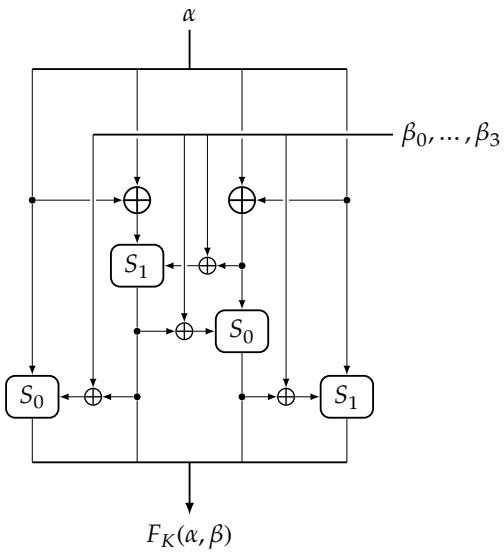


Figure 3.6: The FEAL FK-Function



- The structure of the FEAL F-function, depicted in Figure 3.5, can be viewed is a precursor of the IDEA round function (Section 3.6 on page 142 and in particular Figure 3.9 on page 143): Each half of the input is "compressed" into smaller chunks that are input to a "confusion device" inspired by a Hadamard transform; the confused output is then applied to the rest of the input. In the Figure, α is a 32-bit Feistel branch, whereas β is a 16-bit round key.
- Instead of using S-boxes, non-linearity is achieved by combining the XOR with modular addition. In fact, non linear parts of the F-function just consist in the two functions $S_0(a, b) = ((a + b) \bmod 256) \lll 2$ and $S_1(a, b) = ((a + b + 1) \bmod 256) \lll 2$.
- The key schedule is similar to a Matsui-like Feistel network (cf. Figure 1.4 on page 31 (b)), where one half of the key is "encrypted" by using the other half as the key. The F-function of the key schedule, called the FK-Function, is similar to the F-function of the cipher: it is depicted in Figure 3.6. The main difference w.r.t. the F-function is that the 32-bit "round key" β is split into four bytes that are each XORed with one of the inputs to the S_0 and S_1 functions. Also, intermediate values of the key scheduling process are saved to be reused at later stages of key scheduling.

We note that a XOR and a modular addition share several bits with a strong bias, and this alone is not sufficient to guarantee the hardness of a cipher. This is what ultimately has killed FEAL.

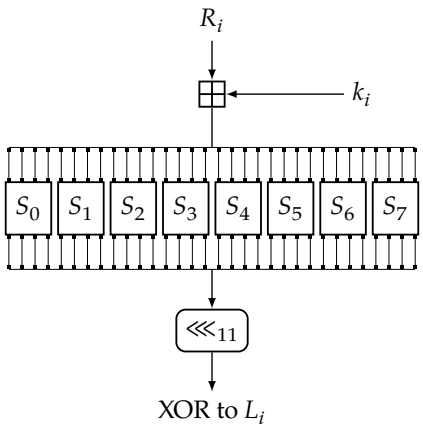
3.3.5 Intellectual Property

U.S. Patent 4,850,019 covered aspects of the cipher.

3.4 The GOST Block Cipher

GOST (Russian: ГОСТ) is a set of technical standards originally developed by the government of the Soviet Union as part of its national standardization strategy, now maintained by the *Euro-Asian Council for Standardization, Metrology and Certification* (EASC), a regional standards

Figure 3.7: The GOST Round Function



organization operating under the auspices of the Commonwealth of Independent States (CIS). GOST is an acronym of *gosudarstvennyy standart* (Russian: государственный стандарт), which simply means *state standard*.

Standard GOST 28147-89 defines several cryptographic algorithms, among them an elliptic curve based signature scheme and a block cipher. GOST 28147-89 is obligatory to use in the Russian Federation in all data processing systems providing public services. The original description of the algorithm is available here (in russian). A description in english of the encryption, decryption, and MAC algorithms is found in RFC 5830.

In the rest of this document GOST simply denotes the block cipher.

Developed in the 1970s as a Soviet alternative to the US standard algorithm DES and originally classified as "top secret," the GOST Block Cipher was standardized in 1989, downgraded to "secret" the following year, and finally declassified and disclosed in 1994.

GOST has a 64-bit block size and a key length of 256 bits. GOST is a balanced Feistel network of 32 rounds. The round function, depicted in Figure 3.7, is very simple: Let L and R be the left and right 32-bit halves of the input to a round; add a 32-bit round key modulo 2^{32} to R; apply a layer of eight 4-bit S-boxes; and rotate the result thereof left by 11 bits. The result of that is the output of the round function, which is then XORed to L. Then, as in DES, R and L are swapped.

The S-boxes of GOST are not fixed and for a specific application a new set of S-boxes can be chosen. Also, they can be public or secret, and contain about 354 ($\log_2((16!)^8)$) bits of information. If they are secret, the total amount of secret material in the cipher is thus 610 bits.

The key schedule is very simple. The 256-bit key is broken into eight 32-bit subkeys, and each subkey is used four times in the algorithm; the first 24 rounds use these subkeys in order, the last 8 rounds use them in reverse order. This broken symmetry allows GOST to eschew slide attacks.

3.4.1 Remarks

GOST is very similar to DES, however there are some significant differences:

(a) It is one of the first widely deployed ciphers to use 4-bit S-boxes.

(b) Since S-boxes can be chosen for specific applications, an implementation using just one S-box could be effectively implemented in SW using bit-slicing.

(c) It uses a simple rotation instead of a more complicated permutation, does not have an expansion permutation. The consequence is that the avalanche effect is slower. This is offset by the larger number of rounds.

(d) Alex Poschmann et al. [PLW10] revisit GOST and observe that its design makes it ideal for low gate count HW implementations and good throughput. They use a single S-box (the same S-box as PRESENT, Section 3.29 on page 204), following recent design trends to use just one good S-box instead of several random(ish) ones. Their performance results are reported in Table 4.1 on page 235, where both GOST implemented with eight different S-boxes as used by the Central Bank of Russian Federation (GOST-FB), and GOST with the PRESENT S-Box eight times (GOST-PS) are measured.

It is also one of the earliest ciphers to combine bitwise XOR and modular addition – two mutually non-linear operations. This is a fundamental aspects of many subsequent cipher designs.

3.4.2 Cryptanalysis

Until about 2010 the cipher was considered very secure, and thus a good alternative to AES-256. However, since 2010 several attacks have been published. The best undisputed cryptanalysis is a differential attack what breaks the cipher with complexity 2^{179} [Cou12a] whereas with the controversial XSL attack [CP02b] Nicolas Courtois claims that it is possible to attack GOST with heuristic complexity 2^{100} (the time complexity of attacking AES-256 with the same methods is claimed to 2^{101}). See also [Cou12b, Cou13].

We mentioned that the GOST S-boxes are free to be be chosen for any specific application and also kept secret. Their entropy is approximately 354 ($\log_2((16!)^8)$) bits, so the effective key size could, in theory, be increased to 610 bits; However, some care is necessary in the implementation and protocols to avoid attacker's access to an oracle where he can set a zero key. Under this assumption, Markku-Juhani Saarinen has shown how to mount an attack (cf. Subsectionsubsec:GOST-sbox-recovery) that recovers the contents of all the S-boxes in time 2^{32}. Therefore the secrecy of the S-boxes does not increase the strength of the cipher in several scenarios.

3.4.3 Intellectual Property

We are not aware of any patents on the GOST block cipher.

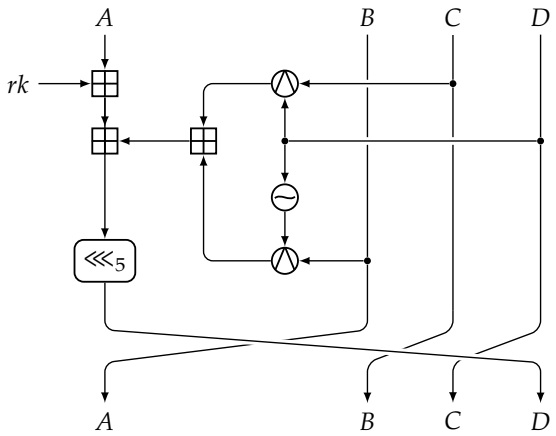
3.4.4 Advantages and Disadvantages

It is a fast and small cipher, however there are concerns about its security.

3.5 RC2

RC2 is a block cipher designed by Ron Rivest in 1987. Its development was sponsored by Lotus, who were seeking a custom cipher that, after evaluation by the NSA, could be exported as part of their Lotus Notes software. The NSA suggested a couple of changes, which Rivest incorporated.

Figure 3.8: The RC2 MIX Function



After further negotiations, the cipher was approved for export in 1989. Along with RC4, RC2 with a 40-bit key size was treated favorably under the US export regulations for cryptography at the time.

Initially, the details of the algorithm were kept secret and proprietary, but on 29 January 1996, source code for RC2 was anonymously posted to the Internet on the Usenet forum, `sci.crypt`. Eventually, in March 1998 Ron Rivest authored RFC 2268, publicly describing RC2 himself.

RC2 is a 64-bit block cipher with a variable size key. The cipher uses a byte substitution table whose elements are derived from the expansion of π, but the table itself is used only to expand the given key to 128 bytes, and not in the data obfuscation part.

The 18 rounds of the cipher are arranged as a source-heavy Feistel network, with 16 rounds of one type (MIXING) and two rounds of another type (MASHING). A MIXING round consists of four applications of the MIX transformation shown in Figure 3.8. The ADD used to combine the outputs of the two ANDs can be replaced with a XOR or an OR.

A MASHING round consists of addition of round key words to all words of the state, the round key words chosen by using some bits of the state itself as an index. The rounds are ordered as follows: five MIXING rounds, one MASHING round, six MIXING rounds, one MASHING round, and five MIXING rounds.

3.5.1 Cryptanalysis

RC2 is vulnerable to a related-key attack using 2^{34} chosen plaintexts [KSW97], however this is a related-key attack and not applicable to all scenarios.

Lars R. Knudsen, Vincent Rijmen, Ronald Rivest and Matthew Robshaw found that RC2 seemed to provide resistance to differential cryptanalysis [KRRR98]. Later, the CRYPTREC evaluation of RC2 [CRY01, CRY03] showed how a differential attack could recover the secret key using about 2^{60} chosen plaintexts – independently of the key length. Thus, for key sizes bigger than 64, there is an attack on RC2 which, is faster than an exhaustive search for the key.

Since today one expects about 80 bits of security as a minimum from a block cipher, even a lightweight one, we conclude that RC2 does not offer sufficient security.

3.5.2 Advantages and Disadvantages

The cipher is fast and has low latency [HMPM05], but its security level is insufficient for today's requirements.

3.5.3 Intellectual Property

The name RC2 is a trademark, registered by RSA Data Security Inc. We are not aware of any patents on the RC2 algorithm (the algorithm was originally protected as a trade secret).

3.6 IDEA

James Massey and his PhD student Xuejia Lai designed the International Data Encryption Algorithm (IDEA) on behalf of the Swiss company Ascom Tech AG and published its details in 1991 [LM90]. The primary reference for IDEA is Lai's PhD Thesis [Lai92].

IDEA is an improvement on a previous cipher by the same authors, the *Proposed Encryption Standard* (PES) [LM90]. Lai and Massey, together with Sean Murphy, showed that differential cryptanalysis could be used to recover PES keys [LMM91]. PES was then corrected, and the resulting cipher, initially called IPES (Improved PES), later was renamed to IDEA. The name IDEA is a trademark. The cipher was patented, but the patents are now expired.

IDEA is used in Pretty Good Privacy (PGP) v2.0, and was incorporated after the original cipher used in v1.0, BassOmatic, was found to be insecure. IDEA is an optional algorithm in the OpenPGP standard. It is also used for Pay-TV applications.

IDEA is an iterative block cipher, and it is one of the oldest public designs of an iterative SPN that is not a Feistel network, in fact this design has its own name, the Lai-Massey Design, after the names of the architects of the cipher (cfr. Section 1.5 on page 36).

IDEA operates on 64-bit blocks using a 128-bit key. It consists of a series of eight identical rounds, depicted in Figure 3.9 on the facing page, followed by an output transformation called the "half-round" and depicted in Figure 3.10. Each 64-bit block is split into 4 16-bit fields and all operations are performed on 16-bit values, with a high level of parallelism. At the end of each round two of the 16-bit fields are swapped. The half-round starts by undoing the swap at the end of the previous round, which in practice is just optimized away.

A full round is composed of two parts. The first part is a key mixing half-round. The second part is a 32-to-32-bit Multiplication-Addition Box (MA Box) bracketed by XOR operations to first reduce the input from 64 bits to 32, and then to spread the output of the MA Box on the whole 64-bit block.

Each round uses six 16-bit subkeys, while the half-round uses four, a total of 52 for 8.5 rounds. The first eight subkeys are extracted directly from the key, with k_1 from the first round being the lower 16 bits, and k_2 from the second round being the upper 16 bits. Further groups of eight subkeys are created by rotating the main key to the left 25 bits before repeating the same round key extraction procedure.

IDEA has a successor, IDEA NXT (Section 3.23 on page 193), that was originally called FOX.

Figure 3.9: A Round of IDEA

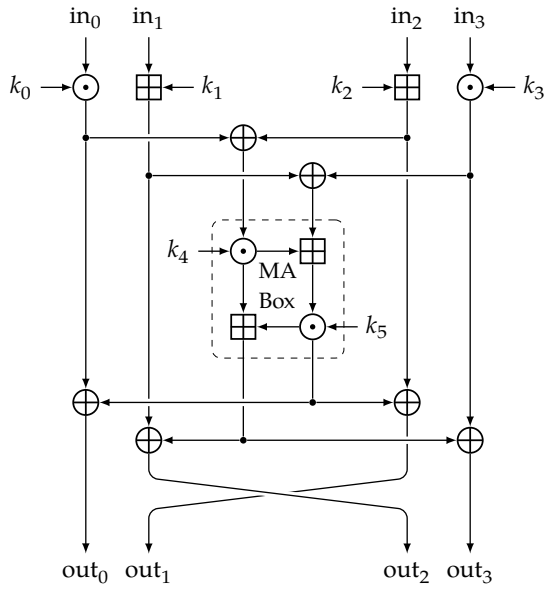
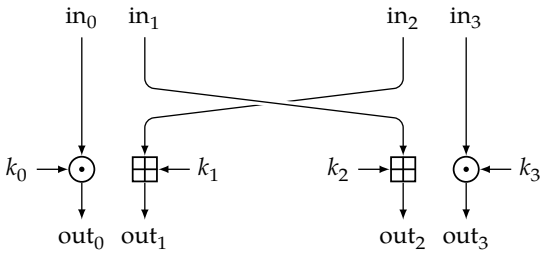


Figure 3.10: IDEA's Final Half-Round



$\odot$ denotes multiplication modulo $2^{16}+1$, where the zero value represents 2^{16}.

$\boxplus$ and $\oplus$ denote addition modulo 2^{16} and bitwise XOR, respectively.

3.6.1 Design Principles

- IDEA's design intends to mitigate the slower diffusion typical of Feistel networks while at the same time keeping the latter design's advantage of using the same data obfuscation path for both encryption and decryption. However, this is achieved at the price that there is linear function of the state that is invariant upon application of the round function. This is solved by applying a simple state permutation and non-linear key mixing.

- IDEA achieves non-linearity by combining different operations on mutually "incompatible" algebraic structures. Indeed MA stands for multiplication-addition where multiplication is in the multiplicative group of the integers modulo $2^{16}+1$ and addition is modulo 2^{16}. A third operation used in the cipher is bitwise XOR. Any two of these three operations do not satisfy any distributive or associative law.

  This incompatibility eliminates any exploitable algebraic property thus making it very difficult – if not infeasible – to solve the cipher algebraically.

- Modular multiplication produces huge mathematical complexity while consuming very few clock cycles on modern processors. It thus greatly contributes to security and efficiency of the cipher. The use of the modulo $p = 2^{16}+1$ is very ingenious, since being p prime, modular multiplication by a fixed value in the set $\mathcal{S} = [1, .., p-1]$ is a biijection on the set of values in the same set $\mathcal{S}$ – and all the values in the set are represented in just 16 bits by using the zero value to represent $p - 1 = 2^{16}$.

- All operations and values depend on the input, the secret key, and the choice of register sizes – no fixed constants are combined with the input, not even "nothing up my sleeve numbers."

- Key schedule is kept very simple, leaving the burden of the confusion of the key bits mostly to the data obfuscation path.

3.6.2 Cryptanalysis

The key schedule is the main weakness of the cipher since keys with too many zeros and ones or long repeating patterns lead to predictable modular multiplications. As a consequence several classes of weak keys have been identified, that reduce the security somewhat. However, the cipher per se is not broken. Until recently, the best cryptanalytic results so far just shave one bit of security out of a reduced 6-round version of the cipher [BDK07a] (see also [BNPV02, NPV03, BDKS11]). At Eurocrypt 2012 Dmitry Khovratovich, Gaëtan Leurent, and Christian Rechberger [KLR12] presented an important breakthrough in IDEA cryptanalysis: they break the first six rounds with memory 2^{41} and time $2^{118.9}$, 7.5 rounds with memory 2^{52} and time $2^{123.9}$, and full IDEA with memory 2^{52} (2^{59}) and time $2^{126.06}$ ($2^{125.97}$).

On the occasion of the expiration of the European patent protecting IDEA, Pascal Junod wrote in his blog (http://crypto.junod.info/2011/05/) :

> *IDEA is really an amazing block cipher and definitely deserves a seat in the Crypto Hall of Fame. [...] One salient feature of the IDEA block cipher is that, despite its (too) simple key-schedule, it has withstood 20 years of intense cryptanalysis, and IDEA is therefore a prominent counter-example to Shamir's law ("A cipher is generally broken after 13 years"). In summary, IDEA remains a very nice piece of engineering!*

3.6.3 Advantages

IDEA leads to very compact and quite fast SW implementations. See for instance "IDEA in 448 bytes of 80x86" at http://cypherspace.org/adam/rsa/idea.html.

3.6.4 Disadvantages

- IDEA has a large class of weak keys.
- It requires separate HW or at least considerable additional resources to support encryption and decryption.
- Decryption is often slower or requires the use of a large precomputed table, mostly because of the key schedule. Whereas the key schedule can be run in parallel with encryption, about 2/3 of the round keys must be inverted modulo $2^{16} + 1$.
- Whereas in SW it is possible to attain very good performance, the multiplication unit takes significant area in HW, and other ciphers rely on more economical ways to attain non linearity.

3.6.5 Intellectual Property

Ascom Tech AG, the owned of IDEA, and the Kudelski group later created the MediaCrypt joint venture in November 1999, to whom the rights of IDEA were transferred. MediaCrypt was tmerged with Nagravision S.A. in November 2006.

The name IDEA is trademarked. The following patents covered aspects of IDEA: U.S. Patent 5,214,703, EU Patent EP0482154, and Japan Patent JP322544B2.

Figure 3.11: A Round of MESH-64

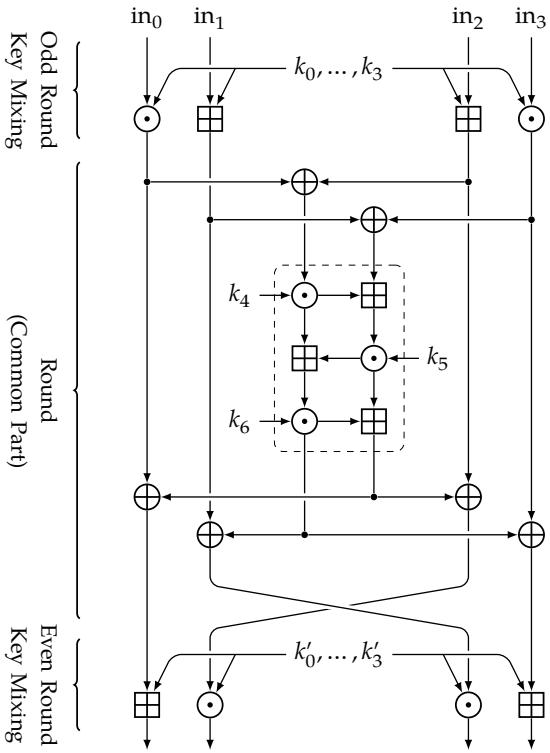
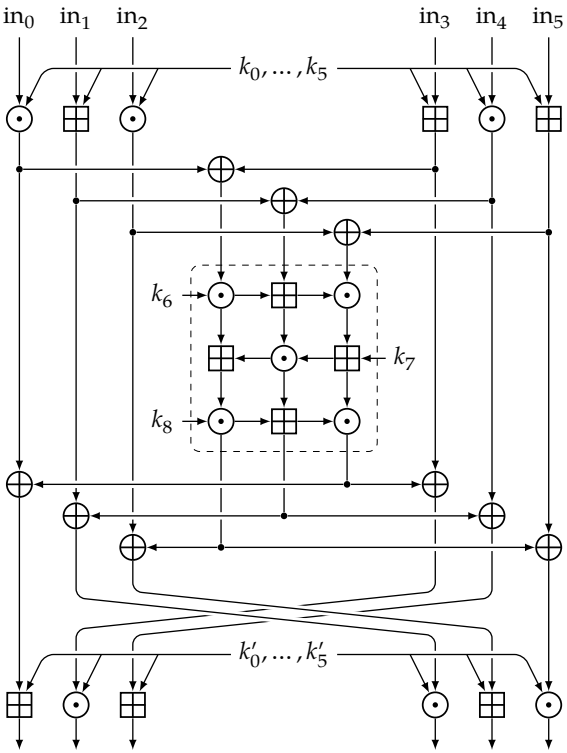


Figure 3.12: A Round of MESH-96



3.6.6 MESH

Jorge Nakahara Jr., Vincent Rijmen, Bart Preneel and Joos Vandewalle designed MESH, a family of ciphers strongly inspired by IDEA [JRPV03]. The main differences are the variable block size (64, 96 and 128 bits) and the larger MA-boxes: just as the IDEA MA-box combines multiplications and additions in a 2×2 checkerboard pattern, the MESH MA-boxes combine the same operations in a similar way, but in larger grids. Figure 3.11 and Figure 3.12 represent rounds of MESH-64 and MESH-96, respectively. MESH-128 is similar to MESH-96, but it has eight branches instead of six, and the MA-box is a 4×4 addition/multiplication checkerboard instead of 3×3.

The even rounds differ from the odd rounds only in the order of multiplications and additions for the key mixing in the first "row" of the round. The ciphers, like IDEA, undo the branch permutation and perform a final key mixing in the last round. The number of rounds of MESH-64, MESH-96, and MESH-128 is 8.5, 10.5 and 12.5 respectively.

The key schedule of MESH is more complex than that of IDEA, in order to prevent weak keys: First, several constants c_i are generated as powers of an element of $\mathbb{F}_{2^{16}}$; The first eight 16-bit subkeys are just obtained by XORing the 16-bit words of the master key with the first eight constants; Each successive subkey is obtained by a recursive function that combines XOR, addition modulo 16, a fixed cyclic shift and addition of a constant c_i. A weakness that the cipher shares with IDEA is that multiplicative inverses of several subkeys are required.

Jorge Nakahara Jr., Bart Preneel and Joos Vandewalle prove in [JPV04] that attacks that have

been proved effective against IDEA are less effective against MESH. There are currently no attack on the full MESH ciphers.

For more details we refer to the paper.

3.7 The CAST Family

The name of the CAST block cipher family comes from initials of the two authors: Carlisle Adams and Stafford Tavares. During the late '80s and early '90s Adams and Tavares did considerable research in various aspects of block cipher design [AT89, AT90a, Ada90, AT93], culminating in 1992-93 with the first ciphers to be used in Bell-Northern and Entrust Technologies products: CAST-1 and CAST-2 [Ada98]. Between 1993 and 1995 CAST-3 and CAST-4 were developed. CAST-3 introduced refinements to the key schedule (cf. [Ada94]), and CAST-4, with refinements in S-box construction, is the direct precursor of CAST-5, known also as CAST-128.

The design procedure is described in a paper by Carlisle Adams [Ada97], that is also available online at http://cryptome.org/jya/cast.html and derives from Carlisle Adams's Ph.D. Thesis [Ada90].

The design strategy was to ensure that the ciphers possessed certain desirable cryptographic properties such as avalanche [Fei73, FNS75], Strict Avalanche Criterion (SAC) and Bit Independence Criterion (BIC) [WT85][1], and an absence of weak and semi-weak keys [JRS88, Cop85, MS86a, MS86b].

The CAST ciphers were the first to use large S-boxes to allow the F-function to have ideal avalanche properties, and to use bent functions in the S-box columns.

In our presentation we shall skip the first four ciphers of the family and present its two most mature members: CAST-128 and CAST-256.

3.7.1 CAST-128

Introduced in 1996, CAST-128 is the fifth member of the CAST family, but also its first widely known design.

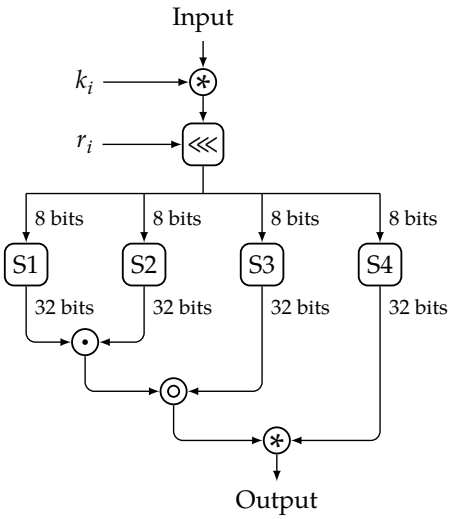
The block size is 64 bits, key sizes range from 40 to 128 bits. The structure is a Feistel network with 12 or 16 rounds. Components include large $8 \times 32 - bit$ S-boxes based on Bent functions and a construction due to Kaisa Nyberg [Nyb91] (see also [AT90b]) a variable key dependent rotation (similarly to RC5) and the use of modular addition and subtraction, and XOR operations. The three F-functions are similar to each other, and differ only in the order of the functions used to mix the round key and the outputs of the S-boxes, as shown in Figure 3.13.

CAST-128 is designed for efficient SW implementation on 32-bit architectures.

The key schedule determines both the round key k_i and the 5-bit rotation key r_i. It is similar to the TEA key schedule in that it uses magic constants that are repeatedly added to parts of the secret key, but also processes these results through the F-functions. The rotation keys are an arithmetic progression modulo 32.

[1] The **bit independence criterion** states that output bits j and k should change independently when any single input bit i is inverted, for all i, j and k.

Figure 3.13: The CAST F-function
The triple $(\circledast, \odot, \circledcirc)$ of operations takes the three values $(\boxplus, \oplus, \boxminus)$, $(\oplus, \boxminus, \boxplus)$ and $(\boxminus, \boxplus, \oplus)$.



3.7.2 CAST-256

CAST-256 (also known as CAST-6), presented in 1998, doubles the block size to 128 bits and accepts keys ranging in lengths from 128 to 256 bits in 32-bit increments. It is described in RFC 2612, and was presented at the First AES Candidate Conference [Ada98].

The design is a four-branch Type 1 generalized Feistel network, with 32-bit branches. The same three variants of F-function of CAST-128 are used cyclically, so the same configuration of operations repeats every 12 rounds. The cipher consists of 48 rounds, but there is an elegant trick: The last 24 rounds are the inverse as the first 24 rounds, with only the key schedule being different. Therefore the same SW or the same HW can be used for both encryption and decryption.

This idea is similar to the design of the ENIGMA machine, where the second half of the encryption process is the inverse of the first half, and is also one of the key design choices in the recent cipher PRINCE (Section 3.35 on page 215).

3.7.3 Cryptanalysis

The ciphers of the CAST family have shown good resistance against various types of cryptanalysis. This is a consequence of their sound design, well founded on extensive theory.

In [WWCH10] differential cryptanalysis is used to attack 8-round CAST-128 with key sizes greater than or equal to 72 bits and 9-round CAST-128 with key sizes greater than or equal to 104 bits.

Currently, the best public cryptanalysis of CAST-256 [BLNW12] is a zero-correlation cryptanalysis breaking 28 rounds with $2^{246.9}$ time and $2^{98.8}$ data.

3.7.4 Advantages and Disadvantages

CAST-128 and CAST-256 offer decent performance, but not extraordinary. CAST-128 has a 64-bit block size, which for some applications may be viewed as too short.

The performance of CAST-256 is not impressive: Among all the ciphers submitted to the AES contest CAST-256 felt roughly in the middle, coming after Twofish, Rijndael, Crypton, E2, Mars and RC6 on several architectures [SKW+98a].

The non optimal key schedule also offers limited key agility.

3.7.5 Remark

In our opinion there is a missed opportunity for higher parallelism in CAST-256, as the designers could have chosen a Type-2 Feistel network.

3.7.6 Intellectual Property

The CAST-128 design, including the idea of using different F-functions, where the main mixing operations (addition and subtraction modulo 2^{32}, and bitwise XOR) are permuted, is patented by Entrust Technologies. U.S. Patent 5,825,886 A expires on 11/01/2015, and is also published as Canada Patent 2,164,768 A1. This covers the use of the same F-functions in CAST-256 as well.

We note that Entrust Technologies has generously released the algorithms for free use.

3.8 The SAFER Family of Ciphers

> *I have many times used the discrete exponential or the discrete logarithm as nonlinear cryptographic functions and they have never let me down.*
>
> James Massey, as quoted by Serge Vaudenay in [Vau94]

James Massey is the principal designer behind the SAFER family of block ciphers. The acronym SAFER means *Secure And Fast Encryption Routine.*

3.8.1 SAFER (S)K-64 and (S)K-128

The original version, SAFER K-64, introduced in 1993 [Mas93] had a 64-bit key size. One year later a 128-bit key version, called SAFER K-128, was introduced [Mas94]. Both versions have a block size of 64 bits. The development of these ciphers were sponsored by Cylink Corporation.

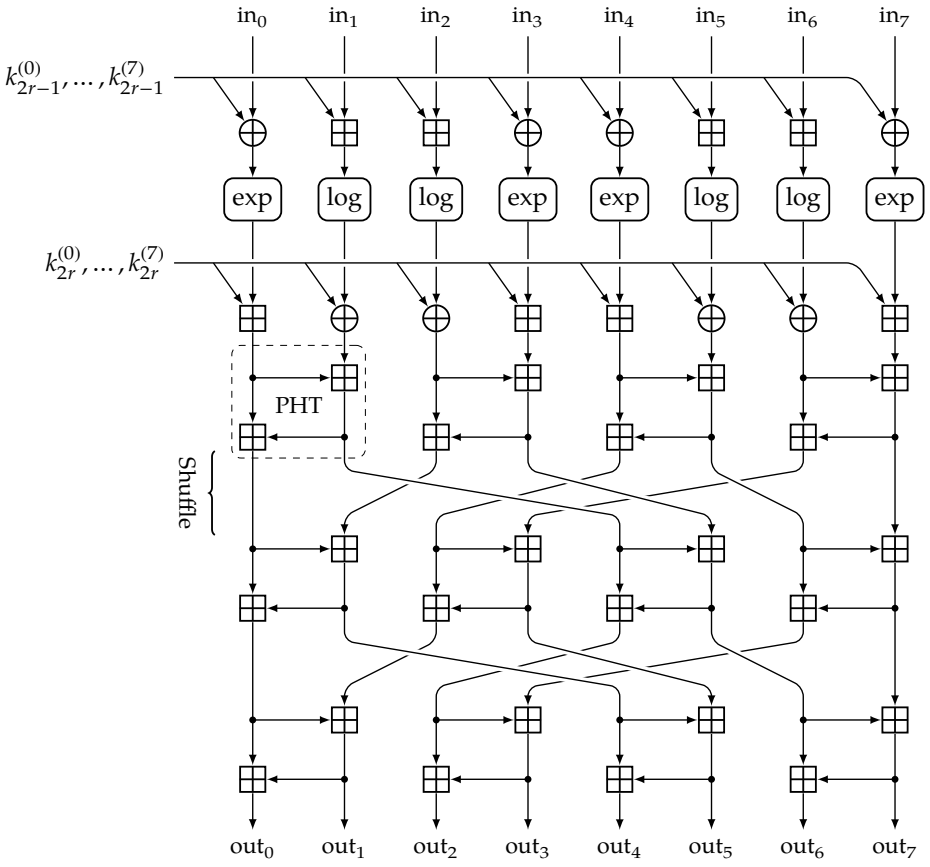
Similarly to IDEA, this is an iterative substitution-permutation design which is not a Feistel network.

The round function, represented in Figure 3.14 on the next page, consists of: key mixing, a substitution layer, a second key mixing, and three layers of pseudo-Hadamard Transformations (PHT) interleaved with state shuffles. The PHT can be rewritten as

$$L(x, y) = (2x + y \bmod 256, x + y \bmod 256) \ . \tag{3.1}$$

All values are just eight bits wide, including the subkeys $k_i^{(0)}, \ldots, k_i^{(7)}$. The PHTs serve to rapidly achieve the desired diffusion of small changes in the plaintext or the key over the resulting

Figure 3.14: A Round of SAFER
Keys with upper indexes (0) to (7) are fed to columns 0 to 7, respectively.



ciphertext. SAFER K-64 consist of six such rounds.

The exp operation denotes modular exponentiation of 45 to the input, modulo $257 = 2^8 + 1$, where the zero input is not admitted and the zero value represents 256 (modulo 257) – in other words $\exp(a) = (45^a \bmod 257) \bmod 256$.

The log operation is defined as the inverse of the exp operation, i.e. the discrete log modulo 257 to the base of 45, where $\log(0) = 128$ because the zero byte represents -1 modulo 257 in the exp operation.

With respect to IDEA the parallelism is increased, but lookup tables or hardwired S-boxes must be implemented.

We have mentioned in Subsection 2.1.10 on page 88 Kaisa Nyberg's result on this type of S-boxes. Her result implies that the exp S-box has differential uniformity 2 in $\mathbb{F}_{257}$, but a simple computer program shows immediately this S-box has differential uniformity 2 in $\mathbb{Z}/256\mathbb{Z}$ as well – and therefore it plays optimally with the dominant operation used in the cipher, namely the modular addition modulo 256.

3.8.2 Key Schedules

In the original version of SAFER K-64, the first 64-bit round key was the key itself. Successive round keys are generated by applying a circular left shift of three bits to each byte of the master key, and then XORing the result with fixed round constants. The round constants serve to address weaknesses in the IDEA key schedule, which consisted in simple bit field extraction from the master key.

The 128-bit key schedule for SAFER K-128 is similar to the original key schedule, with the keys coming alternatively from the two halves of the 128-bit key in such a way that, when the the two halves of the key are identical, the round keys coincide with those produced by the key schedule for SAFER K-64 – to guarantee backwards compatibility, in a way similar to the use of single-key triple-DES. This key schedule was proposed by the Special Projects Team of the Ministry of Home Affairs, Singapore.

Some weaknesses were found by Lars Knudsen [Knu95b, Knu00] and Sean Murphy [Mur98]. This lead to the revision of the key schedule, giving rise to SAFER SK-64 and SAFER SK-128, where the SK means *strengthened key schedule.* The announcement was posted to the USENET newsgroup sci.crypt.research by Lars Knudsen on behalf of James Massey [Knu95a]. The new key schedule, designed by Lars Knudsen, first augments the eight bytes of the 64-bit key by a ninth byte equal to the XOR of the first eight. Then, the nine individual bytes are cyclically rotated at each round as in the original key schedule, but then they are also cyclically permuted among each other. Finally, the first eight bytes are chosen and XORed with round constants to obtain the current round key.

The SK-128 key schedule works in the same way, separately and in parallel on the two halves of the 128-bit key.

3.8.3 SAFER+ and SAFER++

Two further variants were designed by Massey with the assistance of Armenian cryptographers Gurgen Khachatrian (American University of Armenia) and Melsik Kuregian. These variants have made changes to the main encryption routine, mostly in the way the various sub values are shuffled between the modular addition layers. We briefly describe these next:

- SAFER+, with block size of 128 bits, was submitted to the AES contest [MKK98].
  It is basically a "doubling" in size of SAFER-K64: it features the same alternance of "exp" and "log" S-boxes, sandwiched between two rounds of round key mixing based on modular addition and bitwise XOR, and four layers of PHTs (in place of three) interleaved with shuffles to guarantee diffusion.

  The shuffle at the basis of the design is the "Armenian shuffle", given by

  $$(8\ 11\ 12\ 15\ 2\ 1\ 6\ 5\ 10\ 9\ 14\ 13\ 0\ 7\ 4\ 3)$$

  (we decremented the original numbers to follow our zero-based indexing for bundles).
  The shuffles between the three PHT layers in the reference implementation submitted to the AES contest look, however different from each other, which may be a consequence of some optimisations.

  In the description of the cipher the matrix M corresponding to the diffusion layer is explicitly

given: with respect to SAFER's matrix (see Subsection 1.8.2 on page 44) there are no columns predominantly filled with elements divisible by higher powers of 2, therefore reducing the likelihood that an attacker can build useful linear or differential characteristics.

The key schedule is simple: first the XOR of all key bytes is appended to the key, then at each round the expanded key is rotate by 11 bits, and the first 16 bytes of the key are then extracted and XORed with fixed constant before being used as keys.

The cipher was not selected as a finalist.

Bluetooth uses custom algorithms based on SAFER+ for key derivation (called E21 and E22) and authentication as message authentication codes (called E1), but not for encryption. Apart from the use in Bluetooth, SAFER+ is deprecated, and superseded by SAFER++.

- SAFER++ [MKK00], submitted to the NESSIE project in two versions, with 64- and 128-bit block sizes.

  The main new feature of SAFER++ is a the use of a 4-point PHT transform in place of the 2-point PHT transform that was used previously in the SAFER family. A single 4-point PHT can be implemented with six modular additions, and replaces four 2-point PHTs, which require two modular additions each, leading to a speed up. Another consequence of the redesign of the diffusion layer is improved security: the corresponding matrix has most entries equal to 1 and a minority of entries equal to 2 or 4, with at most one 4 and at most five 2's, making even more difficult for an attacker to build useful characteristics.

  The key schedule is essentially the same as that of SAFER+.

  A round of SAFER++ is depicted in Figure 3.15.

3.8.4 Remarks

Generally speaking, the members of the SAFER family have not been successfully cryptanalysed in their full round versions. This type of design has proven quite robust.

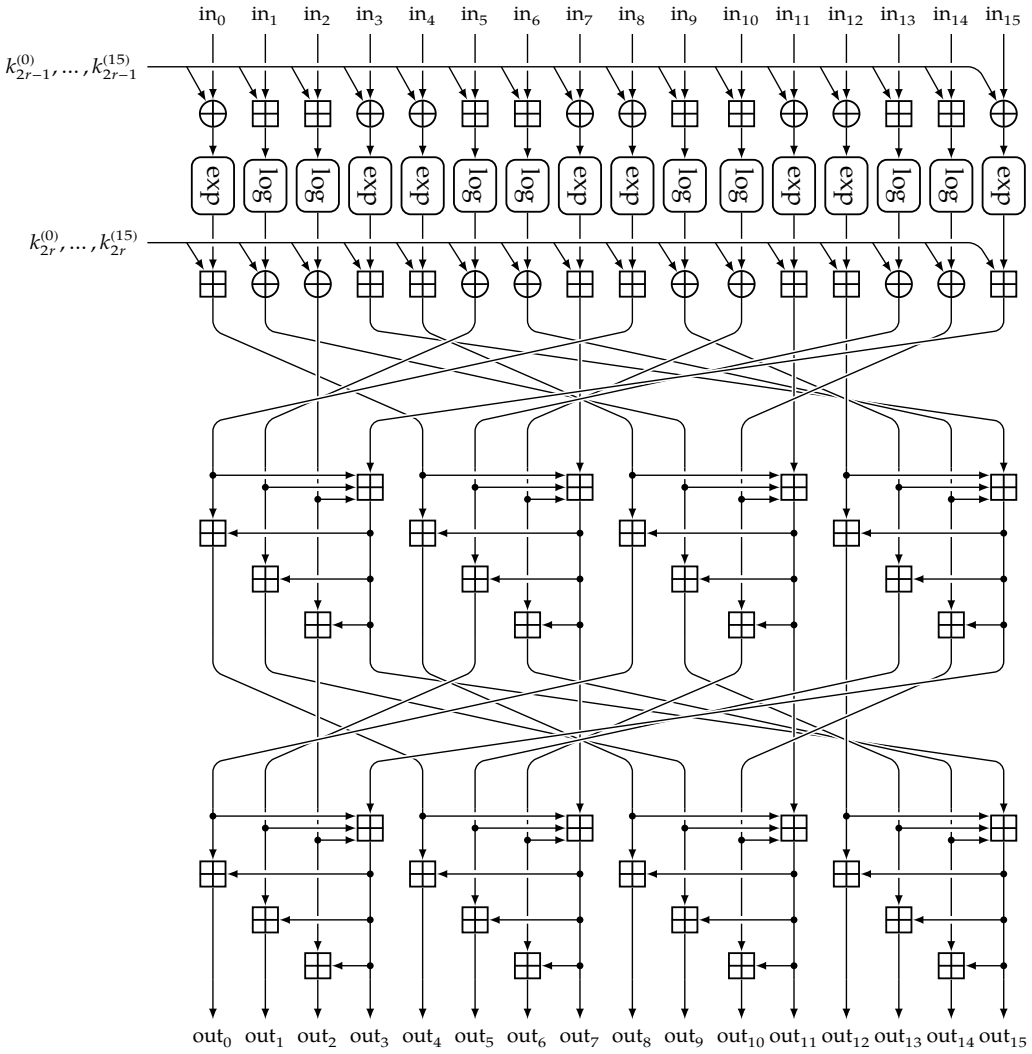
Despite the fact that SAFER dispenses with modular multiplications to use fixed S-boxes, it is noteworthy to observe that:

- There are just two different S-boxes – and this is probably motivated by the fact that, being the cipher not related to a Feistel-network, an S-box must be either self-inverting or there must be two of them, inverting each other. This is in stark contrast with contemporary cipher designs that use several S-boxes, such as DES, and predates more modern designs.

- These are simple S-boxes, that input and output the same number of bits, in other words they are not parametrized by additional input (such as in DES). This also predates recent lightweight and secure designs.

Bruce Schneier warned in 1996 against the use of SAFER: In [Sch96], Section 14.4, he wrote *"SAFER was designed for Cylink, and Cylink is tainted by the NSA,"* in turn citing as its source James Bamford and Wayne Madsen's book *The Puzzle Palace* [BM95]. He continues: *"I recommend years of intense cryptanalysis before using SAFER in any form."* In February 2003 Cylink was acquired by SafeNet, one of the largest suppliers of encryption technology to the United States Government.

Figure 3.15: A Round of SAFER++
Keys with upper indexes (0) to (15) are fed to columns 0 to 15, respectively.



3.8.5 Cryptanalysis

SAFER is a cipher that has withstood cryptanalysis very well, despite some problems in the key schedules and suboptimality of the diffusion layers. Indeed, Schneier's warnings may have been in vain. The robustness of SAFER (and of it variants) is probably to be attributed to the use of S-boxes with very good linear properties "on the right bits." This is exemplified by Serge Vaudenay's analysis of the original cipher.

As we mentioned in Subsection 1.8.2.1 on page 45, Serge Vaudenay in his treatment of multipermutations [Vau94] observes that the PHT in SAFER is not a multipermutation. This is used by Vaudenay to construct linear characteristics for the cipher. Denote by $L_i(x, y)$ the i-th output of $L(x, y)$. Observe that some information about $L(x, y)$, namely the *least* significant bit of the first output, only depends on y and not on x, more precisely $\langle L_1(x, y), 1\rangle = \langle y, 1\rangle$. By following how the least significant bit of the inputs to the "exp" S-box gets transformed through the cipher, Vaudenay derived conditions on the S-boxes for a linear characteristic to have sufficiently hight

probability to be used in a linear attack: The complexity of the attack is about $16/(2q)^{20}$ where q denotes the bias $\mathscr{P}[\langle x, 1\rangle = \langle \gamma(x), 1\rangle] - 1/2$ for a eight-bit S-box γ. Vaudenay's attack is faster than brute force for about 6.6% of all S-boxes (these are the S-boxes for which $|q| > 2^{-4}$), and for for only 9.9% of all S-boxes it is $q = 0$, which this is the case for Massey's exp S-box. This allows linear cryptanalysis of four-round reduced SAFER when the S-boxes are changed with highly biased ones.

SAFER is one of the very few cipher families where differences for the purpose of differential cryptanalysis are computed according to an operation different than the bitwise exclusive or: In Lars Knudsen and Thomas Berson's paper [KB96, Knu00], differences are computed bytewise with respect to subtraction modulo 256; their attack is also one of the few instances of truncated differential attacks in the literature.

Yupu Hu, Yuqing Zhang, and Guozhen Xiao in [HZX99] show some integral characteristics that have higher probability than the differential characteristics found so far, but no concrete attacks are presented.

The results of the attacks on the various members of the SAFER family are compared in Table 3.1 on the next page. The best attacks on SAFER++ break 5.5 rounds of the cipher (out of 7 and 10 for the key size of 128 and 256 bits respectively). They are due to Alex Biryukov, Christophe De Cannière and Gustaf Dellkrantz [BCD03a, BCD03b] for SAFER++128 and to Jingyuan Zhao, Meiqin Wang, Jiazhe Chen, and Yuliang Zheng [ZWCZ12a, ZWCZ12b] for SAFER++256. One of the weaknesses of SAFER++ exploited in the attack by Biryukov et al. is the following: If the (arithmetic) input difference to the linear layer is of the form

$$(a, b, c, d, a, b, c, d, a, b, c, d, a, b, c, d)$$

for any eight-bit values a, b, c, and d, then the output difference will be of the form

$$(x, y, z, t, x, y, z, t, x, y, z, t, x, y, z, t)$$

and the S-box structure will preserve this type of output – this means that only key mixing by XOR can disrupt this pattern. Also, the S-boxes of the SAFER are constructed using exponentiations and logarithms as described previously, and the following mathematical relations, observed by Lars Knudsen in [Knu00] hold:

$$\begin{aligned} \exp(a) + \exp(a + 128 \bmod 256) &= (45^a \bmod 257) + (45^{a+128} \bmod 257) \bmod 256 = 1 \bmod 256 \\ \log(a) - \log(1 - a \bmod 257) &\equiv 128 \bmod 256 \ . \end{aligned}$$

In particular, these relations allow Biryukov et al. to construct boomerangs.

3.8.6 Advantages

Like IDEA, SAFER does not combine its inputs with hardwired constants, except for the S-boxes, which are designed in a very restrictive mathematical fashion and thus should be beyond suspicion. It offers good performance, especially in software.

Table 3.1: Cryptanalysis of the SAFER Cipher Family – Published Results

Cipher / / Key Size	Rounds Attacked	Technique	Attack Complexity Data	Time	Memory	Keys	Reference
K / 64	5/6	Truncated diff. (CPA)	2^{36}	2^{61}	2^{32}	All	[KB96, Knu00]
K / 64	5/6	Truncated diff. (CPA)	2^{46}	2^{46}	2^{32}	All	[KB96, Knu00]
K / 64	5/6	Related-key trunc. diff.	2^{29}	2^{61}	2^{32}	All	[KSW96]
K / 128	5/6	Truncated diff. (CPA)	2^{39}	$2^{64.1}$	2^{24}	All	[KB96, Knu00]
SK / 64	5/6	Linear (CPA)	2^{58}	2^{80}	–	2^{-10}	[JPV00]
SK / 64	3.75/6	Impossible diff. (CPA)	2^{45}	2^{42}	2^{38}	All	[ZWY10]
+ / 128	2.75/7	Impossible diff. (CPA)	2^{64}	2^{58}	2^{104}	All	[JP03]
+ / 128	3.75/7	Impossible diff. (CPA)	2^{78}	2^{75}	2^{68}	All	[ZWY10]
+ / 128	4/7	Linear (CPA)	2^{98}	2^{160}	–	2^{-10}	[JPV00]
+ / 128	4/7	Impossible diff. (CPA)	$2^{122.4}$	2^{121}	$2^{87.4}$	All	[ZWCZ12a, ZWCZ12b]
+ / 256	4/7	Impossible diff. (CPA)	$2^{122.4}$	2^{216}	$2^{89.4}$	All	[ZWCZ12a, ZWCZ12b]
++ / 128	2.75/7	Impossible diff. (CPA)	2^{64}	2^{58}	2^{104}	All	[JP03]
++ / 128	3/7	Linear (KPA)	2^{33}	2^{121}	2^{32}	2^{-6}	[JPV01]
++ / 128	3/7	Integral (CPA)	2^{16}	2^{16}	2^{16}	All	[PQ03]
++ / 128	3/7	Linear (KPA)	2^{33}	2^{121}	2^{32}	2^{-6}	[JPV01]
++ / 128	3/7	Multiset (CCA)	2^{16}	2^{16}	2^{4}	All	[BCD03a, BCD03b]
++ / 128	3.75/7	Impossible diff. (CPA)	2^{78}	2^{66}	2^{62}	All	[ZWY10]
++ / 128	4/7	Integral (CPA)	2^{64}	2^{112}	2^{64}	All	[PQ03]
++ / 128	4/7	Integral (CPA)	2^{64}	2^{120}	2^{16}	All	[PQ03]
++ / 128	4/7	Multiset (CPA)	2^{48}	2^{70}	2^{48}	All	[BCD03a, BCD03b]
++ / 128	4/7	Boomerang (CP/ACC)	2^{41}	2^{41}	2^{40}	All	[BCD03a, BCD03b]
++ / 128	4.5/7	Multiset (CPA)	2^{48}	2^{94}	2^{48}	All	[BCD03a, BCD03b]
++ / 128	5/7	Impossible diff. (CPA)	2^{124}	2^{118}	2^{97}	All	[ZWCZ12a, ZWCZ12b]
++ / 128	5/7	Boomerang (CP/ACC)	2^{78}	2^{78}	2^{48}	All	[BCD03a, BCD03b]
++ / 128	5.5/7	Boomerang (CP/ACC)	2^{108}	2^{108}	2^{48}	All	[BCD03a, BCD03b]
++ / 256	3/10	Linear (KPA)	2^{33}	2^{121}	2^{32}	2^{-6}	[JPV01]
++ / 256	4/10	Integral (CPA)	2^{64}	2^{144}	2^{64}	All	[PQ03]
++ / 256	4/10	Linear (KPA)	2^{81}	2^{178}	2^{32}	2^{-13}	[JPV01]
++ / 256	4/10	Linear (KPA)	2^{91}	2^{167}	2^{32}	2^{-11}	[JPV01]
++ / 256	5.5/10	Impossible diff. (CPA)	2^{124}	2^{246}	2^{97}	All	[ZWCZ12a, ZWCZ12b]

3.8.7 Disadvantages

Encryption and decryption are different enough to require considerably more code or area to support both with respect to an implementation that supports only encryption.

3.8.8 Intellectual Property

Regarding SAFER, we quote Massey from [Mas94]: *"Although our design of SAFER K-64 was sponsored by Cylink Corporation (Sunnyvale, CA, USA), Cylink has explicitly relinquished any proprietary rights to this algorithm. This largesse on the part of Cylink was motivated by the reasoning that the company would gain more from new business than it would lose from competition should many new users adopt this publicly available cipher. SAFER K-64 has not been patented and, to the best of our knowledge, is free for use by anyone without fees of any kind and with no violation of any rights of ownership, intellectual or otherwise."*

At the First Advanced Encryption Standard Candidate Conference, the designers of SAFER+ stated *"Cylink relinquishes all proprietary rights to SAFER+ and consigns this algorithm to the public*

Figure 3.16: The Structure of Blowfish

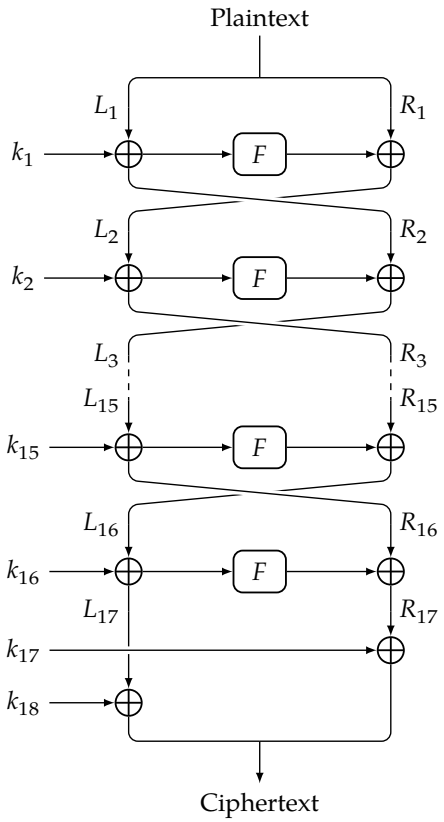
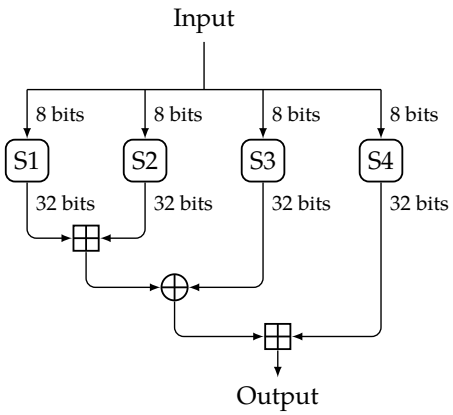


Figure 3.17: The Blowfish F-function



domain" [MKK98]. The NESSIE submission for SAFER++ [MKK00] contains the same sentence with SAFER++ replacing SAFER+.

3.9 Blowfish

Bruce Schneier designed the block cipher Blowfish in 1993 [Sch93] as a drop-in replacement for DES or IDEA. The cipher enjoys widespread usage. It has a 64-bit block size and a variable key length from one bit up to 448 bits. It is a 16-round Feistel cipher.

Notable features of the design include key-dependent wide S-boxes and a highly complex key schedule – similar to RC5 key expansion. The F-function resembles that of CAST family (Section 3.7 on page 146), but there are some important differences: the CAST S-boxes are fixed, whereas those of Blowfish are generated from the secret key, and the CAST F-function appears in three different variants that are used cyclically, whereas the Blowfish F-function has only one form.

The structure of Blowfish (without the details of the key schedule) in Figure 3.16 and its round function is depicted in Figure 3.17.

The key schedule is actually a key expansion procedure that turns 32 to 448 bits into a 576 bits array from which the 18 32-bit round keys $k_1, \ldots, k_{18}$ are extracted. The key schedule uses the Blowfish encryption primitive with the key expansion buffer initially initialized with digits of π (therefore these are "nothing up my sleeve numbers").

Blowfish provides a good encryption rate in software and no effective cryptanalysis of the full-round version has been disclosed to date.

3.9.1 Cryptanalysis

No no effective cryptanalysis of the full cipher has been found (or disclosed) to date.

Vincent Rijmen has shown that four rounds of Blowfish are susceptible to a second-order differential attack [Rij97]; Serge Vaudenay at FSE 1996 has proven that, for a class of weak keys, 14 rounds of Blowfish can be distinguished from a pseudorandom permutation [Vau96].

3.9.2 Advantages

Much faster than many other ciphers such as DES (Section 3.2 on page 127) and IDEA (Section 3.6 on page 142) once the key expansion has been performed.

Decryption and encryption are essentially the same circuit, since it is a canonic Feistel network.

It can be implemented with very compact code.

3.9.3 Disadvantages

Lack of internal parallelism reduces optimisation opportunities.

The key expansion runs the Blowfish algorithm 521 times, hence it takes about the same time as compressing a 4K block. For many purposes this is a deal-breaker.

3.9.4 Successors

Blowfish's has two more modern successors: Twofish (Section 3.13 on page 162) and Threefish (Section 3.30 on page 205). SPECK (Section 3.36 on page 220) borrows some aspects from Threefish.

3.9.5 Intellectual Property

At the time Blowfish was released, many other designs were proprietary, encumbered by patents or were secret. Schneier hence stated *"Blowfish is unpatented, and will remain so in all countries. The algorithm is hereby placed in the public domain, and can be freely used by anyone."*

3.10 RC5

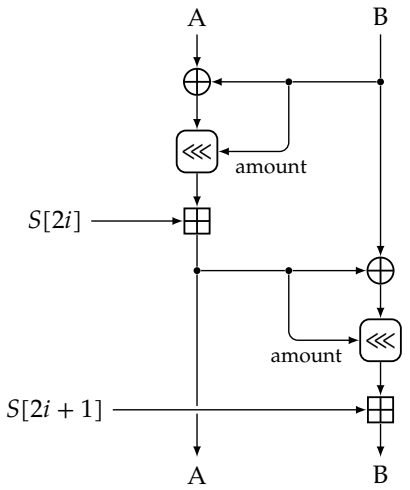
RC5, developed by Ron Rivest in 1994 and published in 1995 [Riv94], is a still widely used block cipher. It is notable for its extreme simplicity and flexibility.

Key sizes can vary from 0 to 2048 bits (with 128 being the most common value) and the block sizes can be 32, 64 or 128 bits (64 being the recommended value).

RC5 consists of three algorithms: key expansion, encryption, and decryption.

The encryption algorithm is an iterative cipher, where a round is repeated r times with $1 \leqslant r \leqslant 255$ ($r = 12$ being the originally suggested value). The design is a kind of Feistel network, where one round – depicted in Figure 3.18 on the facing page – can be split into two Feistel-like halves, each of which consists of XOR, a variable rotation and a modular addition.

Figure 3.18: A Round of RC5



The use of XOR and modular addition follows the principle of achieving non linearity by using mutual incompatible algebraic structures, whereas the variable rotation aims at providing unpredictable diffusion. The input that determines the amount of variable rotation is explicitly marked as such in the Figure. Note that $S[\,]$ is an array of *expanded keys*. The rounds are numbered from 1 to r. Before the first round, $S[0]$ and $S[1]$ are added to the two halves of the state. Hence, $t := 2r + 2$ round key words are required.

Decription is just encryption in reverse.

The AES submission RC6 (see Section 3.15 on page 165) is an evolution of RC5.

3.10.1 Key Expansion

The key schedule is actually called *key expansion* for RC5. It is a complex function that essentially fills a key array S with a PRNG seeded by the secret key.

The first step is initialization. The array S is filled as follows: the first element $S[0]$ is the truncated binary expansion P of Napier's constant e, and then the truncated binary expansion Q of the Golden Ratio φ is successively added to obtain all other entries (continuing a trend to use constants above suspicion), i.e. $S[i] = S[i-1] + Q$. Also, the key is copied into an array L.

The second step is the *key mixing* stage. Both arrays S and L are mixed in several passes, using modular additions and both fixed and input dependent rotations. First, two constants A, B and two indexes i, j are all set to zero. Then, the following operations are repeated $3t$ times, where t, as defined above, is the size of S.

$$\begin{aligned} A, S[i] &\leftarrowtail \big(S[i] + (A + B)\big) \lll 3 \\ B, L[j] &\leftarrowtail \big(L[j] + (A + B)\big) \lll (A + B) \\ i &\leftarrowtail (i + 1) \bmod t \\ j &\leftarrowtail (j + 1) \bmod c \end{aligned}$$

(Note that these operations are not performed simultaneously, but sequentially in the displayed

order.) As a consequence, the key schedule is more expensive than encryption/decryption.

3.10.2 Cryptanalysis

12-round RC5 (with 64-bit blocks) is susceptible to a differential attack using 2^{44} chosen plaintexts [BK98b]. 18 to 20 rounds are now suggested as sufficient protection.

Distributed.net has also organized distributed brute force attacks on RC5. Distributed.net has brute-forced RC5 messages encrypted with 56-bit and 64-bit keys, and is working on cracking a 72-bit key; as of July 31, 2013, 2.909% of the keyspace has been searched. At the current rate, it will take approximately 120 years to search the whole key space.

3.10.3 Advantages

The design is extremely simple and leads to SW implementations that are very fast (provided a barrel shifter is available) and very compact.

Decryption and encryption require different SW or HW, but since most of the complexity is in the key schedule and the data obfuscation paths are very small, it it possible to support decryption beside encryption with little additional resources. Another consequence is that the cipher is very attractive if used with a mode of operation that keeps the same key.

3.10.4 Disadvantages

In HW the complexity of a large barrel shifter can make implementations larger than desirable.

Because of the relatively heavy key schedule, RC5 is less attractive than other ciphers with modes of operations that require key agility. Using 64-bit blocks, 128-bit keys and 16 rounds, key setup has a cost comparable to encrypting 25 blocks, i.e. about 200 bytes. This is roughly 20 times faster than the Blowfish key schedule.

3.10.5 Intellectual Property

RC5 is patented: U.S. Patent 5,724,428 and U.S. Patent 5,835,600 expire on 01.11.2015.

3.11 SQUARE

SQUARE is a block cipher invented by Joan Daemen and Vincent Rijmen. The design was published in 1997 [DKR97]. It was introduced together with a new form of cryptanalysis discovered by Lars Knudsen, called the *square attack*, now known as integral cryptanalysis.

SQUARE is a substitution-permutation network with eight rounds, operating on 128-bit blocks and using a 128-bit key. A 128-bit block is represented as a 4×4 square matrix A of 8-bit values, a concept that influenced the design of several subsequent ciphers, including Twofish (Section 3.13 on page 162) Rijndael, which has been adopted as the Advanced Encryption Standard (Section 3.20 on page 180), Klein (Subsection 3.37.1 on page 224), CRYPTON and mCrypton (Section 3.25 on page 196)

In the case of SQUARE, a round is composed of:

(a) A bitwise round key addition;

(b) A byte permutation corresponding to a transposition of the matrix;

(c) A nonlinear substitution layer, where an single 8-bit S-box (constructed by composing inversion in the field $\mathbb{F}_{2^8}$ with an affine operation) is applied in parallel to all 16 bytes; and

(d) A linear transformation, where the matrix A is multiplied by another 4×4 matrix M. Here all bytes are interpreted as elements of the field $\mathbb{F}_{2^8}$. The matrix M is the transformation matrix of a maximum distance separable (MDS) code. This operation is omitted from the last round (a common trait of wide trails ciphers, cf. Section 1.4 on page 33).

Similar functions are also the building blocks of AES and will be better described in that context.

The key schedule is called in SQUARE *key evolution* and it is a linear process, that combines XOR, rotations, and XORing with fixed constants.

3.11.1 Cryptanalysis

The cipher has withstood cryptanalysis well. The current best analysis is a biclique cryptanalysis by Hamid Mala [Mal11] that attacks the full cipher with a data complexity of about 2^{48} chosen plaintexts and a time complexity of about 2^{126} encryptions. A related-key boomerang attack is presented in [BKS10] that, requiring 2^{123} data, is not practical.

3.11.2 Advantages

The code can be extremely compact, and the high internal parallelism allows high throughput.

3.11.3 Disadvantages

Two different circuits must be implemented for encryption and decryption. This can be optimised but only sharing a few resources. Also, the key schedule can be performed in parallel with encryption, but must be performed in advance for decryption.

The biggest change between SQUARE and AES is, arguably, the key schedule. We noted the lack of successful cryptanalysis – but we should also remark that the cipher has not undergone the intense scrutiny of other ciphers. There seems therefore no rationale for using SQUARE in place of AES.

3.11.4 Intellectual Property

We are aware of no patents on SQUARE. We also note that any patent on SQUARE would cover aspects of Rijndael, and Rijndael, in the words of its designers, which are also the designers of SQUARE, is unencumbered by patents.

3.12 The TEA Family of Block Ciphers

3.12.1 TEA, the Tiny Encryption Algorithm

Roger Needham and David Wheeler designed the Tiny Encryption Algorithm (TEA) and published it in 1994 [WN94].

TEA has a block size of 64 bits and a key size of 128 bits. It is a 2-branch balanced Feistel network. The 32-bit branches are considered as unsigned integers. The suggested number of rounds is 64, typically implemented in pairs termed *cycles*, a cycle being depicted in Figure 3.19 on the next page. It has an extremely simple key schedule, mixing all of the key material in exactly the same way for each cycle: the key words k_i are the four 32-bit words of the secret key, used cyclically. Different magic constants are added in to prevent simple attacks based on the symmetry of the rounds: This idea is used also by other ciphers developed after TEA, such as PRINCE (Section 3.35 on page 215). In this case these magic constants are consecutive multiples of $\sigma = \texttt{9E3779B916}_x$, the integer part of $2^{32}/\varphi$, where φ is the golden ratio (a *nothing up my sleeve* number).

Noted for its simple design, the cipher was well studied and came under a number of attacks.

No cipher in the TEA family is subject to any patents.

3.12.2 Cryptanalysis of TEA

In 1996 John Kelsey, Bruce Schneier and David Wagner [KSW96] established that the effective key size of TEA was 126 bits, since keys could be grouped into sets of four equivalent ones. Whereas this fact does not affect the security of TEA if used for the purpose of encryption, it considerably weakens its use to build a hash function. As a result, in the hash function used to verify the integrity of the flash on Microsoft's original Xbox game console, it was possible to simultaneously flip two bits in a 64-bit block (for instance, the 16^{th} and the 31^{st}) and obtain the same hash value. This allowed hackers to patch a jump the flash code for the secondary bootloader of the XBOX in order to force the device to recover by executing a payload of their choice [Ste05].

In 1997 Kelsey, Schneier and Wagner showed a related-key attack on TEA which requires 2^{23} chosen plaintexts under a related-key pair, with 2^{32} time complexity [KSW97].

Further attacks have been discovered in the meantime:

- The best linear cryptanalysis so far is a zero correlation analysis [BW12] that breaks 23 rounds.
- The best differential attack breaks 17 rounds using impossible differentials [CWP12].

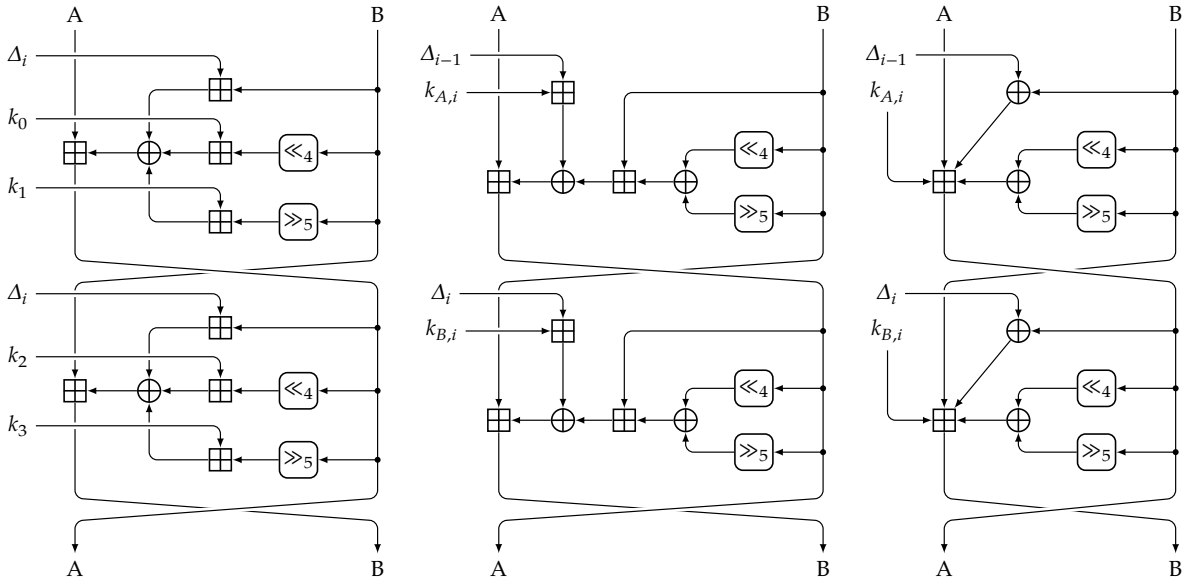
3.12.3 XTEA and Block TEA

In light of the weaknesses discovered in the original design, TEA was redesigned by Needham and Wheeler resulting in Block TEA and XTEA (eXtended TEA) [WN97].

While XTEA has the same block size, key size and number of rounds as TEA, Block TEA has variable block size. Both TEA and XTEA are implemented in the Linux kernel [Gro04].

A cycle (pair of rounds) of XTEA is depicted in Figure 3.19 on the next page. The value Δ_i is obtained by repeatedly adding σ to itself, i.e. $\Delta_i = i \cdot \sigma$. The 128-bit key K is divided into four 32-bit blocks $K[0], \ldots, K[3]$ and $k_{A,i} = K[\Delta_{i-1} \bmod 4]$, $k_{B,i} = K[(\Delta_i \gg 11) \bmod 4]$. The key derivation of the cipher is completed by the addition (modulo 2^{32}) of Δ_i to the selected key segment. This avoids the regularity in the key schedule that was typical of TEA.

Figure 3.19: A Cycle – Two Rounds – of TEA (left), XTEA (center) and XETA (right)



Jens-Peter Kaps [Kap08] observes that an ultra-low power implementation of XTEA might be better suited for low resource environments than AES. XTEA's smaller block size also makes it advantageous if an application requires fewer than 128 bits of data to be encrypted at a time.

Block TEA works on a block of size $N \cdot 32$ bits for $N \geqslant 2$. It is a Feistel network (unbalanced when $N > 2$) with N branches $v[0], \ldots, v[N-1]$. A mixing function $f(z)$ works on two adjacent branches (viewed cyclically) repeatedly: In fact $f(z) = ((z \ll 4) \oplus (z \gg 5)) + z$ with $z = v[p-1]$ is used to modify $v[p]$ for $0 \leqslant p < N$ where p is taken modulo N (the arithmetic sum of a round key and a magic constant are also XORed in as in XTEA, and the mixing is in fact the same as in the first half of a XTEA cycle). Such a cycle is applied $6 + \lfloor 52/N \rfloor$ times.

As observed by the designers, for $N = 2$ block TEA is twice as slow as XTEA, but for $2 \leqslant N \leqslant 10$ the encryption time for a single block is essentially the same, hence already for $N > 4$ Block TEA catches up and is faster than XTEA.

3.12.4 Cryptanalysis of XTEA and Block TEA

XTEA has not been broken so far – the only attacks are on reduced round versions of the cipher. The best cryptanalytic results for XTEA to date are the following:

- In 2004, Youngdai Ko et al. presented [KHL+04] the first related-key differential attack on XTEA, breaking 27 out of 64 rounds with running time $2^{115.15}$ using $2^{20.5}$ chosen plaintexts.
- In 2010 Charles Bouillaguet et al. [BDLF10] show a related-key attack on 37 rounds of XTEA with complexity 2^{125}, which can be extended to 51 rounds and complexity 2^{123} for a set of $2^{107.5}$ weak keys. These are the best attacks so far in the related-key setting.
- In 2011 Gautham Sekar et al. [SMVP11] present several MITM attacks on reduced round XTEA. These are the best attacks so far in the single key setting, breaking 15 rounds in time 2^{95} and 23 rounds in time 2^{117}. These attacks work also on the cipher XETA (see below)

and are the first cryptanalytic results for that cipher. The paper gives also an overview of previous attacks on XTEA and their complexities.

An attack on Block TEA, however, was disclosed by Markku-Juhani Saarinen in a post to the Usenet newsgroup `sci.crypt.research` in 1998 [Saa98a]. Independently of N, it is broken using 2^{34} chosen ciphertexts. The attack exploits the fact that the function f (as defined in the previous subsection) used for the mixing is not injective. If two ciphertexts are equal with exception of the last words, but that have the same image under f, then their decryption will be equal for the correct key guess (and the second last words will have different decryption). This can be used to perform a brute force search on the key word, which will be guesses in time 2^{32}. This has to be done for the four 32 bit words of the 128-bit key, hence the complexity 2^{34}.

3.12.5 XXTEA

To correct the weakness found by Saarinen in Block TEA, Needham and Wheeler designed Corrected Block TEA or XXTEA, and published it in a technical report [WN98].

XXTEA is similar to Block TEA in the sense that it applies a mixing function cyclically to adjacent branches. However, the mixing function is more complex and uses both the previous and the following branch in the mixing.

An attack published in 2010 by Elias Yarrkov [Yar10] presents a chosen-plaintext differential cryptanalysis attack against full round XXTEA, requiring just 2^{59} queries.

3.12.6 XETA

XETA was created by accident when in C implementations of XTEA higher precedence was incorrectly given to exclusive-OR over addition in the round function. The code

```
y += ((z << 4 ^ z >> 5) + z) ^ (sum + ctx->KEY[sum&3]);
```

became

```
y += (z << 4 ^ z >> 5) + (z ^ sum) + ctx->KEY[sum&3];
```

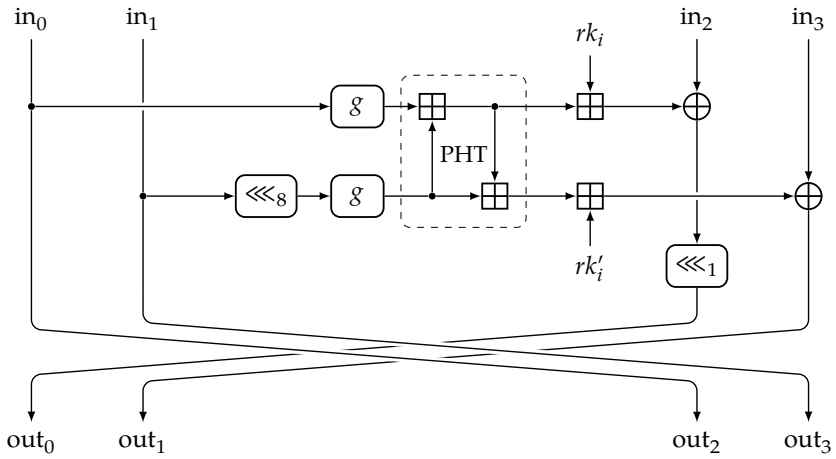
and this is reflected in Figure 3.19 on the preceding page, where a cycle of both XTEA and XETA are depicted side by side (together with TEA) in a way to underline how the mistake transformed XTEA into XETA. For the purpose of compatibility with these implementations, the Linux kernel source code includes this cipher [Gro04].

The attacks on XTEA described in [SMVP11] apply to XETA as well. That is the only cryptanalysis of XETA disclosed so far.

3.13 Twofish

Twofish was designed by Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson and submitted in 1998 to the AES Contest [SKW+98b]. It became one of the five finalists. It uses the same Feistel structure as DES (Section 3.2 on page 127). The block size is 128 bits and key sizes of 128, 192 or 256 bits. Twofish has a web page, with full specifications, free source code, and other resources.

Figure 3.20: A Round of Twofish



Twofish is related to the earlier block cipher Blowfish (Section 3.9 on page 155). Twofish's distinctive features are the use of pre-computed key-dependent S-boxes, and a relatively complex key schedule. One half of an n-bit key is used as the actual encryption key and the other half of the n-bit key is used to generate the key-dependent S-boxes. However, a big difference with respect to Blowfish is that the key schedule is lighter and can be computed for the most part in parallel with encryption and decryption.

One design criterion was to achieve large parallelism in the structure, and this is reflected in the design of the round function. A round of Twofish is depicted in Figure 3.20. The F-function is essentially composed of two equal instances of a smaller function (called "g") operating on half of the input each, whose outputs are then mixed by a PHT, a design element borrowed from the SAFER (cf. Section 3.8 on page 148) family of ciphers. The function g is itself a simple one round SPN: the substitution layer uses 4 different 8-bit S-boxes and it is followed by a linear diffusion layer based on a 4×4 MDS matrix over the field $\mathbb{F}_{2^8}$. Round key mixing occurs *after* the PHT. There are 16 such rounds, whereby the branch swap at the end of the last round is omitted. The whole cipher is bracketed by pre- and post-whitening steps.

3.13.1 Cryptanalysis

The key dependent S-boxes of Twofish are claimed to improve the security of the cipher. However, the literature contains several examples where the replacement of fixed S-boxes with key dependent S-boxes reduce the strength of block ciphers (see for example [BB94]).

The best published cryptanalysis on the Twofish block cipher [MY00] is a truncated differential cryptanalysis of the full 16-round version. The paper claims that the probability of truncated differentials is $2^{-57.3}$ per block and that it will take roughly 2^{51} chosen plaintexts (32 petabytes worth of data) to find a good pair of truncated differentials. This paper was published in the year 2000 and we are not aware of any further advancements since then.

3.13.2 Remarks (Including Advantages and Disadvantages)

There is nothing obviously wrong with Twofish: it offers good performance and has withstood cryptanalysis well. It probably offers a security level similar to Rijndael. On most software plat-

forms where the AES finalists were tested, Twofish was slightly slower than Rijndael for 128-bit keys, but was somewhat faster for 256-bit keys. Brian Gladman's analysis [Gla00] put it among the slowest AES finalists, with key schedule being particularly heavy. Other studies [GTR+07] show it can be much slower than RC6, Rijndael, XTEA and even Serpent.

The design seems to be more complex than needed, and simpler designs (such as wide trail designs) allow for more effective analysis. David Wagner, one of the designers, in a discussion on `sci.crypt` expressed himself the opinion that Rijndael was a better cipher.

Twofish serves also as proof that some design elements, such as the PHT and MDS matrices are solid foundations for the construction of secure block ciphers.

3.13.3 Intellectual Property

The cipher has not been patented and the reference implementation has been placed in the public domain, cf. http://www.counterpane.com/twofish.html. It is one of a few ciphers included in the OpenPGP standard RFC 4880.

3.14 Skipjack

Skipjack is a block cipher designed by the U.S. National Security Agency to be used in the Clipper chip for secure telecommunication. The design process started in 1987 and was finalized in 1993, when the Clipper chip was introduced. The Clipper chip was not embraced by consumers or manufacturers and this, together with the discovery of protocol vulnerabilities, de facto killed the initiative by 1996. Two years later, the design of Skipjack was declassified [NIS98].

Skipjack uses an 80-bit key to operate on 64-bit data blocks. It is an unbalanced Feistel network with 32 rounds. It has an elegant and efficient design, with a regular key schedule.

One interesting feature of Skipjack is the use of two different types of rounds. These are referred to as "rule A" and "rule B" rounds and encryption with Skipjack consists of first applying 8 rule A rounds, then 8 rule B rounds, once again 8 rule A rounds and finally 8 rule B rounds. The two rounds share some similarities, in that the 64-bit input is split into four 16 bit words, a nonlinear keyed function G is applied to one of the four words, one of the words and a counter are XORed to a second word, and the four words are permuted.

3.14.1 Remarks

The idea of mixing heterogeneous types of rounds has proven fruitful in more recent developments. Not only the cipher MARS used this idea (Section 3.16 on page 167) but also the lightweight cipher PRINCE (Section 3.35 on page 215) uses two sequences of two types or rounds that are functionally the inverse of each other, and sandwiching a *different* operation in the middle. This idea is also fundamental in the design of E2 and Camellia (Section 3.18).

The function G itself is a little 4 round Feistel network based block cipher which uses a simple 8-bit S-box. The idea of using nested Feistel networks is seen again, for instance, in KASUMI (described in Subsection 3.18.8).

3.14.2 Cryptanalysis

A reduced version of Skipjack was cryptanalysed and broken within one day of declassification [BBD+98].

Shortly after that, it was found that 31 rounds are susceptible to impossible differential cryptanalysis [BBS99a]. The attack is however only slightly faster than exhaustive search and it is the best cryptanalytic result so far.

3.14.3 Advantages

No particular advantages.

3.14.4 Disadvantages

Despite the lack of successful cryptanalysis, some aspects raise doubts about the strength of the cipher. The rounds have poor diffusion, and there is some poor interaction between round types. Also, in some cases a one bit difference in input to the S-box may cause a difference of only one bit in its output.

Skipjack also has a much lower throughput than AES-128, while using a shorter key.

3.14.5 Intellectual Property

In [DL96] we read *"A provision of the U.S. Code (Title 35, US Code 181) allows the Patent and Trademark Office (PTO) to withhold a patent and order that the invention be kept secret if publication of the patent is detrimental to national security. Relevant to cryptography is the fact that a patent application for the Skipjack encryption algorithm was filed on February 7, 1994. This application was examined and all of the claims allowed, and notification of the algorithm's patentability was issued on March 28, 1995. Based on a determination by NSA, the Armed Services Patent Advisory Board issued a secrecy order for the Skipjack patent application; the effect of the secrecy order is that even though Skipjack can be patented, a patent will not be issued until the secrecy order is rescinded. Since applications are kept in confidence until a patent is issued, no uninvolved party can find out any information concerning the application. In this way, the patentability of the algorithm has been established without having to disclose the detailed information publicly. Since Title 35 USC 181 also provides that the PTO can rescind the secrecy order upon notification that publication is no longer detrimental to national security, compromise and subsequent public revelation of the Skipjack algorithm (e.g., through reverse-engineering of a Clipper chip) might well cause a patent to be issued for Skipjack that would give the U.S. government control over its subsequent use in products."*

The existence of this patent was reported by Tom Knight on June 3rd, 1994, on the USENET group `sci.crypt`, after Brent Morris and Mark Unkenholz from NSA confirmed it during a talk at the MIT.

3.15 RC6

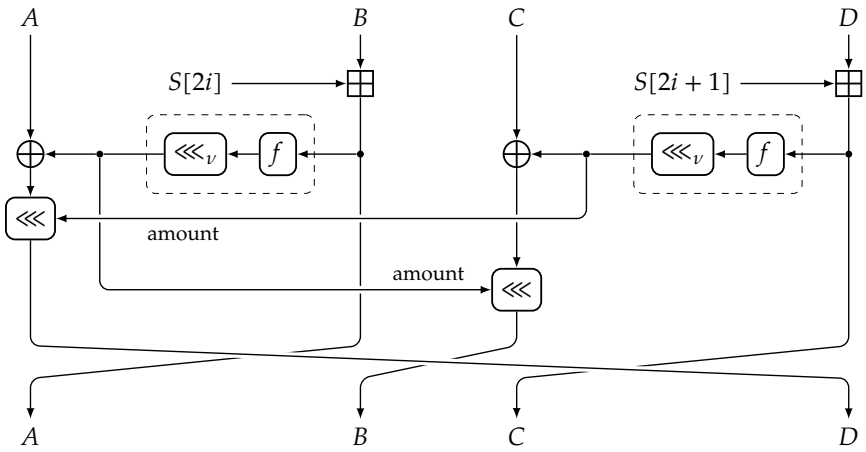
The block cipher RC6 is an evolution of RC5, which we described in Section 3.10 on page 156. It was designed by Ron Rivest with Matt Robshaw, Ray Sidney and Yiqun Lisa Yin in 1998 to be submitted to the AES competition [RRY00]. It became one of the five AES finalists. It was also submitted to the NESSIE and CRYPTREC projects.

Figure 3.21: A Round of RC6

The function f is defined as the integer multiplication $f(x) = x(2x + 1)$, truncated to w bits.

The constant ν is defined as the word size being $w = 2^\nu$ bits.

The actual F-function is the composition of f and of the fixed rotation by ν bits.



RC6 has a block size of $\ell = 4 \cdot w$ bits – for AES contest purposes $w = 32$ is fixed so we have a ℓ =128-bit block size. The value w is assumed to be a power of two and $w = 2^\nu$. Key lengths are variable. The notation RC6-$w/r/b$ denotes an instance of RC6 with w being the chosen word size, r the number of rounds and b the length of the key in bytes. The AES submission assumed $w = 32$ and $r = 20$, with b being 16, 24, or 32, corresponding to key length of 128, 192, or 256 bits.

Like RC5, it is a Feistel design, but with four branches instead of two and the inclusion of integer multiplication as an additional primitive operation, with the aim to increases the diffusion achieved per round, thus allowing for greater security with fewer rounds - resulting in increased throughput. Like RC5, RC6 makes essential use of data-dependent rotations. In fact, it is a variation on the Type 2 Feistel network, because the output of an F-function not only is composed with another input, but it also determines the amount of cyclic rotation of a second input.

In each round the state is split into *four* w-bit branches, where $w = 2^r$.

A round of RC6 is described in Figure 3.21. As already said, there are b such rounds, and at the end the outputs of all four branches are added to round keys. Therefore a total of $2b + 4$ w-bit round keys mush be generated. The key schedule algorithm is the same as in RC5, the only difference being that for RC6-$w/r/b$, a different number of words (in general higher) are derived from the user-supplied key for use during encryption and decryption.

3.15.1 Cryptanalysis

The best statistical attack on RC6 by Henri Gilbert et al. [GHJV00] breaks RC6-32/14/16 and the best correlation attack by Knudsen and Meier [KM00b] further enables a distinguisher and a key-recovery attack on RC6-32/15/32.

For a fraction of the keys called weak keys, RC6 is vulnerable to a multiple linear attack up to 18 rounds [STK02].

3.15.2 Advantages and Disadvantages

Brian Gladman's analysis [Gla00] showed that RC6 had a performance advantage over the other AES finalists in SW. However, the key schedule is slow and put it at a disadvantage with respect to Rijndael.

3.15.3 Intellectual Property

RC6 is a patented encryption algorithm covered by U.S. Patent 5,724,428 (expires on 01.11.2015), U.S. Patent 5,835,600 (expires on 01.11.2015), and U.S. Patent 6,269,163 (expires on 15.06.2018).

3.16 MARS

MARS [BCD+98] was IBM's submission to the AES Contest. It was designed by a quite numerous team that included Don Coppersmith. It has a 128-bit block size and a variable key size from 128 to 448 bits in 32-bit increments.

MARS uses a variant of the Feistel structure which its authors call a "Type 3 Feistel network:" the 128-bit block is treated as four 32-bit sub-blocks; each round uses one sub-block as input and modifies all of the other three sub-blocks – we call this type a *MARS Type-3 Feistel network* (see Figure 1.4 on page 31, Subfigure 8), to distinguish it from the different, non-equivalent, definition of Type 3 generalized Feistel network due to Zheng, Matsumoto and [ZMI89] and also studied by Hoang and Rogaway [HR10a, HR10b].

Like RC5 (and RC6), it uses data-dependent rotations. A single 9 × 32 S-box is used; for some operations it is treated as two 8× 32 S-boxes.

Unlike most block ciphers, MARS has a heterogeneous structure: several rounds of a cryptographic core are "jacketed" by unkeyed mixing rounds, together with key whitening – this approach is a design idea that it shares with Skipjack (Section 3.14 on page 164). The unkeyed external rounds of S-box based mixing are meant to provide rapid avalanche of key bits; the first set of such rounds just after key pre-whitening is called "forward mixing" and the corresponding inverse operations – called "backwards mixing" are performed at the end before the key post-whitening. This design plays the same role as bracketing a cipher with decorrelation functions (cf. Subsection 1.10.3 on page 63).

By adopting this layered approach of heterogeneous rounds, In the words of the designers of MARS, this design rationale is motivated by the fact that "*a cipher consisting of two radically different structures is more likely to be resilient to new attacks than a homogeneous cipher, since in order to take advantage of a weakness in one structure one has to propagate this weakness through the other structure. Viewed in this light, the mixed structure can be thought of as an 'insurance policy' to protect the cipher against future advances in cryptanalytical techniques.*"

A more detailed description is found at http://www.quadibloc.com/crypto/co040406.htm.

MARS ranked as the fifth and last AES contest finalist.

3.16.1 Cryptanalysis

The NESSIE project raised several doubts about the security of this cipher. The large amount of unkeyed rounds was considered a potential design vulnerability. The key schedule is a weak point of MARS, as it was overtly complex and since round keys with long runs of ones or zeroes may lead to efficient attacks on MARS, part of the design was meant to avoid that. In turn, to avoid timing attacks on the key schedule, the latter has to run very slowly, i.e. as the slowest case and in a homogeneous way, to avoid side channel attacks.

Despite the above doubts, only a few attacks on reduced-round versions of MARS have been presented to date. Due to its heterogeneous structure, MARS can be downscaled in many different ways. A first approach is to concentrate on the core rounds. In [BF00b], Biham and Furman have shown impossible differentials over 8 out of 16 core rounds. An attack breaking 11 rounds using amplified boomerang techniques is presented by Kelsey, Kohno, and Schneier [KKS00, KS00]. The same authors also proposed a straight forward meet-in-the-middle attack on a MARS version with only five core rounds, but with full forward and backwards mixing.

3.16.2 Intellectual Property

IBM has filed at least one patent application on MARS, which had been granted U.S. Patent 6,243,470. IBM has stated that they *"... are making MARS available on a royalty-free basis, worldwide, regardless of AES outcome."* In any case, the fee status of the patent is "lapsed."

3.17 Serpent

Serpent is a block cipher designed by Ross Anderson, Eli Biham, and Lars Knudsen published in 1998 [BAK98]. The first version is sometimes called Serpent-0, and a tweaked version, Serpent-1, became one of the five AES finalists [ABK00]. It ranked second behind Rijndael, essentially because of worse performance.

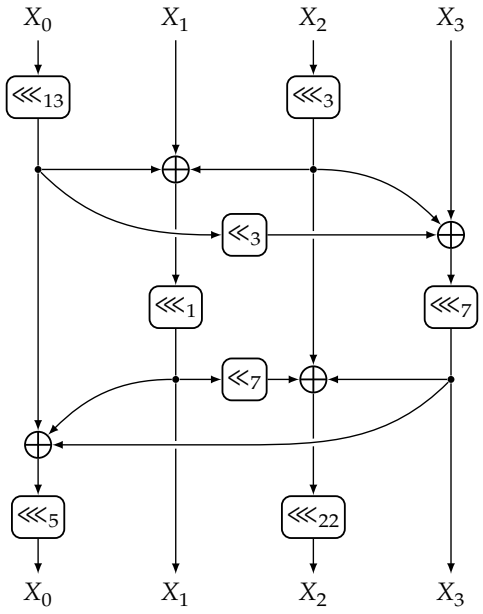
Like other AES submissions, Serpent has a block size of 128 bits and supports a key size of 128, 192 or 256 bits. It is a 32-round substitution-permutation network operating on a block of four 32-bit words. We note that the authors chose a very conservative approach, since they considered 16 rounds to be sufficient, yet they proposed 32 rounds in the AES submission. Each round applies one of eight 4-bit to 4-bit S-boxes 32 times in parallel. Serpent was designed so that all operations can be executed in parallel, using 32 1-bit-slices, greatly improving parallelism in SW implementations. Also, by suitably implementing the bit-slice technique, one can eschew table lookups, and use operations like fixed shift/rotate and logical operators instead.

The round function in Serpent consists of (i) key-mixing by XOR, (ii) thirty-two parallel applications of the same 4×4 S-box, and (iii) a linear transformation of the 128-bit output of the S-boxes, interpreted as four 32-bit words, as depicted in Figure 3.22 on the facing page. The linear transformation is omitted in the last round, where another key-mixing XOR takes its place.

3.17.1 Cryptanalysis

Serpent is considered to have a rather high security margin. The best attacks published so far break about 1/3 of the rounds.

Figure 3.22: The Serpent Linear Mixing Stage



John Kelsey, Tadayoshi Kohno and Bruce Schneier [KKS00] presented a first attack breaking 9 rounds with a time complexity slightly faster than exhaustive key search. This amplified boomerang attack was improved and extended by one round by Eli Biham, Orr Dunkelman and Nathan Keller [BDK02b]; The same authors [BDK01a] using a 9-round linear approximation for Serpent with probability of $1/2 + 2^{-52}$ attack 10-round Serpent with all key lengths with data complexity of 2^{118} and running time 2^{89}. A variant of this approximation is used in the first attack against an 11-round Serpent with 192-bit and 256-bit keys, requiring the same amount of data and 2^{187} running time. Another attack that breaks 11 rounds is a differential analysis [BDK03].

A 2011 attack by Hongjun Wu, Huaxiong Wang and Phong Ha Nguyen [NWW11] breaks 11 rounds of Serpent-128 using linear cryptanalysis with 2^{116} known plaintexts, $2^{107.5}$ time and 2^{104} memory. The same paper also describes two attacks which break 12 rounds of Serpent-256. The first requires 2^{118} known plaintexts, $2^{228.8}$ time and 2^{228} memory. The other attack requires 2^{116} known plaintexts and 2^{121} memory but also requires $2^{237.5}$ time.

3.17.2 Advantages

Since Serpent uses only one S-box, applied 32 times in parallel, the cipher can be implemented by using Eli Biham's bit-slicing technique (cf. Subsection 3.2.7 on page 131), thus avoiding table lookups.

The small S-box can help make HW implementations compact.

Reduced round versions could give good performance and acceptable security in constrained environments.

3.17.3 Disadvantages

Optimised SW implementations are larger than RC6, Rijndael, Twofish and XTEA [GTR+07], even if the S-box code is left in its own functions and not inlined. In general, SW implementations, especially in constrained environments, are slower than competing ciphers.

3.17.4 Intellectual Property

Quoting from http://www.cl.cam.ac.uk/~rja14/serpent.html: *"Serpent is now completely in the public domain, and there are no restrictions on its use. This was announced on the 21st August 2000 at the First AES Candidate Conference."*

3.18 Camellia

Camellia is a 128-bit block cipher jointly developed by Mitsubishi and NTT and first presented in 2000 [AIK+00]. It is an evolution of the previous ciphers MISTY-1 and MISTY-2 [Mat97], and E2 [KMA+98]. It also shares several design aspects with KASUMI. All these ciphers form indeed a single family, with Camellia and KASUMI being the most important members.

E2 (described below) was one of the 15 AES candidates, but not a finalist. NTT adopted many of E2's special characteristics in Camellia, which has essentially replaced E2. Therefore we focus here on the description of Camellia, instead of E2 (though we shall briefly describe the differences between the two ciphers).

The cipher has been standardised by ISO/IEC, and approved for use by the European Union's NESSIE project and the Japanese CRYPTREC project. It is widely standardised and certified for several uses, (see Subsection 3.18.6 on page 173).

It was designed to be suitable for both software and hardware implementations, from low-cost smart cards to high-speed network systems. So far, it seems that the cipher has security levels and throughput comparable to the Advanced Encryption Standard.

3.18.1 Design

Camellia can use 128-bit, 192-bit or 256-bit keys. It is a Feistel cipher with either 18 rounds (when using 128-bit keys) or 24 rounds (when using 192- or 256-bit keys). After the sixth, twelfth and, if the number of rounds is 24, eighteenth round, a logical transformation layer is applied: the so-called "FL-function" and its inverse (the FL^{-1}-Function) are applied to both source and target branches of the Feistel structure. The cipher also employs input and output key whitening, using keys derived from the secret key.

The F-function of Camellia is depicted in Figure 3.23 on the next page. The FL- and FL^{-1}-Functions of Camellia are depicted in Figure 3.24 on the facing page: they just use the XOR, AND, OR operators and single bit rotations and they were added as a simple and economical way to attempt to thwart future types of cryptanalysis. They are indeed effective: despite being, at least apparently, weaker than, say, decorrelation modules (cf. Subsection 1.10.3 on page 63), they make the cipher stronger, as it is shown by the fact that Camellia (and the MISTY ciphers) has been more successfully analysed with the FL-functions removed.

Camellia uses four different 8-bit S-boxes, which are each used twice in the S-box layer of the F-function. The structure of the F-function is superficially strikingly similar to a round of SAFER

Figure 3.23: The Camellia F-function
Round keys with indexes 0 to 7 are fed to columns 0 to 7, respectively.

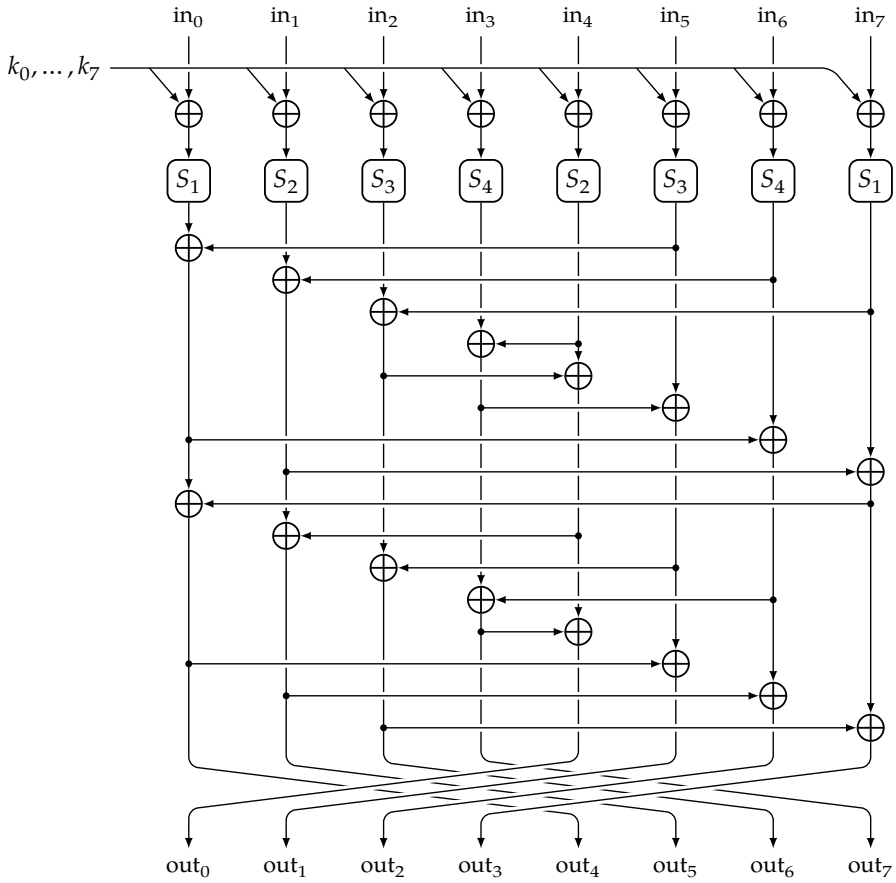
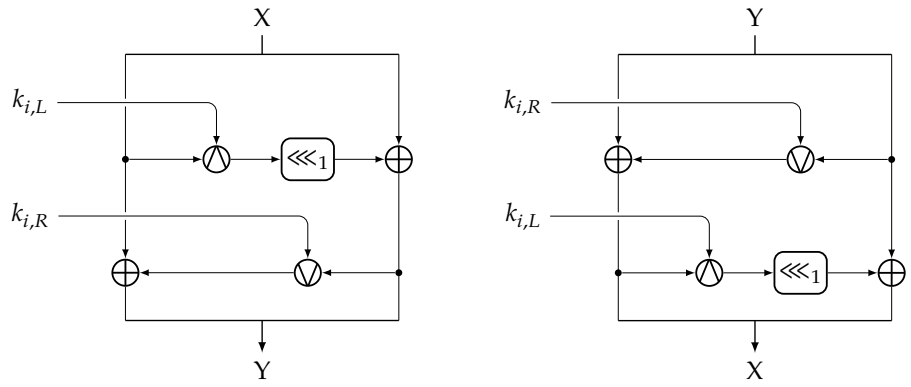


Figure 3.24: The Camellia FL and FL^{-1} Functions



(cf. Section 3.8 on page 148), but it is also simpler in that key masking is performed only once and bitwise XORs are used in place of modular additions.

The key schedule uses the encryption function itself. After combining some keys using subfields of the secret key, these are encrypted by two rounds of the cipher with some predefined

constants in place of keys: The i-th constant is just the binary expansions of the square root of the i-th prime number. From the outputs of these reduced encryptions, all sub keys are obtained using circular bit rotation and bit field extraction.

3.18.2 Cryptanalysis

The cipher has withstood intense cryptanalytic efforts.

The best attacks on reduced-round Camellia published so far are square and rectangle attacks (i.e. integral and boomerang attacks).

A nine-round square attack presented by Yongjin Yeom, Sangwoo Park, and Iljun Kim [YPK02] requires 2^{61} chosen plaintexts and an amount of work equivalent to 2^{202} encryptions. The rectangle attack proposed by Taizo Shirai [Shi02] breaks ten rounds with 2^{127} chosen plaintexts and requires 2^{241} memory accesses.

Yasuo Hatano, Hiroki Sekine, and Toshinobu Kaneko [HSK02] analyse an 11-round variant of Camellia using higher order differentials. The attack would require 2^{93} chosen ciphertexts, but is probably not significantly faster than an exhaustive search for the key, even for 256-bit keys.

Note that more rounds can be broken if the FL-layers are discarded. For Camellia-128 only 8-round reduced version without FL-layer have been broken [WZF06].

A linear attack on a 12-round variant of Camellia-256 without FL-layers is presented in [Shi02]. The attack requires 2^{119} known plaintexts and recovers the key after performing a computation equivalent to 2^{247} encryptions.

For the 192- and 256-bit variants, only 11- and 12-round reduced versions respectively have been theoretically broken, with complexity $2^{167.1}$ and $2^{220.87}$ respectively [LGLL11, LGLL12].

3.18.3 Advantages

The cipher can attain very good performance, and it seems that it has a strong advantage when fully unrolled and optimized hardware implementations are needed, as these can be implemented in a more compact way than competing ciphers (cf. the single clock HW implementations in Chapter 4 on page 233). Being a Feistel design, encryption and decryption use the same circuit and there is only a marginal overhead to support both.

3.18.4 Disadvantages

The key schedule can be partially overlapped with decryption, but it cannot run fully in parallel with it as with encryption.

3.18.5 Intellectual Property

Camellia is patented. Patents that cover aspects of the ciphers Camellia and E2 (cf. Subsection 3.18.7 on the facing page) include U.S. Patent 7,697,684, U.S. Patent 7,760,870, U.S. Patent 7,760,871, and U.S. Patent 7,822,196. Some of the protected aspects are the particular form of Feistel Network (Figure 3.23 on the previous page), the use of the FL-function and its inverse (Figure 3.24 on the preceding page), and the key schedule – but this is by no means a complete list.

However, the cipher as a whole is available under a royalty-free license [NTT01]. This has allowed the Camellia cipher to become part of the OpenSSL Project, under an open-source license, since November 2006 [NTT06], and become part of the Mozilla's NSS (Network Security Services) module [Kan07].

3.18.6 Certifications

It is widely standardised for several applications, including smart grid applications:

- RFC 3713 gives a description of the Camellia encryption algorithm, and RFC 5528 documents its use in various modes of operation;
- Camellia is widely used in Cryptographic Message Syntax (CMS): RFC 3657, RFC 5990;
- It is used in TLS RFC 4132, RFC 5932, RFC 6367 and IPSec RFC 4312, RFC 5529;
- It is certified for OpenPGP RFC 5581;
- See also RFC 4051 Additional XML Security Uniform Resource Identifiers (URIs), RFC 6030 Portable Symmetric Key Container (PSKC), and RFC 6272 Internet Protocols for the Smart Grid; and
- It is included in ISO/IEC standard 18033-3:2010.

3.18.7 E2

E2 [KMA+98] is similar to Camellia, but has a more complex F-function. E2's F-function is similar to Camellia's, but after the diffusion layer there are additional key mixing and S-box layers, and the byte permutation at the end is a rotation by one byte, instead of four bytes.

E2 does not use the FL and FL^{-1} functions, but it has more complex and more expensive keyed transformations (IT and FT, for Initial and Final Transformation) at the beginning and at the end of the cipher based on the XORing and modular multiplication modulo 2^{32} of the state with subkeys – thus requiring the use of modular inversions in order to invert the transformations. These transformations are thus related to decorrelation modules (Subsection 1.10.3 on page 63) but also to IDEA (Section 3.6 on page 142).

Key schedule is not very expensive, but the fact that it takes time between two and three encryptions reduced key agility.

As a result, Camellia is a much lighter cipher than E2 and its security is also better understood. The multiple FL/FL^{-1} layers in the cipher make also better use of the idea of mixing heterogeneous rounds first encountered in Skipjack (Section 3.14 on page 164).

3.18.7.1 Cryptanalysis

The best cryptanalysis of E2 so far is still Mitsuru Matsui and Toshio Tokita's analysis from 1999 [MT99]. They break up to 9 rounds of the cipher with IT and FT or 10 rounds without IT and FT using truncated differential cryptanalysis, requiring 2^{109} plaintext pairs.

3.18.8 MISTY-1, MISTY-2 and KASUMI

MISTY-1, MISTY-2 and KASUMI are 64-bit block ciphers with a 128-bit secret key. MISTY-1 and MISTY-2 were described first by Matsui in [Mat97] and are patented by Mitsubishi Electric Corporation, whereas KASUMI was introduced in 1999 by 3GPP to be used in the UMTS security system by the Security Algorithms Group of Experts (SAGE).

MISTY stands for *Mitsubishi Improved Security TechnologY*, and it is also the acronym formed by the initials of the family names of its designers: Mitsuru Matsui, Tetsuya Ichikawa, Toru Sorimachi, Toshio Tokita, and Atsuhiro Yamagishi. Finally, "kasumi" (霞み) is the Japanese word for "mist.".

The complete description of these three ciphers is complex, however, we want to mention a few important high-level aspects of this cipher family.

MISTY-1 and KASUMI are Type-1 Feistel networks and MISTY-2 is a "Matsui network" as represented in Figure 1.4 on page 31, Subfigure 2. Eight rounds are suggested for MISTY and are mandated for KASUMI.

The F-function of the round functions is called the "FO-function." it is a three-round, two-branch, balanced 32-bit Matsui network. The round function of the FO-function is also implemented as a three-round (in MISTY-1 and MISTY-2) or four-round (KASUMI) *unbalanced*, 9 + 7-bit Matsui network, without repartitioning of the inputs between rounds. The data from the source branch to the target branch is either truncated or zero-extended to the width of the target branch. The FO-function uses two S-boxes, one of which is a 9-bit S-box and the other one is 7-bit.

An additional function is used, the keyed "FL-function." It is similar to Camellia's FL-function, but with some differences: In MISTY-1 and MISTY-2, it omits the cyclic rotation but it is otherwise unchanged; in KASUMI the FL-function has instead *two* cyclic rotation, one between the AND and the XOR, and one between the OR and the XOR. In MISTY-1, FL-layers separate every two rounds of the cipher (as in Camellia). and the FL-functions are placed on both the source and target branches. In MISTY-2, FL-functions are placed on both the source and target branches every four rounds, and every other round on the source-to-target mixing connection. In KASUMI, they are inserted before and after the FO-functions for odd and even rounds respectively, i.e. the F-function is a functional composition of FO- and FL-functions.

The choice of the Matsui network allows several operations to be performed in parallel.

These ciphers herald a hierarchical (also recursive, or nested) design, that is later used also by other types of ciphers, for instance by the block cipher Hierocrypt [OMSK00].

There are several round keys, but the key schedule is very straightforward.

3.18.8.1 Cryptanalysis

MISTY has been widely studied since its publication, but no serious flaws have been found.

Until recently, the best attack on reduced-round MISTY was the 5-round integral attack by Knudsen and Wagner [KW02] The attack requires 2^{34} chosen plaintexts and has a time complexity of 2^{48}. In a recent paper, Yasutaka Igarashi et al. [IKE+13] break 8 Rounds of MISTY-2 without FL functions in time $2^{57.4}$ using 2^{35} blocks of chosen plaintext.

In 1999 Hidema Tanaka, Kazuyuki Hisamatsu, and Toshinobu Kaneka [THK99] break five rounds of MISTY-1 without FL functions using 11 different seventh order differentials. A first explanation of this attack was given by Steve Babbage and Laurent Frisch at ICISC 2000 [BF00a], and the fact that seventh order differential cryptanalysis works (at least partially) is related to the fact that several non-linear components have degree bounded, in this case, by seven. Anne Canteaut and Marion Videau [CV02] further prove that this is caused precisely by the use of highly non-linear almost bent functions, i.e. the specific type of SBOX that has been chosen.

The best non-related-key attack cryptanalysis of KASUMI is still Ulrich Kühn's EUROCRYPT 2001 paper [Küh01] where an impossible differential attack on a six rounds reduced version of KASUMI is described.

In 2010, Orr Dunkelman, Nathan Keller, and Adi Shamir, broke KASUMI by a related key attack with very modest computational resources [DKS10b]. This attack takes less than two hours on a single PC, but isn't applicable to 3G due to the plaintext and related key requirements. Hence, KASUMI as used in 3G is not broken. The attack is ineffective against MISTY.

A 32nd order differential attack breaks eight rounds of MISTY-2 without FL functions [IKE+13] with 2^{35} blocks of chosen plaintext and in time $2^{57.4}$.

3.18.8.2 Intellectual Property

As already mentioned, MISTY-1 is patented – see for instance U.S. Patent 6,201,869 (PCT number PCT/JP1996/002154) – but, as stated in RFC 2994, *"the algorithm is freely available for academic (non-profit) use. Additionally, the algorithm can be used for commercial use without paying the patent fee if you contract with Mitsubishi Electric Corporation."* The patent cover several ideas, such as the use of the Feistel/Matsui networks, including unbalanced 9 + 7 bits variants thereof (Embodiment 6). According to [3GP09], Mitsubishi holds essential patents on KASUMI, and a separate IPR License Agreement from Mitsubishi is necessary to use the cipher.

3.19 The AES Contest

This section serves as (i) an historical interlude, (ii) a prelude to the description of Rijndael, and (iii) a placeholder to give brief descriptions and references to other block ciphers which were submitted to the AES contest and are worth mentioning.

The US National Institute of Standards and Technology (NIST) launched in January 1997 a contest to find a block cipher to replace the aging DES. DES had become obsolete because its 56-bit key size was inadequate to resist brute force attacks, given modern technology, the block length was too short for several types of applications, for instance to construct hash functions, and was also found to be vulnerable to several types of attacks.

On October 3rd, 2000 Rijndael (described in Section 3.20) was chosen as the winning cipher. The ratification of Rijndael as the new Advanced Encryption Standard, AES, took place on November 26th, 2001.

The entire process was completely open; even the requirements document was first published as a draft and comment or criticism invited before a final version was written, and all of the later work toward the choice of a standard was public as well. Because of this, the process was widely praised for its laudable goal of taking Kerckhoffs' Principle into account – the security of

an encryption method should be based on key secrecy, not on the secrecy of (design aspects of) the cipher. The final requirements specified a block cipher with 128-bit block size and support for 128, 192 or 256-bit key sizes. Evaluation criteria included security, performance on a range of platforms from 8-bit CPUs (e.g. in smart cards) to high end CPUs, and ease of implementation in both software and hardware.

Fifteen submissions meeting basic criteria were received, from the whole world. All of the entries were iterated block ciphers. Most designs were substitution-permutation networks or Feistel structures, or variations of those. Several had proofs of resistance to various attacks.

In alphabetical order, the 15 first round candidates were: CAST-256 (Section 3.7 on page 146), CRYPTON [Lim99] (the very similar cipher mCrypton is described in Section 3.25 on page 196), DEAL (Subsection 3.19.1 on the next page), DFC (Subsection 3.19.2 on the facing page), E2 (discussed in Section 3.18 on Camellia), FROG (Subsection 3.19.3 on page 178), Hasty Pudding (Subsection 3.19.4 on page 178), LOKI97 (Subsection 3.19.5 on page 178), MAGENTA (Subsection 3.19.6 on page 179), MARS (Section 3.16 on page 167), RC6 (see Section 3.10 on page 156 on RC5), Rijndael (Section 3.20 on page 180), SAFER+ (discussed in the section about SAFER, Section 3.8 on page 148), Serpent (Section 3.17 on page 168), and Twofish (Section 3.13 on page 162).

After intense scrutiny by many of the world's best-known cryptographers, and two conferences, the field was narrowed to five finalists: Twofish, MARS, Serpent, RC6, and Rjindael. One could summarize the relative strengths and weaknesses of these five finalists as follows:

- MARS - very complex, complicated key schedule, reasonably fast but difficult to achieve good performance and secure implementation in key schedule, good security margin;
- RC6 - very simple and quite fast, but low security margin;
- Rijndael - clean and well understood design, fast, high security margin;
- Serpent - slow, especially in SW and more so in constrained environments, clean design, very high security margin;
- Twofish - complex, quite fast but not as fast as Rijndael, high security margin.

After another year of analysis and testing focused on the finalists, and another conference with all of the finalist teams giving presentations, a winner was chosen by vote. And so, on October 2, 2000, NIST announced that it had chosen Rijndael as the AES. On November 26, 2001, AES was formally approved as a US federal standard.

RC6 is the only one of the five finalists which does not have a completely open license; it is still proprietary to RSA Security. The other finalists can be used freely.

In 2003 the NSA announced that it allows the use of AES to encrypt classified documents up to the level SECRET for all key lengths, and up to the TOP SECRET level for key lengths of either 192 or 256 bits. Prior to that date, only non-public algorithms had been used for the encryption of classified documents.

We already described many of the 15 submissions or ciphers strongly related to them. We briefly comment here on the remaining algorithms, in alphabetical order, except for CRYPTON, for the reasons that it is very similar to SQUARE and Rijndael and we shall discuss the also very similar mCrypton in Section 3.25 on page 196.

3.19.1 DEAL (and Ladder-DES)

DEAL, the Data Encryption Algorithm with Larger blocks, is a Feistel network which uses DES as the F-function, similarly to its precursor Ladder-DES by Terry Ritter. The design was proposed in a report by Lars Knudsen in 1998 [Knu98], and was submitted to the AES contest by Richard Outerbridge (who notes that Knudsen had presented the design at the SAC conference in 1997).

It did not make it into the finals, and was entered mainly to provide a baseline for comparison to other ciphers. DEAL has approximately the overheads of Triple DES, making it too slow to be a competitive candidate for AES.

It has a 128-bit block size and a variable key size of either 128, 192, or 256 bits. It uses 6 rounds for key sizes of 128 and 192 bits, 8 rounds for 256-bit keys. Key schedule consists in concatenating a sufficient number differently masked copies of the given private key, and DES-CBC encrypting this buffer with fixed IV and key.

The interest of this cipher is that it is a proper exemplification of the Luby-Rackoff construction (cf. Section 1.3 on page 27). Using that construction, it is easy to double the block size of an established cipher by using it as the F-function, at the price of a considerably slower operation and potentially amplifying the effects of biases in the original cipher. Furthermore, since cryptographic ciphers used in practice are *not* perfect pseudorandom functions, three or four rounds are often not sufficient. Indeed, Ladder-DES, which is a four round Luby-Rackoff construction, was quickly broken by Eli Biham [Bih97a]. Ladder-DES uses four 56-bit keys, and therefore it is expected that a meet-in-the-middle attack will have a complexity of 2^{112}. However, Eli Biham showed hot to recover the complete key in time 2^{88}.

Stefan Lucks analyzed the security of DEAL in [Luc99] and proved that the DEAL with with 128, 192, and 256 bits keys, can be broken in time 2^{121}, 2^{121}, and 2^{224} respectively.

3.19.2 DFC

Another theoretically interesting submission is DFC, the *De-correlated Fast Cipher*, described in [GGH+98] and revised in [GNNV00].

It is an eight-round Feistel cipher.

The secret key K is first turned into a 1024-bit "Expanded Key" EK through an Expanding Function EF: these 1024 bits are then split into the eight 128-bit round keys used by the encryption. The EF function performs a four round Feistel scheme using the same round function RF as the encryption, where the inputs and the round keys are derived from K and some predefined constants.

The function RF is related to a Type IV decorrelation module (cf. Subsection 1.10.3 on page 63) that maps a 64-bit string onto a 64-bit string by using one 128-bit string parameter. It mixes modular multiplication modulo $2^{64} + 13$, reduction modulo 2^{64} and it uses a single 6×32 bits S-box to determine a 32-bit value to mix in according to six bits of the input.

All the constant used in the cipher are taken from the expansion of Napier's constant e.

This cipher was based on Serge Vaudenay's theoretical work on decorrelation theory. That theory gives methods of constructing ciphers which are provably immune to differential crypt-

analysis, linear cryptanalysis, and any other attacks that meet some fairly broad assumptions. However, some attacks on DFC were found by going outside those assumptions, such as a variant of differential analysis [KR99a].

3.19.3 FROG

FROG is a variable size block cipher. It supports block sizes from 8 to 128 bytes and key sizes from 5 to 125 bytes.

Its design is very unorthodox: it uses data derived from the primary key as a program for an interpreter, so that each round can use a different sequence of operations. Eight rounds are used. Encryption and decryption are fast, but the key schedule is rather slow because it has to build a program for the interpreter.

Despite the fact that the sequence of computations is variable, David Wagner, Niels Ferguson, and Bruce Schneier in [WFS99] broke it using differential cryptanalysis - and also found that it is faster to attack decryption than encryption. Also, the design makes it very difficult to implement both the key schedule and the ciphering operations resistant against power analysis.

3.19.4 Hasty Pudding

Rich Schroeppel's Hasty Pudding Cipher or HPC is a variable size block cipher; blocks can be any size the application requires. It therefore might be ideal for things like encrypting disk blocks. Key size is also variable; any integer number of bits. It is designed for architectures with 64-bit operations.

The design of the cipher is obscure and it is not well understood. It has a very expensive key setup. These two factors led to the cipher not being admitted to the final round.

Hasty Pudding's interest lies in the fact that it is the first tweakable block cipher, and it was designed before that term was actually introduced.

A **tweakable** block cipher [LRW02, LRW11] is a block cipher that accepts a *third* input called the **tweak**. The tweak, along with the key, selects the permutation computed by the cipher. Changing the key can be expensive, but changing the tweak should remain a lightweight operation. A tweak makes creating modes of operation for block ciphers easier to construct and to analyse. In the case of a tweakable block cipher, to be secure means that an adversary should not be able to break the cipher even with control of the tweak input.

The tweak in Hasty Pudding is called "spice."

3.19.5 LOKI97

LOKI97 was the first published candidate in the AES contest [BP98]. The design and analysis of LOKI97 was performed by Lawrie Brown with assistance and critique from Josef Pieprzyk – that also designed the S-box functions – and Jennifer Seberry.

It is a 16-round Feistel cipher with an F-function that is basically two rounds of an SPN – whereas in DES the F-function is a single SP round. We recall that the F-function has two inputs, a half of the cipher state (L) and a round key k. So, we use the notation $F = F(L, k)$. The F-function uses two large 11- and 13- bit S-boxes.

The key schedule is a source heavy 4-branch Feistel network. The input is quartet of words $[k_4, k_3, k_2, k_0]$, where each k_i is a 64-bit value. Initially this is the encryption key, padded to 256 bits for shorter keys. The quartet is then updated using same F-function as the encryption, as follows: first the next round key $\sigma \leftarrow\!\shortmid F\left(k_1 \oplus k_3 \oplus (i \cdot \Delta), k_2\right) \oplus k_4$ is computed, then the substitution $[k_4, k_3, k_2, k_1] \leftarrow\!\shortmid [k_3, k_2, k_1, \sigma]$ is performed. The constant Δ is $\left\lfloor \left(\sqrt{5} - 1\right) \cdot 2^{63} \right\rfloor$.

For each round of the cipher, a total *three* round keys have to be generated: one as the key to the keyed F-function and two to be directly mixed to the branches via modular additions. The Feistel round itself uses a XOR to mix the output of the F-function.

This description makes it clear that the cipher is computationally heavy and in fact, according to Schneier, it ranked 12^{th} in performance among the 15 AES candidates. Also, the key schedule can run in parallel with encryption, but not with decryption.

The cipher was broken by Lars Knudsen and Vincent Rijmen [KR99b] using both linear cryptanalysis and differential cryptanalysis. Both attack require only about 2^{56} known plaintexts.

Further cryptanalysis includes [WLFQ99] (where also AES candidates DFC and MAGENTA are analysed) and [WLFQ00].

3.19.6 MAGENTA

The *Multifunctional Algorithm for General-purpose Encryption and Network Telecommunication Applications* was Deutsche Telekom's entry in the AES competition [JH98].

MAGENTA has a block size of 128 bits.It is a Feistel cipher with six rounds for the key size of 128 and 192 bits, and eight rounds for the key size of 256 bits.

It is often cited as an example of Kerckhoffs' Principle, a demonstration of why unpublished and therefore unanalysed ciphers cannot be trusted. Unlike all other candidates, this cipher was made available to the conference attendees only on the day of presentation. Its presentation was given on the morning of August 20^{th}, 1998, to an audience that included many of the world's top cryptographers. Some saw flaws, and there was intense discussion over lunch. By that evening, a draft paper on breaking the cipher was circulating and the final version [BBF+99] was presented at the second AES conference.

The S-box is constructed in an interesting way, inspired by the exponential S-boxes found in ciphers like SAFER (Section 3.8 on page 148): The S-box is essentially the map $\mathbb{N} \to \mathbb{F}_{2^8} = \mathbb{F}_2[x]/(\phi(x))$ where $n \mapsto x^n \mod \phi(x)$ and $\phi(x) = x^8 + x^6 + x^5 + x^2 + 1$, except for the last entry $\phi(255) = 0$. This S-box present decent differential properties (but far from those of AES), but bad linear properties.

Another design principle of MAGENTA worth mentioning is the construction of the F-function. The F-function first concatenates a 8-byte branch of the Feistel network with an 8-byte round key - resulting in a 16 byte vector w_0. The resulting 16-byte vector is then shuffled (the first eight bytes are interleaved with the last eight), and the 8 pairs of adjacent bytes are transformed as $(x, y) \mapsto \left(\phi(x \oplus \phi(y)), \phi(y \oplus \phi(x))\right)$. The combination of shuffling and substitution is called the Π operation in MAGENTA. In order to guarantee diffusion, Π is performed four times in a row, this giving rise to a modified FHT (fast Hadamard transform).

Then w_0 is XORed to the state, Π is applies again four times, w_0 XORed to the state a second time, Π is applies another four times. The value of the F-function is formed by concatenating

the bytes in the even numbered positions of the final state.

The key schedule is perhaps the biggest weakness of the design. If a $64\,t$-bit key K is used, it is first split into 64-bit chunks k_i for $1 \leqslant i \leqslant t$, which are then used as the round keys. For 128, resp. 192 and 256-bit key the sequences are $[k_1, k_1, k_2, k_2, k_1, k_1]$; $[k_1, k_2, k_3, k_3, k_2, k_1]$; and $[k_1, k_2, k_3, k_4, k_4, k_3, k_2, k_1]$. This immediately leads to splice-and-cut attacks (Subsection 2.4.4 on page 100): MAGENTA-128 can be broken using 2^{64} chosen plaintexts in time 2^{64}, MAGENTA-192 can be broken using 2^{64} chosen plaintexts in time 2^{128}, and MAGENTA-256 can be broken using 2^{128} chosen plaintexts in time 2^{128}. All these attacks can be turned into known plaintext attacks with different tradeoffs, for instance reducing the memory usage by n bits while increasing the attack complexity by about n bits. These attack were presented (without recognising them as Merkle-Hellman attacks and the corresponding van Oorschot-Wiener known plaintext variants) already in [BBF+99].

Another remark, made also by the designer of the cipher themselves, is that, due to the symmetry of the key scheduling, encryption and decryption are identical except for the order of the two halves of the plaintexts and ciphertexts. Given a ciphertext, it be can decrypted just by swapping its two halves, reencrypting the result, and swapping again. The main consequence is that MAGENTA cannot be used in scenarios where an attacker has access to an encryption oracle, considerably reducing its fields of application. Also, for any fixed key, the cipher (without final swap) will have an expected number of 2^{64} fixed points which means that blocks of plaintexts may be revealed with a significantly higher likelihood than the claimed security level.

3.20 AES (Rijndael)

Rijndael [DR02a, DR02b], the winner of the AES contest, was designed by Joan Daemen and Vincent Rijmen. As AES, Rijndael's block size is 128 bits (additional block sizes are supported by the original submission). It is a SP network with 10, 12, or 14 rounds for key sizes of 128, 192, and 256 bits respectively. The cipher is designed according to the wide trails strategy (Section 1.4 on page 33), and in particular it is directly derived from SQUARE (Section 3.11 on page 158).

The 128-bit state is represented as a 4×4 matrix of bytes

$$\begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}.$$

The entries can be interpreted as machine bytes as well as elements of the Galois field $\mathbb{F}_{2^8}$. The latter is represented using a polynomial basis defined by the polynomial $x^8 + x^4 + x^3 + x + 1$. A byte is interpreted as an element of $\mathbb{F}_{2^8}$ in the following way: the i-th bit of the byte is the coefficient of x^i.

The rounds are constructed from following operations:

(a) `AddRoundKey`: Mixing (XOR) of key derived material. The state is considered as a 128-bit value which is then bitwise XORed to a round key.

(b) `SubBytes`: Run all entries through an 8-bit S-box. The Rijndael S-box is derived from the inversion operation, i.e. the input is considered as an element of $\mathbb{F}_{2^8}$ and any non-zero ele-

ment is mapped to its inverse; the zero is mapped to zero. (This choice has been influenced by Kaysa Nyberg's theoretical groundwork [Nyb93].) In order to avoid fix points, the inversion is composed with an affine operation.

(c) `ShiftRows`: Cyclicly shift each row by 0, 1, 2 and 3 positions respectively. In mathematical notation

$$\begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \mapsto \begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,1} & a_{1,2} & a_{1,3} & a_{1,0} \\ a_{2,2} & a_{2,3} & a_{2,0} & a_{2,1} \\ a_{3,3} & a_{3,0} & a_{3,1} & a_{3,2} \end{pmatrix} .$$

(d) `MixColumns`: Multiply the state matrix by a fixed MDS matrix, as follows

$$\begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \mapsto \begin{pmatrix} x & x+1 & 1 & 1 \\ 1 & x & x+1 & 1 \\ 1 & 1 & x & x+1 \\ x+1 & 1 & 1 & x \end{pmatrix} \cdot \begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} .$$

Multiplication by x corresponds to a left shift, and multiplication by $x+1$ means XOR of the input with its left shift. After shifting, a conditional XOR with $\texttt{1B}_x$ should be performed if the shifted value is larger than $\texttt{FF}_x$.

This is equivalent to considering each column as a polynomial over the Galois field $\mathbb{F}_{2^8}$, and then multiply it modulo $X^4 + 1$ with fixed polynomial $C(X) = (x+1) \cdot X^3 + X^2 + X + x$.

Without `MixColumns` the cipher would be equivalent to the interleaving of three strongly related 32-bit block ciphers and four 8-bit ones – as the `ShiftRows` operation only permutes entries within three of the four rows.

Let r be the number of rounds (10,12 or 14). The four fundamental operation are used in the following way to construct the whole cipher.

(a) Key Expansion (or Key Schedule) – round keys are derived from the cipher key. During encryption this can run in parallel with the data encryption path.

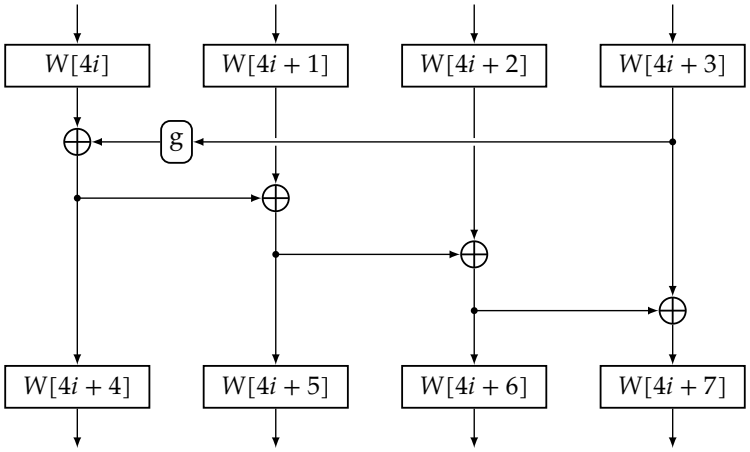
(b) The Initial Round consists of just `AddRoundKey`, that is it corresponds to pre-whitening.

(c) The following $t-1$ Rounds consist each of `SubBytes`, `ShiftRows`, `MixColumns`, and `AddRoundKey`, in this order.

(d) The Final Round consists only of the `SubBytes`, `ShiftRows` and `AddRoundKey` operations. Note that `MixColumns` is omitted: This does not affect security, since it is a linear operation and therefore having it or not just corresponds to a linear change of the last round key - that has a post-whitening effect.

The key schedule generates 11, 13, or 15 128-bit round keys for the 10, 12, and 14 round versions of the cipher. The first sub key is equal to the private key. To generate all subsequent keys, a structure is used that is loosely related to 4-branch Feistel network using a round function g. The key schedule is represented in Figure 3.25 on the following page in the case of 128-bit keys. It can be seen as a kind of Feistel network: The initial words $W[0]$ to $W[3]$ are the unmodified

Figure 3.25: The AES Key Schedule for 128-bit Keys



encryption key – which is also the first round key. Each iteration of the key schedule generates a further 128-bit round key. The versions for 192- and 256-bit keys present minor differences.

The function g is a 32-bit to 32-bit function consisting of: (i) a right rotation of the input by 8 bits, (i) running all four bytes of this rotated input through the AES S-box, and (iii) the addition of a fixed round coefficient to the output of the first S-box.

3.20.1 Remarks

The lineage of the cipher (and that of SQUARE before it) can clearly be traced to precursors such as SAFER (Section 3.8), where the round function consists of key mixing, an array of S-boxes, and a layer of regular, homogeneous arithmetic operations in an algebraically incompatible arithmetic structure – in SAFER's case the integers modulo 256 and in Rijndael's case the Galois field $\mathbb{F}_{2^8}$.

3.20.2 Cryptanalysis

Despite the best efforts of the whole cryptographic community, the AES has withstood cryptanalysis well. In fact, there is no attack on the full round versions of the cipher that is more than 4 times faster than brute force in the single key setting.

Tables 3.2, 3.3 and 3.4 report the best attacks on AES-128, AES-192 and AES-256 in the single key model that we were able to find in the published literature. In these tables as well as in all the other attack complexity tables in this chapter, the "data" column usually contains the amount of CP data, unless otherwise noted – the first letter being "K" meaning "known," and the second letter being "C" meaning "Ciphertext."

So-called *Biclique* analysis [BKR11a, BKR11b] brought the first key recovery attacks on the full AES-128, AES-192, and AES-256 with computational complexities better than brute force, namely $2^{126.2}$, $2^{189.4}$ and $2^{254.4}$, respectively. We do not consider these attacks to actually affect the security of the AES.

Until a few years ago, no attack could break more than 7 rounds of any variant of the AES, and it was first in [LDKK08], resp. [DS08a], that 7 rounds of AES-128 and 8 rounds of AES-192,

resp. 8 rounds AES-256 have been attacked with time complexity significantly lower than brute force.

The attacks with the lowest complexities [BKN09, BK09] are all in the related-key model, that could be relevant, for instance, if an attacker can flip some bits in the keys by fault injection. However note that the complexity of such attacks greatly increases with the number of bits involved and all the related key relations are not trivial. These attacks are reported in Table 3.5. So far they only apply to the 192- and 256-bit key variants of AES.

Low-data attacks are a potentially more serious type of attack, because in many adversarial scenarios an attacker cannot collect many chosen text pairs. Table 3.6 reports these attacks, that so far do not seem to be able to break more than four rounds.

The algebraic structure of the AES is fairly simple, making it possible to describe the whole cipher by a relatively structured system of equations, which in particular is quite sparse over $\mathbb{F}_{2^8}$. In 2002 Nicolas Courtois and Josef Pieprzyk in [CP02a] presented an algorithm to solve such systems of equations, named the "XSL attack," and conjectured that it should be able to break the AES, independently from the key size, in time 2^{203}. Murphy and Robshaw argue that this complexity may be as low as 2^{100}. Since then, Carlos Cid and Gaëtan Leurent [CL05], and Chu-Wee Lim and Khoongming Khoo [LK07] have shown that the attack cannot possibly work as originally presented.

3.20.3 Advantages

The AES's performance performance is in general pretty good. The cipher was designed with both SW and HW performance optimisations in mind. It allows for considerable parallelism in implementation. There are several ways to merge different steps into simple table lookups – on the other hand care must be taken to avoid cache attacks.

As shown in [RDJ+01], the field $\mathbb{F}_{2^8}$ can be implemented using subfields, and this leads to compact representation of the arithmetic and of the S-box in HW. bit-slicing can also be used, since only one S-box is used, allowing fast SW implementations.

Intel introduced special ISA extensions to speed up AES computation (ARM has defined similar extensions as well).

3.20.4 Disadvantages

Encryption and decryption are different enough that the overhead of an implementation of both over an implementation of encryption only is considerable. Despite this, encryption and decryption can be implemented by a single circuit (see [RSQL04]).

The key schedule can be computed in parallel with encryption, but not with decryption, making slower decryption an issue in modes of operation that change key often (unless the key schedule for the new key can be computed in parallel with the previous operations).

The cipher is still too slow for very high-performance applications where power and area are a concern - in other words it is not a lightweight or ultra-lightweight cipher.

Table 3.2: Single Key Cryptanalysis of AES-128 – Selected Published Results

Rounds Attacked	Attack Complexity Time	Data	Memory	Technique	Reference
6/10	2^{72}	$6 \cdot 2^{32}$	2^{8}	Square	[FKL+00]
6/10	$2^{106.17}$	2^{8}	$2^{106.17}$	MitM	[DF13, DF15]
7/10	2^{120}	$2^{127.997}$	2^{64}	Square	[FKL+00]
7/10	$2^{117.2}$	$2^{112.2}$	$2^{112.2}$	Imp. diff.	[LDKK08]
7/10	$2^{117.2}$	$2^{90.4}$	2^{106}	Imp. diff.	[MDRM10]
7/10	$2^{110.2}$	$2^{106.2}$	$2^{90.2}$	Imp. diff.	[MDRM10]
7/10	$2^{119.32}$	$2^{115.32}$	2^{45}	Imp. diff.	[Yua10]
7/10	$2^{126.47}$	2^{32}	$2^{126.47}$	MitM	[DF13, DF15]
7/10	2^{99}	2^{105}	2^{90}	MitM	[DFJ13]
7/10	2^{99}	2^{99}	2^{96}	MitM	[DFJ13]
8/10	$2^{125.3}$	2^{88}	2^{8}	Bicliques	[BKR11a]
10 (full)	$2^{126.2}$	2^{88}	2^{8}	Bicliques	[BKR11a]

Imp. diff: Impossible differentials

Table 3.3: Single Key Cryptanalysis of AES-192 – Selected Published Results

Rounds Attacked	Attack Complexity Time	Data	Memory	Technique	Reference
6/12	$2^{109.67}$	2^{8}	$2^{109.67}$	MitM	[DF13, DF15]
7/12	2^{155}	$19 \cdot 2^{32}$	$19 \cdot 2^{32}$	Square	[FKL+00]
7/12	$2^{117.2}$ MA	$2^{112.2}$	$2^{93.2}$	Imp. diff.	[LDKK08]
7/12	2^{143}	2^{95}	2^{143}	MitM	[DS08a]
7/12	2^{116}	2^{116}	2^{116}	MitM	[DKS10a, DKS15]
7/12	2^{172}	2^{113}	2^{129}	MitM	[DKS10a, DKS15]
7/12	$2^{154.67}$	$2^{109.67}$	2^{45}	Imp. diff.	[Yua10]
7/12	2^{163}	2^{8}	$2^{153.34}$	MitM	[DF13, DF15]
7/12	$2^{129.67}$	2^{32}	$2^{129.67}$	MitM	[DF13, DF15]
7/12	2^{99}	2^{97}	2^{98}	MitM	[DFJ13]
8/12	2^{188}	$2^{128} - 2^{119}$	2^{64}	Square	[FKL+00]
8/12	$2^{118.8}$ MA	$2^{113.8}$	$2^{113.8}$	Imp. diff.	[LDKK08]
8/12	$2^{139.2}$	$2^{91.2}$	2^{101}	Imp. diff.	[LDKK08]
8/12	$2^{166.3}$	$2^{102.3}$	2^{45}	Imp. diff.	[Yua10]
8/12	$2^{187.63}$	2^{41}	2^{186}	MitM	[WLH11]
8/12	2^{172}	2^{107}	2^{96}	MitM	[DFJ13]
8/12	2^{172}	2^{113}	2^{82}	MitM	[DFJ13]
8/12	$2^{182.17}$	2^{32}	$2^{182.17}$	MitM	[DF13, DF15]
8/12	2^{140}	$2^{104.83}$	$2^{138.17}$	MitM	[DF13, DF15]
8/12	2^{140}	2^{113}	2^{130}	MitM	[DF13, DF15]
9/12	$2^{180.89}$	$2^{115.89}$	2^{45}	Imp. diff.	[Yua10]
9/12	$2^{150.89}$	$2^{125.89}$	2^{45}	Imp. diff.	[Yua10]
9/12	$2^{188.8}$	2^{80}	2^{8}	Bicliques	[BKR11a]
12 (full)	$2^{189.4}$	2^{80}	2^{8}	Bicliques	[BKR11a]

MA: Memory Accesses (as opposed to Encryptions)

Table 3.4: Single Key Cryptanalysis of AES-256 – Selected Published Results

Rounds Attacked	Attack Complexity Time	Data	Memory	Technique	Reference
6/14	2^{122}	2^{8}	$2^{114.34}$	MitM	[DF13, DF15]
7/14	2^{172}	$21 \cdot 2^{32}$	$21 \cdot 2^{32}$	Square	[FKL+00]
7/14	2^{143}	2^{95}	2^{143}	MitM	[DS08a]
7/14	2^{116}	2^{116}	2^{116}	MitM	[DKS10a, DKS15]
7/14	$2^{170.34}$	2^{8}	2^{186}	MitM	[DF13, DF15]
7/14	2^{178}	2^{16}	$2^{153.34}$	MitM	[DF13, DF15]
7/14	$2^{133.67}$	2^{32}	$2^{133.67}$	MitM	[DF13, DF15]
7/14	2^{99}	2^{97}	2^{98}	MitM	[DFJ13]
8/14	$2^{205.8}$	$2^{34.2}$	$2^{205.8}$	MitM	[DS08a]
8/14	$2^{227.8}$ MA	$2^{111.1}$	$2^{116.1}$	Imp. diff.	[LDKK08]
8/14	$2^{229.7}$ MA	$2^{89.1}$	2^{101}	Imp. diff.	[LDKK08]
8/14	2^{196}	2^{113}	2^{129}	MitM	[DKS10a, DKS15]
8/14	$2^{234.17}$	2^{8}	$2^{234.17}$	MitM	[DF13, DF15]
8/14	2^{195}	2^{32}	$2^{193.34}$	MitM	[DF13, DF15]
8/14	2^{156}	$2^{102.83}$	$2^{140.17}$	MitM	[DF13, DF15]
8/14	2^{156}	2^{113}	2^{130}	MitM	[DF13, DF15]
8/14	2^{196}	2^{107}	2^{96}	MitM	[DFJ13]
9/14	$2^{254.17}$	2^{32}	$2^{254.17}$	MitM	[DF13, DF15]
9/14	2^{203}	2^{120}	2^{203}	MitM	[DFJ13]
9/14	$2^{251.9}$	2^{120}	2^{8}	Bicliques	[BKR11a]
11/14	$< 2^{254.4}$	$2^{122.4}$	2^{45}	Imp. diff.	[Yua10]
14 (full)	$2^{254.4}$	2^{40}	2^{8}	Bicliques	[BKR11a]

Table 3.5: Related Key Cryptanalysis of AES – Selected Published Results

	Version and Rounds Attacked	Attack Complexity Time	Data	Memory	Keys	Technique	Reference
AES-192	9/12	2^{182}	2^{85}	n/r	64	Partial Sums	[BDK05, KHP07]
	10/12	2^{182}	2^{125}	n/r	256	Rectangle	[KHP07]
	10/12	2^{183}	2^{124}	n/r	64	Rectangle	[KHP07]
	12 (full)	2^{176}	2^{123}	2^{152}	4	Amplified Bmrg.	[BK09]
AES-256	8/14	$2^{26.5}$	$2^{26.5}$ CC	$2^{26.5}$	2	Differential	[BDK+10]
	8/14	2^{31}	2^{31}	2	2	Differential	[BDK+10]
	9/14	$5 \cdot 2^{224}$	2^{85}	2^{32}	256	Partial Sums	[FKL+00]
	9/14	2^{32}	2^{32} CC	2^{32}	2	Differential	[BDK+10]
	9/14	2^{39}	2^{39}	2^{32}	2	Differential	[BDK+10]
	10/14	$2^{171.8}$	$2^{114.9}$	n/r	256	Rectangle	[BDK05, KHP07]
	10/14	2^{45}	2^{44} CC	2^{33}	2	Differential	[BDK+10]
	10/14	2^{49}	2^{48}	2^{33}	2	Differential	[BDK+10]
	11/14	2^{70}	2^{70}	2^{33}	2	Differential	[BDK+10]
	13/14	2^{76}	2^{76}	2^{76}	4	Boomerang	[BK10]
	14 (full)	2^{131}	2^{131}	2^{65}	2^{35}	Differential	[BKN09]
	14 (full)	$2^{99.5}$	$2^{99.5}$	2^{77}	4	Boomerang	[BK09]

Bmrg: Boomerang

Table 3.6: Low Data Cryptanalysis of AES (all variants) – Selected Published Results

Rounds Attacked	Attack Complexity Time	Data	Memory	Technique	Reference
2.5	2^{80}	2 KP	2^{80}	G&D–MitM	[BDF11]
3	2^{32}	2	2^{1}	Diff.–MitM	[BDD+12]
2.5–3	$2^{31.6}$	2	2^{8}	Trunc. Diff.	[GRR16a, GRR16b]
2.5	2^{24}	2	2^{16}	G&D–MitM	[BDF11]
3	2^{16}	2	2^{8}	G&D–MitM	[BDF11]
2.5–3	$2^{11.2}$	3	1	Trunc. Diff.	[GRR16a, GRR16b]
3	2^{8}	3	2^{8}	G&D–MitM	[BDF11]
2.5–3	$2^{25.7}$	3	2^{12}	Trunc. Diff.	[GRR16a, GRR16b]
4	2^{104}	2	1	Diff.–MitM	[BDD+12]
3.5–4	2^{96}	2	1	Trunc. Diff. (EE)	[GRR16a, GRR16b]
4	2^{88}	2	2^{8}	G&D–MitM	[BDF11]
4	2^{80}	2	2^{80}	G&D–MitM	[BDF11]
3.5	2^{72}	2	2^{72}	G&D–MitM	[BDF11]
3.5–4	$2^{74.7}$	3	1	Trunc. Diff. (EE)	[GRR16a, GRR16b]
4	2^{72}	3	2^{8}	G&D–MitM	[BDF11]
3.5–4	$2^{69.7}$	3	2^{12}	Trunc. Diff. (EE)	[GRR16a, GRR16b]
4	2^{32}	4	2^{24}	G&D–MitM	[BDF11]
4	2^{64}	5	2^{68}	Diff.–MitM	[BDD+12]
3.5–4	2^{38}	8	2^{15}	I-Pol	[Tie16]
4	2^{40}	10	2^{43}	Diff.–MitM	[BDD+12]
3.5–4	$2^{35.1}$	24	2^{17}	Trunc. Diff. (EB)	[GRR16a, GRR16b]
3.5–4	2^{14}	29	small	Integral	[DKR97]

G&D: Guess and Determine – Trunc. Diff.: Truncated Differentials
I-Pol: Impossible Polytopic – EE: Extension at End – EB: Extension at Beginning

3.20.5 Intellectual Property

The designers did not patent any aspect of the cipher, which is also not encumbered by previous intellectual property.

3.21 The NESSIE Block Cipher Selection

NESSIE (New European Schemes for Signatures, Integrity and Encryption) was a European research project funded from 2000–2003 to identify secure cryptographic primitives. The project was comparable to the NIST AES selection process (Section 3.19 on page 175) and the Japanese Government-sponsored CRYPTREC project (Section 3.22 on page 192) – but the three projects did not necessarily reach the same conclusions. The NESSIE participants include some of the foremost active cryptographers in the world, as does the CRYPTREC project.

NESSIE's goal was to identify and evaluate quality cryptographic designs in several categories. To that end a public call for submissions was issued in March 2000. Forty-two submissions were received, and in February 2003 twelve submissions were selected. In addition, five publicly known algorithms that were not submitted to the project were chosen as "selectees." The project has publicly announced that *"no weaknesses were found in the selected designs."*

Details for all submissions can be obtained from this web page. Of the forty-two submissions,

seventeen were block ciphers:

- Six 64-bit block ciphers: CS-Cipher [SV98], Hierocrypt-L1 [OMSK00, Tos01], IDEA (Section 3.6 on page 142), Khazad [BR02], MISTY-1 (see Section 3.18 on page 170), and Nimbus [Mac00];
- Seven 128-bit block ciphers (none of which coming from the AES process): Anubis (Subsection 3.21.3 on page 190), Camellia (Section 3.18 on page 170), Grand Cru [Bor00], Hierocrypt-3 [OMSK00, Tos02], Noekeon (Subsection 3.21.4 on page 191), Q, and SC2000 [SYY+01] (see also Section 3.22 on page 192);
- One 160-bit block cipher: SHACAL (see below); and
- Three block ciphers with a variable block length: NUSH [LV00] (64, 128, and 256 bits), RC6 (at least 128 bits, see Section 3.15 on page 165), and SAFER++ (64 and 128 bits, see Section 3.8.3 on page 150).

The NESSIE list of recommended block ciphers includes: MISTY-1, Camellia, SHACAL-2, and AES (Section 3.20 on page 180).

With the exception of SHACAL, all the recommended block ciphers are described elsewhere in this document. SHACAL will be described next, together with submissions Khazad, Anubis, and NOEKEON.

3.21.1 SHACAL

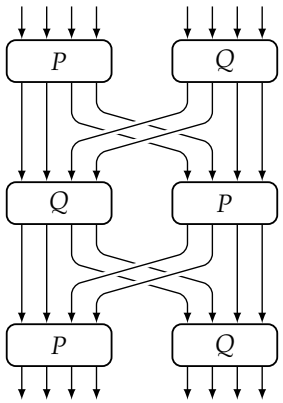
SHACAL is a family of block ciphers developed by Gemplus, introduced by Helena Handschuh and David Naccache.

SHACAL-1 is a 160-bit block cipher based on the hash function SHA-1. SHA-1 is designed around a *compression function*, that takes as input a 160-bit state and a 512-bit data word and outputs a new 160-bit state after 80 rounds. The hash function works by repeatedly calling this compression function with successive 512-bit data blocks and each time updating the state accordingly. For a fixed 512-bit data block, the transformation of the state is invertible, so the compression function is in fact a block cipher where the 512-bit data word is the key used to encrypt the 160-bit state. If keys shorter than 512 bits are to be used, these are first just padded with zeros.

All full round attacks on SHACAL-1 to date are related-key attacks. The best one key attack so far, by Jiqiang Lu, Jongsung Kim, Nathan Keller, and Orr Dunkelman in [LKKD06], breaks 52 internal rounds in time 2^{493} requiring about 2^{160} known plaintexts.

In [Saa03], Markku-Juhani Saarinen observed that it is possible to construct slid pairs in the compression function of SHA-1 using about 2^{97} chosen chaining values (for two different blocks of message). Saarinen used the slid pairs to mount a related-key distinguishing attack against SHACAL-1 requiring 2^{97} chosen plaintexts encrypted under two related keys and time complexity of 2^{97} encryptions. Eli Biham and Orr Dunkelman and Nathan Keller in [BDK07b] exploit the property found by Saarinen to mount a related-key key recovery attack on the full SHACAL-1, using from two to eight related keys. If k related keys are used, their attack has complexity $\max\left\{(k-1)\cdot 2^{98.5}, 2^{510-(k-1)\cdot 62}\right\}$. When a maximum of eight related keys are used, the complexity becomes $2^{101.3}$.

Figure 3.26: The Khazad/Anubis Composite S-box



SHACAL-2 has a 256-bit block size and is based on SHA-256. It has not been broken yet. The best attack, by Jiqiang Lu and Jongsung Kim, is a related-key attack that breaks 44 rounds out of 80 [LK08].

3.21.1.1 Intellectual Property

In the NESSIE submission package, the designers of the cipher declare: *"We do not intend to apply for any patent covering SHACAL and undertake to up date the NESSIE project whenever necessary."* To our knowledge, no updates have been submitted to the NESSIE project.

3.21.2 Khazad (and Shark)

Paulo S.L.M. Barreto and Vincent Rijmen designed Khazad, a 64-bit block cipher with a 128-bit key, and submitted it to the NESSIE project in the year 2000. The specification can be found at the NESSIE submission web page. Khazad is a SPN designed according to the wide trail strategy. Although it is not a Feistel cipher, the inverse operation of the cipher differs from the forward operation in the key scheduling only. This is achieved choosing all round transformation components to be involutions. This property makes it possible to reduce the required chip area in a hardware implementation, as well as the code and table size.

The cipher shares some design aspects with Rijndael, in that the internal state is represented as 8 bytes, each byte is an element of $\mathbb{F}_{2^8}$, diffusion is provided by an 8×8 involutory MDS matrix over F_{2^8} with elements of low Hamming weight.

The S-box is of course also involutory and is recursively constructed using two smaller 4×4-bit S-boxes, as depicted in Figure 3.26. The two 4×4 components are involutory themselves and have been generated pseudo-randomly. This S-box construction has left some legacy: the block cipher CLEFIA (Section 3.28 on page 201) also defines a 8-bit S-box recursively using two smaller 4×4-bit S-boxes, but the construction is different. The two "mini boxes" (as they are

Table 3.7: Differences between Khazad and Shark

	SHARK	KHAZAD
Rounds	6	8
Key schedule	Affine mapping derived from the cipher itself in CFB mode	Feistel key evolution using the cipher round function itself
$\mathbb{F}_{2^8}$ definition polynomial	$x^8+x^7+x^6+x^5+x^4+x^2+1$ ($1F5_x$)	$x^8+x^4+x^3+x^2+1$ ($11D_x$)
S-box	Inversion in $\mathbb{F}_{2^8}$, followed by affine transformation	Recursive structure
Origin of diffusion matrix	Reed-Solomon code	Involutional MDS code

called by the authors) are defined as follows:

x:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$P[x]$:	3	F	E	0	5	4	B	C	D	A	9	6	7	8	2	1
$Q[x]$:	9	E	5	6	A	2	3	C	F	0	4	D	7	B	1	8

The diffusion layer is a linear mapping $\theta : \mathbb{F}_{2^8}^8 \mapsto \mathbb{F}_{2^8}^8$ corresponding to the $[16, 8, 9]$ MDS code with generator matrix $G_H = [I\,H]$, where $H = \mathrm{had}(01_x, 03_x, 04_x, 05_x, 06_x, 08_x, 0b_x, 07_x)$, is a Hadamard matrix

$$H = \begin{pmatrix} 01_x & 03_x & 04_x & 05_x & 06_x & 08_x & 0B_x & 07_x \\ 03_x & 01_x & 05_x & 04_x & 08_x & 06_x & 07_x & 0B_x \\ 04_x & 05_x & 01_x & 03_x & 0B_x & 07_x & 06_x & 08_x \\ 05_x & 04_x & 03_x & 01_x & 07_x & 0B_x & 08_x & 06_x \\ 06_x & 08_x & 0B_x & 07_x & 01_x & 03_x & 04_x & 05_x \\ 08_x & 06_x & 07_x & 0B_x & 03_x & 01_x & 05_x & 04_x \\ 0B_x & 07_x & 06_x & 08_x & 04_x & 05_x & 01_x & 03_x \\ 07_x & 0B_x & 08_x & 06_x & 05_x & 04_x & 03_x & 01_x \end{pmatrix},$$

so that $\theta(a) = H \cdot {}^t a$. About the notation for the matrix enties: the hexadecimal numbers represent the field elements of $\mathbb{F}_{2^8} = \mathbb{F}[x]/(x^8+x^4+x^3+x^2+1)$ as bytes, where the i^{th} bit is the coefficient of (the image of) x^i. For instance, 01_x represents 1 and 03_x represents (the image of) $x+1$ in $\mathbb{F}_{2^8}$. A simple inspection shows that matrix H is symmetric and unitary. Therefore, θ is an involution.

Shark [RDP+96] is very similar to Khazad and therefore we do not perceive the need to treat both: the main differences are summarized in Table 3.7.

Also Khazad (and Anubis) left a strong legacy on the design of CLEFIA (Section 3.28 on page 201) since one of the two S-boxes of the latter cipher is recursively constructed from smaller 4×4 S-boxes, and the diffusion matrices are Hadamard matrices as well.

3.21.2.1 Cryptanalysis

Alex Biryukov has analysed involutional ciphers in [Bir03], including Khazad and Anubis. He found that there is a variant of the slide with a twist attack (cf. Subsection 2.6.3.2 on page 111) which can be applied to this type of designs, i.e. ciphers with involutory data encryption path where encryption and decryption necessarily differ only in the key schedule, and the round are all equal (with the exception of a final round without linear mixing layer). The attack can only work if some degree of symmetry can be enforced on the key schedule, which means that the keys must be specially selected, i.e. they form a specific class of weak keys (see Section 2.5 on page 105). An an application he broke five of Khazad's eight rounds in time 2^{40} requiring $O(2^{32})$ memory – for one in 2^{64} keys.

Frédéric Muller [Mul03] has discovered an attack which can break five of Khazad's eight rounds for all keys in time 2^{91}.

3.21.2.2 Advantages

Khazad can be implemented both in SW and HW in a very compact way. HW implementations can be both very performing and small: As it can be seen in Table 4.1 on page 235, even a fully unrolled and single cycle implementation of Khazad can be much smaller than other comparable ciphers.

3.21.2.3 Disadvantages

None in particular.

3.21.2.4 Intellectual Property

In the NESSIE submission package, the designers of the cipher declare: *"to the best of our knowledge the practice of the Khazad algorithm, as well as the reference implementations we have submitted, are not covered by any patents or patent applications worldwide."*

3.21.3 Anubis

Anubis (the specification can be found at the NESSIE submission web page), is a block cipher designed by Vincent Rijmen and Paulo S.L.M. Barreto that operates on data blocks of length 128 bits, and uses keys of length 128 to 320 bits in steps of 32 bits. It is another member of the SQUARE/Rijndael/Khazad family, and the main differences w.r.t. Rijndael are summarized in Table 3.8 on the next page.

The S-box is the same recursively defined S-box used in Khazad. The diffusion matrix is also a Hadamard matrix.

There is to date no attack that has broken the cipher. The cipher was not admitted to round two of the NESSIE selection process only because it was deemed too similar to Rijndael.

3.21.3.1 Intellectual Property

In the NESSIE submission package, the designers of the cipher declare: *"to the best of our knowledge the practice of the Anubis algorithm, as well as the reference implementations we have submitted, are not covered by any patents or patent applications worldwide."*

Table 3.8: Differences between Anubis and Rijndael

	Rijndael	Anubis
Key size (bits)	128, 192, or 256	128, 160, 192, 224, 256, 288, or 320
Block size (bits)	128, 192, or 256	always 128
Number of rounds	10, 12, or 14	12, 13, 14, 15, 16, 17, or 18
Key schedule	dedicated a priori algorithm	key evolution based on variant of round function, and extraction using linear projection
$\mathbb{F}_{2^8}$ definition polynomial	$x^8 + x^4 + x^3 + x + 1$ ($11B_x$)	$x^8 + x^4 + x^3 + x^2 + 1$ ($11D_x$)
S-box	Inversion in $\mathbb{F}_{2^8}$, plus affine transformation	Recursive structure
Origin of the round constants	polynomials x^i over $\mathbb{F}_{2^8}$	successive entries of the S-box

3.21.4 NOEKEON

NOEKEON was designed and submitted to NESSIE by Joan Daemen, Michaël Peeters, Gilles Van Assche, and Vincent Rijmen. This SPN cipher has 128-bit key and block sizes, with 16 rounds.

The cipher is extremely compact in both SW and HW. It is designed to be implemented using only bit-wise Boolean operations and (cyclic) shift operations, similarly to 3-WAY and BASEKING, which are described in Joan Daemen's PhD Thesis [Dae95] and the AES proposal Serpent (Section 3.17 on page 168). Because of this it is possible to implement efficient DPA-resistant software implementations. Also, since several operations operate in parallel on 32 nibbles, bitslicing implementations in SW or highly parallel HW architectures are possible.

Decryption can be performed by the same circuit/program as encryption.

The round function is composed of following steps:

1. Key mixing, a linear transformation, further key mixing;
2. A non-linear function called Γ, sandwiched between two bit permutations.

Γ can be expressed as a 4-bit S-box or as a compact straight line program – in fact all non-linearity is provided by the binary AND operator. The Γ layer can be implemented using bitslicing, allowing for very efficient implementations.

One peculiarity of NOEKEON is that the cipher can work in two different modes which can be effectively considered as two different ciphers: "Direct-key mode" NOEKEON is to be used for maximum efficiency where related-key attacks are not possible, and "indirect-key mode" NOEKEON would be used when protocols are employed that can make related-key attacks possible. The difference between the two modes is that in indirect-key mode a key schedule algorithm is used that is based on the encryption cipher itself, whereas direct-key mode just reuses the secret key as round key.

During the NESSIE selection process, Lars R. Knudsen and Håvard Raddum [KR01b] raised

doubts about NOEKEON's resistance under related-key attacks even if the key schedule is used. As a result the cipher was not selected. Despite this, no concrete attack has been published and the best cryptanalysis so far seems to be [ZRHD08], which breaks five rounds of NOEKEON with a variant of integral attacks.

The complete specification of the cipher can be downloaded at the cipher's web site [DPAR].

3.21.4.1 Intellectual Property

In the NESSIE submission package, the designers of the cipher declare: *"to the best of our knowledge the practice of the Noekeon algorithm, as well as the reference implementations we have submitted, are not covered by any patents or patent applications worldwide."*

3.22 The CRYPTREC Block Ciphers Recommendations

CRYPTREC is the Cryptography Research and Evaluation Committees set up by the Japanese Government to evaluate and recommend cryptographic techniques for government and industrial use. It is comparable in many respects to the European Union's NESSIE project (Section 3.21 on page 186) and to the Advanced Encryption Standard process run by NIST in the US (Section 3.19 on page 175).

The project is constantly reviewing the recommendations by asking several important cryptographers to deliver regular security evaluations:
http://www.cryptrec.go.jp/english/estimation.html.

In August 2003 the project published a draft containing several recommendations. The recommended block ciphers were:

- The four 64-bit ciphers CIPHERUNICORN-E [NEC13], Hierocrypt-L1 [OMSK00, Tos01], MISTY-1 (see Section 3.18 on page 170), 3-key Triple DES (permitted "for the time being" if used as specified in FIPS Pub 46-3, and if specified as a de facto standard); and
- The five 128-bit ciphers AES (Section 3.20 on page 180), Camellia (Section 3.18 on page 170), CIPHERUNICORN-A [NEC13], Hierocrypt-3 [OMSK00, Tos02], and SC2000 [SYY+01].

CRYPTREC explicitly states that 128-bit block ciphers are preferred whenever possible. In 2013 the list of recommended ciphers was considerably shortened, and now it consist only of: 3-key Triple DES, AES and Camellia (cf. http://www.cryptrec.go.jp/english/method.html).

3.22.1 Remarks on some of the ciphers

1. CIPHERUNICORN-E [NEC13] is a standard Feistel network with the added twist that every two rounds there is an additional round called an L-Function that mixes key material in a non-linear way. The F-function is very elaborate and uses four different 8-bit S-boxes that are all based on inversion in F_{2^8} combined with affine transformations, but the the definition polynomials of the field is different for each S-box.
2. SC2000 [SYY+01] is a block cipher designed by Fujitsu Laboratories and the Science University of Tokyo. It has a block size of 128 bits and key lengths of 128, 192, and 256 bits. It is a complex product cipher where three types of operations, called I, B and R, are alternated

in a regular pattern that is equal to its inverse. These functions operate on the state split as four 32-bit branches. They borrow from different design methodologies:

- The I function is just key mixing by XOR.
- The R function is a 4-branch Feistel-type round where two branches are first passed through an array of four 5-bit and two 6-bit S-boxes, then mixed linearly, and then some bits of the two results are intermixed (the L function, described later). Finally, the two results are XORed to the other two branches. In the cipher, the two pairs of branches are exchanged and there is another repetition of the R function.
- The L function maps two 32-bit values as follows:

$$(a, b) \mapsto \left(\left(a \wedge \text{mask}\right) \oplus b, \left(b \wedge \overline{\text{mask}}\right) \oplus a\right)$$

  where the mask can take one of the two values $\texttt{55555555}_x$ and $\texttt{33333333}_x$. In each pair of R rounds the same mask is used. The masks are alternated between different R round pairs.
- The B function consists of an input linear transformation, an array of eight equal 4-bit S-boxes, and an output linear transformation.

SC2000 is one of the very few ciphers around that mixes SPN-like and Feistel-type rounds in the data obfuscation path. All the components of the cipher have been designed to allow for efficient bit-slicing implementations.

For a 128-bit key there are 7 rounds of the B function and 12 rounds of the R function. For larger keys, there are 8 rounds of the B function and 14 rounds of the R function. The number of I rounds is always twice that of the B rounds. The alternating pattern of the rounds is $(I, B, I, R \times R)^*, \ldots, I, B, I$ (where $\times$ denotes the Feistel branch swap).

In the cryptanalytic literature an $(I, B, I, R \times R)$-block is considered a round and the final (I, B, I) a half-round, so with this terminology SC2000 consists of 6.5 or 7.5 rounds depending on the key size. The cipher has not been broken, however differential attacks exist against 5-round reduced SC2000 [Lu10, Lu11], and therefore the residual security margin is quite small.

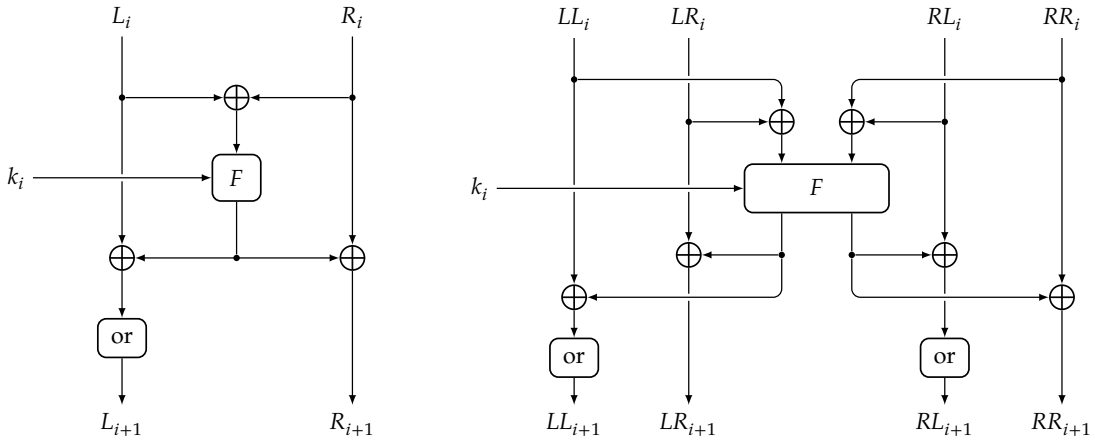
3.23 IDEA NXT (FOX)

IDEA NXT is a block cipher designed by Pascal Junod and Serge Vaudenay between 2001 and 2003. The cipher was originally named FOX and was disclosed in 2003 and published in 2004 [JV04a]. In May 2005 it was announced by MediaCrypt under the name IDEA NXT, as the successor of IDEA.

IDEA NXT has 64- or 128-bit blocks, and a variable key size up to 256 bits. It is a 16-round Lai-Massey scheme with an ortohomorphism, but versions with up to 255 rounds can be defined. IDEA NXT-64, resp. NXT-128 divides the state into two, resp. four 32-bit branches.

The rounds of NXT-64 and NXT-128, depicted in Figure 3.27 on the following page, are constructed around similar round functions F. The NXT-64 round function accepts a 32-bit input

Figure 3.27: Rounds of IDEA NXT-64 and NXT-128



and a 64-bit round key. The NXT-128 round function accepts a 64-bit input and a 128-bit round key. Inspired by SPNs, F is built as follows:

1. A key mixing layer using the first half of the round key;
2. An S-box layer based on four, resp. eight 8-bit S-boxes;
3. A linear diffusion layer based on an 4×4, resp. 8×8 MDS Matrix over $\mathbb{F}_{2^8}$;
4. A second key mixing layer using the second half of the round key;
5. A second S-box layer;
6. A third key mixing layer that reuses the first half of the round key.

The S-box is itself a 3-round Lai-Massey scheme with two 4-bit branches whose round functions are pseudo-randomly generated permutations on $\mathbb{F}_{2^4}$.

The ortohomorphism is a 1-round Feistel scheme with the identity as round function (i.e. it is a map $u||v \mapsto (u \oplus v)||u$ where u and v are two halves of the input). It is omitted in the last round of the cipher and also in the third round of the S-box.

The key-schedule algorithms are complex and designed to be very strong. There are three different versions, two for the 64-bit cipher and one for the 128-bit cipher. They recycle the components of the round functions.

3.23.1 Advantages

Encryption and decryption are the same algorithm, where only the round keys have to be used in the reverse order, leading to savings in code size and area.

3.23.2 Disadvantages

The time to compute the subkeys is quite high (approximately equal to the time to encrypt 6 blocks, or 12 blocks for the "stronger" key schedule for the 64-bit cipher).

3.23.3 Intellectual Property

Patent applications cover the encryption path (U.S. Patent Application 0040247117) and the key schedule (U.S. Patent Application 0050053233).

These patent applications and the need for a license in order to use IDEA NXT may influence the extent of its adoption, particularly given that there are viable public domain alternatives, e.g., AES, Serpent and the Twofish algorithm, and have no restrictions on them whatsoever.

3.24 ICEBERG

François-Xavier Standaert et al in 2004 introduced ICEBERG [SPR+04].

ICEBERG operates on 64-bit blocks and uses a 128-bit key. It is a involutional 16-round SPN designed to be efficient when implemented on reconfigurable hardware. In particular, the design goal was to minimise the number of 4-but LUT (look up tables) which are the basic logic block of FPGAs.

ICEBERG uses two different 4-bit S-boxes called S_0 and S_1.

A round consists of a non-linear substitution layer, and a two part linear diffusion layer that sandwiches the key mixing.

The non-linear substitution layer operates on bytes by essentially building a 8-bit S-box out of the two involutory 4-bit S-boxes, in a five layer structure similar to the Khazad/Anubis construction (cf. Subsection 3.21.2 on page 188), with the same bit wiring among each two S-box layers, but the same S-box is applied to the two nibbles of a byte, first S_0, then S_1, then again S_0 – as opposed to both S-boxes being used in the same layer.

The linear diffusion cum key mixing layer consists of: bit wiring of the whole 64-bit state, via a permutation called P64; parallel application of multiplication of each nibble with a 4×4 involutory matrix; key mixing; swap of each two adjacent bits; another application of P64.

The key schedule is simple but it is not trivial. It is made non-linear and non-periodic by combining S-box applications, variable shifts and bit permutations. It is also symmetric, i.e. the master key is expanded into a sequence $k_0, k_1, \ldots, k_7, k_8, k_7, \ldots, k_0$.

The only difference between encryption and decryption is in the key schedule. The same circuit can be used for both cipher directions by modifying a single configuration bit for the key schedule part.

The design goals of ICEBERG have been achieved. ICEBERG needs a total of 704 LUTs, whereas implementing Khazad needs 1344 LUTs according to [SRQL02] and implementing AES requires 3376 LUTs according to [SRQL03b] (we note, however, that AES can be implemented with as few as 877 LUTs if 10 RAM blocks are employed on some FPGAs, cf. [SRQL03a]).

Efficiency in software implementations is not one of the design goals of ICEBERG. But table lookups can be used to obtain efficient round functions using the method described in [PA93] attaining performance (as claimed by the authors) comparable to Khazad. ICEBERG can also be implemented in bit-slice mode following [Bih97b].

3.24.1 Intellectual Property

We are not aware of any patents on ICEBERG. François-Xavier Standaert privately communicated to us that none of his cipher designs is patented [Sta14].

3.25 mCrypton (and Crypton)

At WISA 2005, Chae Hoon Lim and Tymur Korkishko [LK05] introduced mCrypton, a very small cipher targeted at Low-Cost RFID Tags and Sensors.

It is a 64-bit block cipher with three key size options of 64 bits, 96 bits and 128 bits. It is essentially a "half size" SQUARE or AES. mCrypton processes an 8-byte data block by representing it into a 4×4 nibble (half byte) array. The internal state is processed by four operations: nibble-wise substitution using four different 4-bit S-boxes, column-wise bit permutation, matrix transposition, and key addition. These operations are combined to form a 12-round SPN.

mCrypton and Crypton [Lim99] are very similar, so we describe here only mCrypton as it is more interesting for our purposes. Crypton has a nearly identical structure, but it is byte oriented instead of nibble oriented. We shall limit ourselves to some comments on the differences between the two ciphers.

The four S-boxes of mCrypton come in pairs that invert each other. Therefore there is no need for additional tables or circuits to support decryption beside encryption. In fact, the same circuit or program can be used to implement the data obfuscation part of the cipher, only the key has to be suitably modified. This was true also of Crypton.

The nibble-wise substitution works as follows

$$\begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \mapsto \begin{pmatrix} S_0(a_{0,0}) & S_1(a_{0,1}) & S_2(a_{0,2}) & S_3(a_{0,3}) \\ S_1(a_{1,0}) & S_2(a_{1,1}) & S_3(a_{1,2}) & S_0(a_{1,3}) \\ S_2(a_{2,0}) & S_3(a_{2,1}) & S_0(a_{2,2}) & S_1(a_{2,3}) \\ S_3(a_{3,0}) & S_0(a_{3,1}) & S_1(a_{3,2}) & S_2(a_{3,3}) \end{pmatrix}$$

and thus it commutes with matrix transposition. (Crypton has two such substitutions, which are byte-oriented, and one is used in the odd rounds, the other in the even rounds.)

The column-wise bit permutation is actually a linear combination over $\mathbb{F}_2$ of the entries in a column. The i-th column (with $i = 0, 1, 2, 3$) is transformed using a map π_i. The map π_i is defined as follows: if two columns are represented as $\mathbb{a} = {}^t(a_0, a_1, a_2, a_3)$ and $\mathbb{b} = {}^t(b_0, b_1, b_2, b_3)$ with $\mathbb{b} = \pi_i(\mathbb{a})$, then

$$b_j = \bigoplus_{k=0}^{3} (m_{i+j+k \bmod 4} \wedge a_k) \ ,$$

where the *masks* m_i are defined as

$$m_0 = \mathtt{1110}_2 \, , \quad m_1 = \mathtt{1101}_2 \, , \quad m_2 = \mathtt{1011}_2 \, , \quad \mathtt{m}_3 = \mathtt{0111}_2 \ .$$

(The masks used by Crypton are, correspondingly, 8 bits long.)

Key scheduling is quite easy: first some data is generated through nonlinear S-box transformation and then the other keys are generated through simple rotations. This allows key schedule to be performed in parallel not only with encryption but also with decryption. The same S-

boxes used in the data encryption path are used in the key schedule, reducing coding overhead or memory requirements for tables.

3.25.1 Cryptanalysis

Mohsen Shakiba, Mohammad Dakhilalian and Hamid Mala in [SDM13] apply related-key impossible differential cryptanalysis to 8 and 9 reduced versions of mCrypton-96 and mCrypton-128. The attack on mCrypton-96 requires $2^{59.9}$ chosen plaintexts, and has a time complexity of about $2^{74.9}$ encryptions. The data and time complexities for the attack on mCrypton-128 are $2^{59.7}$ chosen plaintexts and $2^{66.7}$ encryptions, respectively.

Kitae Jeong et al. [JKL+13] manage to apply biclique cryptanalysis to full round mCrypton, the resulting attacks require computational complexities of $2^{63.18}$, $2^{94.81}$ and $2^{126.56}$ for key sizes of 64, 96 and 128 bits, respectively.

3.25.2 Advantages

mCrypton, like Crypton, is very fast. In fact, according to some tests the original cipher Crypton was the fastest AES submission and had also the smallest key setup overhead - what ultimately killed it was the presence of weak keys, as shown by Johan Borst [Bor99].

Encryption and decryption are very similar reducing the overhead to support both: In the paper that introduced the cipher, a straightforward 1 cycle/round HW implementation required 2400 to 3000 gates to support only encryption, and 3500 to 4100 gates for both encryption and decryption. Key scheduling can run in parallel with both encryption and decryption.

3.25.3 Disadvantages

In some scenarios other ciphers could be preferable because the overhead to implement both decryption beside encryption, despite being smaller than a 2x factor, is not negligible.

3.25.4 Intellectual Property

We are not aware of any patents encumbering the use of mCrypton (and Crypton), but, since the IP statements of the AES competition (for Crypton) seem to be no longer archived, we cannot be sure that there is no protected intellectual property either.

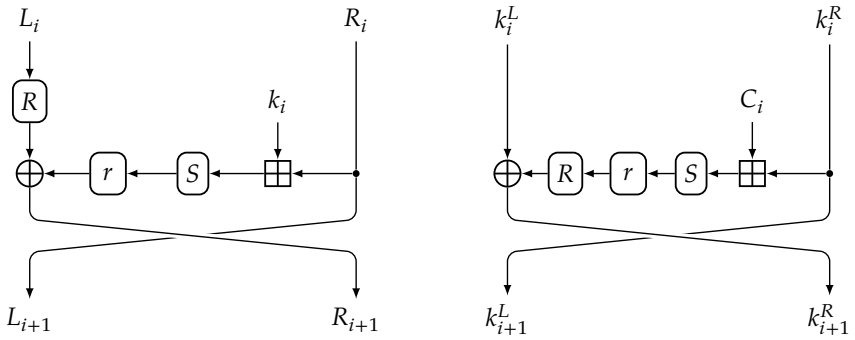
3.26 SEA, the Scalable Encryption Algorithm

At CARDIS 2006, François-Xavier Standaert, Gilles Piret, Neil Gershenfeld and Jean-Jacques Quisquater introduced the Scalable Encryption Algorithm, or SEA for short [SPGQ06].

This is a cipher intended not to be viable for a large variety of platforms - instead, the design assumes very limited processing resources and throughput requirements, targeting very low memory requirements for both data and code, at the price of performance.

An important design choice is that the cipher has parameters that can be adjusted to the characteristics of the target platform. As a result, the cipher $\text{SEA}_{n,b}$ has a block and key size n and a word size b. The word size b is ideally the processor's word size (which can be emulated on architectures with a different word size, but at a price) and n is a multiple of $6b$.

Figure 3.28: Data Encryption and Key Schedule Rounds of SEA



Hence, as the authors examplify, using an 8-bit processor, one can derive 48-, 96-, 144-, ...-bit block ciphers, respectively denoted as $\text{SEA}_{48,8}$, $\text{SEA}_{96,8}$, $\text{SEA}_{144,8}$, and so on.

$\text{SEA}_{n,b}$ is constructed from a reduced number of basic operations operating on words, namely bitwise XOR, AND and OR, addition modulo 2^b, and bit rotation.

The structure of the cipher is a 2-branch balanced Feistel network tweaked with a rotation of a branch. The round key is mixed at the beginning of the F-function. The key schedule is also based on a 2-branch balanced Feistel network where round constants are mixed at the beginning of the F-function as well.

Modular addition is used for the key/round constant mixing, whereas the output of the F-functions are mixed to the target branch using bitwise XOR.

The F-function involves a 3-bit S-box applied in parallel to the input, and a word-wise bit rotation. The 3-bit S-box is applied in a bit-slicing way to each three words of the input, and can be implemented using a short program of logical operations.

Resistance against several known attacks and some variants is also proved in [SPGQ06]. The proofs imply that the required number of rounds to meet the desired resistance level is quite high, namely, the number n_r of rounds is an odd number larger than

$$\frac{3n}{4} + 2(n_b + \lfloor b/2 \rfloor) \ ,$$

where $n_b = n/(2b)$ is the number of words per Feistel branch – and in fact it is suggested in the paper to at least double that number.

3.26.1 Description

In Figure 3.28 single rounds of the data encryption path and of the key schedule of SEA are shown.

Representing a branch as $(x_{n_b-1}, x_{n_b-2}, \ldots, x_1, x_0)$, the building blocks of the rounds are

- R is a rotation of the words by one position, i.e. $x_i \mapsto x_{i+1}$ and $x_{n_b-1} \mapsto x_0$;
- r replaces three adjacent words $x_{3i+2} \| x_{3i+1} \| x_{3i}$ by $(x_{3i+2} \lll 1) \| x_{3i+1} \| (x_{3i} \ggg 1)$;

- S is the bit-slicing S-box

x:	0	1	2	3	4	5	6	7
$S[x]$:	0	5	6	7	4	3	1	2

  It acts on three adjacent words in a bit-sliced way and can be implemented purely by logic operations as follows

$$\begin{aligned} S(x_{3i+2}\|x_{3i+1}\|x_{3i}) &= x'_{3i+2}\|x'_{3i+1}\|x'_{3i} \quad \text{where} \\ x'_{3i} &= (x_{3i+2} \wedge x_{3i+1}) \oplus x_{3i} \ , \\ x'_{3i+1} &= (x_{3i+2} \wedge x_{3i}) \oplus x_{3i+1} \ , \text{ and} \\ x'_{3i+2} &= (x_{3i} \vee x_{3i+1}) \oplus x_{3i+2} \ . \end{aligned}$$

Another interesting aspect of SEA is that the key schedule is symmetric.

The round keys k_i for $0 \leqslant i \leqslant \lfloor n_r/2 \rfloor$ are given by $k_i = k_i^R$, whereas for $\lfloor n_r/2 \rfloor < i < n_r$ the same sequence is used in reverse order, starting with $k_{\lfloor n_r/2 \rfloor+1} = k^R_{\lfloor n_r/2 \rfloor-1}$. C_i is a n_b word vector of which all the words have value 0 excepted the least significant word, that equals i – hence only one word is effectively modified by a modular addition.

The round key sequence can be generated entirely in parallel with encryption (and decryption) without having to store previous values, using a nifty programming trick that consists in undoing the Feistel swap of the key schedule after the middle round and using the round constants in reverse order, thus undoing each partial "encryption" step. Therefore, for $\lfloor n_r/2 \rfloor < i < n_r$ one can use $k_i = k_i^L$. More details and an example implementation can be found in [SPGQ06].

The fact that the same round keys are used in two different orders is a trait in common with the GOST block cipher (cf. Section 3.4 on page 138) and it is the same sequence used in the cipher ICEBERG [SPR+04] (Section 3.24 on page 195).

The SEA F-functions bear some resemblance to the GOST F-function in that they consist of modular addition followed by an S-box layer and rotation – however these operations are also fundamentally different in some crucial aspects: the rotations are either "local" inside the words or are essentially word based permutations; the S-box layer, by working simultaneously on three words in parallel also contributes to diffusion, together with the fact that the amounts of the subsequent rotations inside the same word triplets.

3.26.2 Cryptanalysis

To the best of our knowledge there has been no successful cryptanalysis of SEA.

3.26.3 Advantages

SEA can be implemented easily with minimal code and data footprint. It is adaptable to various architectures.

3.26.4 Disadvantages

SEA is quite slow and with a high latency, due to the very high number of rounds – this is only partially mitigated by the simplicity of each round.

On very limited architectures such as the ATMEL ATtiny45, it performs better than some other ciphers, such as DESXL and KATAN, and it is less than 3 times slower than HIGHT (Section 3.27), mCrypton (Section 3.25 on page 196) and NOEKEON (Subsection 3.21.4 on page 191) as one can see in Table 4.2 on page 238, that is taken from [EGG+12]. However, despite having a very small code footprint and using very little memory, it does not have a strong advantage over the other credited ciphers, and it is on the power-hungry side. The tests reported in Table 4.4 on page 240 (from [CMM13]), performed on a TI MSP 430 16-bit RISC microcontroller, do not paint a better picture, with ciphers such as TWINE (Section 3.34 on page 213) and XTEA (Section 3.12 on page 159) being faster, requiring less RAM and having a smaller code footprint, and several others, such as HIGHT, showing performance and RAM usage with code only about 10% bigger.

3.26.5 Intellectual Property

SEA is not patented [Sta14].

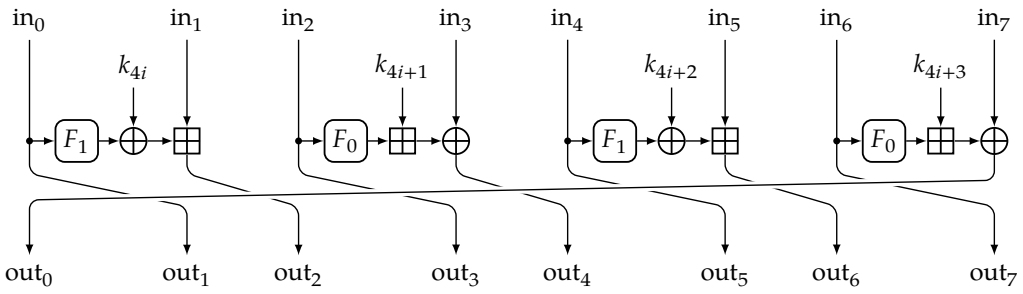
3.27 HIGHT

Introduced in 2006 by a team from the Center for Information Security Technologies (CIST), Korea University, the Department of Mathematics, University of Seoul, and the South Korean National Security Research Institute (NSRI), HIGHT [HSH+06] is an ultralightweight a block cipher with 64-bit block length and 128-bit key length. HIGHT is used as a standard encryption algorithm (TTAS.KO-12.0040) in South Korea.

It was designed for low-resource hardware implementation, such as a sensor nodes or RFID tags, but at the same time withstand the same level of cryptanalysis of a generic encryption algorithm. In fact, HIGHT was designed to withstand several types of cryptanalysis: nearly all types of known cryptanalytic methods are discussed in the original paper.

The cipher is a 8-branch type-2 Feistel network. All branches are 8-bit wide. It consists of 32 rounds with pre and post key whitening, where some of the keys are added, other are XORed to bytes of the state. A round is depicted in Figure 3.29.

Figure 3.29: A Round of HIGHT



The functions F_0 and F_1 are here linear functions that guarantee good diffusion, the nonlinearity being ensured only by the use of algebraically incompatible arithmetic operations.

The key schedule generates the whitening and round keys by bit rotation and addition modulo 256 of predefined constants. There seems to be no hidden rationale for the generation of these constants: these are generated by a simple LFSR over $\mathbb{F}_{2^7}$, another incompatible algebraic structure.

3.27.1 Cryptanalysis

The best attacks so far are impossible differential attacks. In [öVTK09] a 26-round impossible differential attack with time complexity of $2^{119.53}$ (reduced round) encryptions is described, that is improved to $2^{114.35}$ in [CWP12], where 27 rounds can also be attacked, but with time only slightly better than brute force. In [öVTK09] 31 rounds are broken by a related-key variant of the attack with time only slightly better than exhaustive search.

3.27.2 Intellectual Property

We are not aware of patents encumbering the use of HIGHT.

3.28 CLEFIA

CLEFIA is a proprietary block cipher algorithm, jointly developed by Sony and the University of Nagoya and presented in 2007 [SSA+07, SON07, SON10]. SONY has a website dedicated to this algorithm. http://www.sony.net/Products/cryptography/clefia/

The name is derived from the French word *clef*, meaning "key." It has been standardized as the lightweight block cipher at the 128-bit block size level as ISO/IEC 29192 (Lightweight Cryptography) Part 2: Block ciphers (PRESENT was chosen for the 64-bit block size level).

It is a 128-bit block cipher with a 4-branch Type 2 Feistel network. Each branch is 32 bits wide. The rationale for using this design can be summarised in the fact that the F-functions are smaller than in a corresponding 2-branch design and that there are more opportunities for parallel processing. However, it also tends to require more rounds than the traditional Feistel structure, and to counter this issue the Diffusion Switching Mechanism principle (see below) is adopted in the cipher.

The overall structure of the encryption algorithm is represented in Figure 3.30: The design is in our opinion extraordinarily elegant. CLEFIA has 18, 22 and 26 rounds for 128-bit, 192-bit and 256-bit keys, respectively. It uses whitening keys (*wk*) at the beginning and the end of the cipher, and round keys (*rk*). In each round two different F-functions are used: These are formed by key mixing, an S-box layer and diffusion by means of multiplication with a 4×4 matrix over $\mathbb{F}_{2^8}$. There are two different matrices for the two F-functions. The common structure of the two F-functions is represented in Figure 3.31 on the next page: The figure represents the F-function F0; The 32-bit input is split into four 8 bit values, which are rejoined after the linear transform M_0 on the vector formed by the four outputs of the S-boxes. F1 is similar, with S_0 and S_1 exchanged and a different linear transform M_1 in place of M_0.

An especially interesting design decision is that of the two S-boxes S_0 and S_1. Whereas S_1 is similar to the Rijndael S-box (inversion in $\mathbb{F}_{2^8}$ composed with affine transformations), S_0 is

Figure 3.30: CLEFIA's Feistel Structure

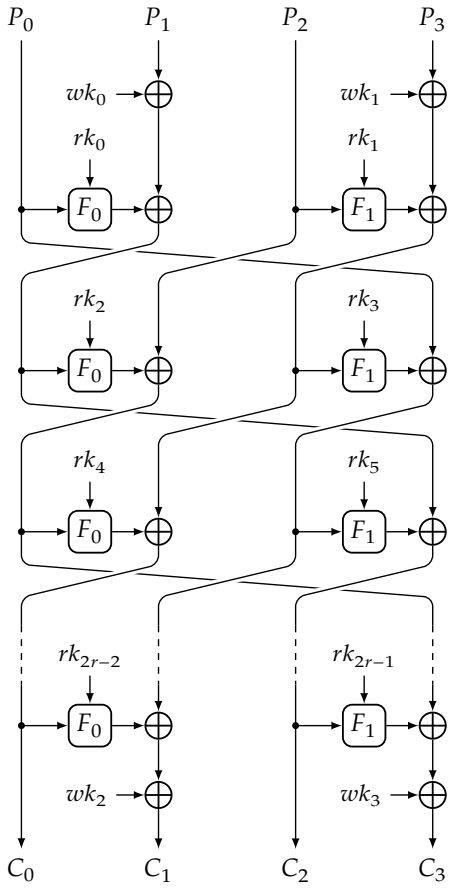


Figure 3.31: CLEFIA's F-function F0

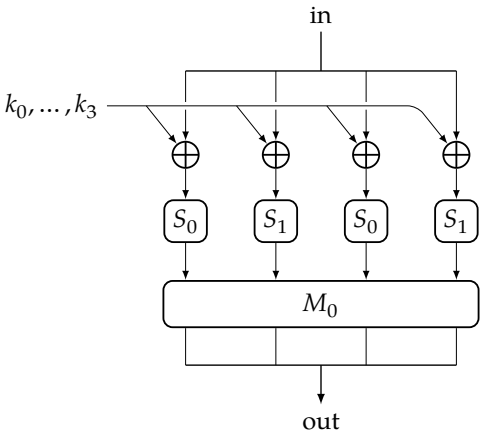
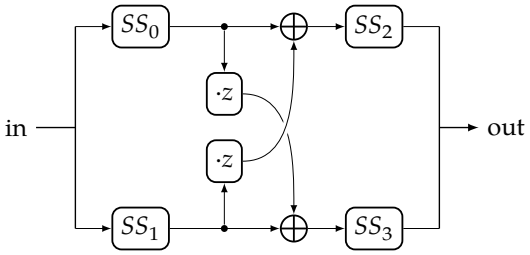


Figure 3.32: CLEFIA's First S-box S_0



defined recursively by combining four smaller 4×4-bit S-boxes S SS_i for $i = 0, 1, 2, 3$. as shown in Figure 3.32. The "multiplication by z" is multiplication by (the image of) z in $\mathbb{F}_{2^4} := \mathbb{F}[z]/(z^4 + z + 1)$. As a result of the two very different design approaches, the two S-boxes have different strengths, as seen in Table 3.9.

Table 3.9: Security Parameters of the Clefia S-boxes

	S_0	S_1
Max. differential prob.	$2^{-4.67}$	$2^{-6.00}$
Max. linear prob.	$2^{-4.38}$	$2^{-6.00}$
Min. algebraic degree	6	7
Min. number of terms over $\mathbb{F}_{2^8}$	244	252

The use of two different matrices follows the Diffusion Switching Mechanism design principle by Taizo Shirai and Kyoji Shibutani [SS04] (cf. Subsection 1.8.4 on page 53). These define linear transformations of 4-dimensional vector fields over $\mathbb{F}_{2^8}$ and they have been chosen so that the branch numbers $\mathscr{B}_{M_0} = \mathscr{B}_{M_1} = 5$, i.e. they are maximal, and also the composed matrices $M_0 \| M_1$ and ${}^tM_0^{-1} \| {}^tM_1^{-1}$ have branch number 5. The matrices themselves have been obtained by searching circulant and Hadamard-type matrices, looking for matrices of low weight and

that can be implemented efficiently in hardware due to low XOR gate count. The choice fell then on two Hadamard-type MDS matrices, as in Khazad (Subsection 3.21.2 on page 188) and Anubis (Subsection 3.21.3 on page 190).

The key schedule of CLEFIA is also a Type 2 Feistel network but this time with *eight* equally wide branches. The same two F-functions are used – each twice per key schedule round – and instead of round keys predefined constants are used. These constants are derived from the binary expansions of $e-2$ and $\pi-3$, in order to maximize diffusion according to the Diffusion Switching Mechanism technique. 32-bit whitening keys wk_i and round keys rk_i are generated. The round keys are considered as arrays of four 8-bit values in the F-functions. In other words, the design is more conventional than it appears from the specifications, as the round key is just used to mask a branch before feeding the value to a single round SP network – the output of which is then XORed to another branch.

3.28.1 Cryptanalysis

Shortly after the presentation of the cipher, Yukiyasu Tsunoo et al. [TTS+08] show the existence of 9-round impossible differentials for CLEFIA and use it to attack some reduced round versions of the cipher. 12 rounds of CLEFIA-128, 13 rounds of CLEFIA-192, and 14 rounds of CLEFIA-256 are attacked with the time complexities 2^{119}, 2^{146}, and 2^{212}, respectively.

An improbable differential attack breaks 13 rounds of CLEFIA-128 with a time complexity of $2^{126.83}$ [Tez10a, Tez10b]. Similar attacks apply to 14 and 15 round reduced versions of CLEFIA for the key sizes 192 and 256 bits, respectively. We note that the validity of the improbable differential cryptanalysis has been recently challenged by Celine Blondeau [Blo13].

In [LWZ11] Yanjun Li, Wenling Wu and Lei Zhang present a 9-round integral distinguisher for CLEFIA and use it on to attack 12, 13 and 14-round CLEFIA with whitening keys. 12-round CLEFIA-128/192/256 is attacked with time complexity $2^{116.7}$, 13-round CLEFIA-192/256 with time complexity $2^{180.5}$, and 14-round CLEFIA-256 with time complexity $2^{244.5}$.

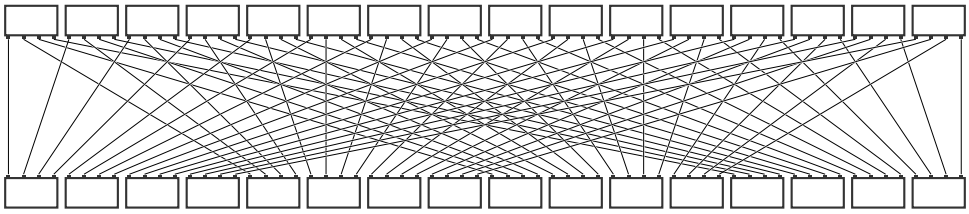
Multidimensional zero correlation linear cryptanalysis with FFT [BGW+13] breaks 14 rounds of CLEFIA-192 in time $2^{180.2}$ and 15 rounds of CLEFIA-256 in time $2^{244.2}$, essentially improving on some of the attacks in [LWZ11] by one round at roughly the same time complexity.

3.28.2 Intellectual Property

The cipher is patented, but SONY has declared that it *"is prepared to make licenses available to use CLEFIA technology under CLEFIA essential patents on fair, reasonable and non-discriminatory terms."* The patents (and patent applications) covering CLEFIA we are aware of are:

- U.S. Patent Application 2013/0016829 (priority: Japan Patent Application 2003-339364);
- U.S. Patent Application 2012/0324243 (priority: Japan Patent Application 2004-256465);
- U.S. Patent Application 2010/0061548 (priority: Japan Patent Applications 2006-206376 and -224674);
- U.S. Patent 8577023 (priority: Japan Patent Application 2006-238225);
- Japan Patent 2008-058830 (Japan Patent Application 2006-238227); and

Figure 3.33: The PRESENT pLayer
Bit (4i+j) is sent to position (16j+i) with $0 \leqslant i < 16$ and $0 \leqslant j < 4$.



- U.S. Patent 8369522 (priority: Japan Patent Application 2006-238228).

The list has been derived from the list of Japan Patent Applications given in the CRYPTREC submission package for CLEFIA.

3.29 PRESENT

PRESENT is a 64-bit block cipher developed by Andrey Bogdanov et al. and introduced at CHES 2007 [BKL+07]. There are two versions, with keys of 80 and 128 bits respectively, called PRESENT-80 and PRESENT-128. It was designed for low-cost devices like RFID-tags.

PRESENT is a SPN with 31 rounds and one final key addition. Each round consists of: a round key addition, a substitution layer, and a permutation layer.

A single 4-bit S-box is applied to all nibbles of the 64-bit state, and its action, interpreting the nibbles as hexadecimal digits, is given by the following table:

x:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$:	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

This S-box was designed with cryptanalytic resistance in mind. Conditions were posed on the S-box to improve the so-called *avalanche of change*:

1. For any output difference, there are most four inputs differences.
2. There are *no* single bit input differences that give single bit output differences.
3. Let $S_b^w(a)$ be the Walsh-Fourier coefficient of S (as defined in Subsection 2.2.8 on page 92). Then for all non-zero $a, b \in \mathbb{F}_2^4$ we require that $|S_b^w(s)| \leqslant 8$.
4. For all $a, b \in \mathbb{F}_2^4$ both with Hamming weight equal to 1 it holds that $|S_b^w(a)| = 4$.

These conditions help against linear and differential attacks by bounding from above the probabilities of linear trails and differential characteristics (in this context, property 2. is important since there is no proper diffusion layer).

The permutation layer is a pure bit permutation layer, given in Figure 3.33.

A notable aspect of this cipher is that it is one of the few contemporary SPNs with a pure bit permutation layer.

Table 3.10: Cryptanalysis of PRESENT – Published Results

Version	Rounds Attacked	Technique	Attack Complexity Time	Data	Memory	Reference
80	16/31	Differential	$2^{64.0}$	$2^{64.0}$	$2^{32.0}$	[Wan08]
80	18/31	Multiple diff.	$2^{64.0}$	$2^{64.0}$	$2^{32.0}$	[BG11b, BG11a]
80	18/31	Multiple diff.	$2^{76.0}$	$2^{39.0}$	$2^{32.0}$	[BG11b, BG11a]
80	18/31	Multiple diff.	2^{78}	2^{62}	2^{13}	[BN13]
80	24/31	Linear	$2^{40.0}$	$2^{63.5}$	$2^{40.0}$	[Ohk09]
80	24/31	Statistical sat.	$2^{57.0}$	$2^{57.0}$	$2^{32.0}$	[CS09]
80	26/31	Multiple linear	$2^{72.0}$	$2^{64.0}$	$2^{32.0}$	[Cho09, Cho10]
128	8/31	Integral	$2^{100.1}$	$2^{24.3}$	$2^{77.0}$	[ZRHD08]
128	17/31	Related keys	$2^{104.0}$	$2^{63.0}$	$2^{53.0}$	[öVTK09]
128	18/31	Multiple diff.	$2^{124.0}$	$2^{64.0}$	$2^{32.0}$	[BG11b, BG11a]
128	19/31	Algebraic diff.	$2^{113.0}$	$2^{62.0}$	n/r	[AC09]
128	19/31	Multiple diff.	2^{126}	2^{62}	2^{60}	[BN13]
128	25/31	Linear	$2^{96.7}$	$2^{64.0}$	$2^{40.0}$	[NSZW09]

Key schedule is very simple: in the case of 80-bit keys, an 80 bit register containing the secret key is rotated by 61 positions to the left, the least significant nibble is passed through the S-box, and then the value of a counter is XORed to some of the bits of the register. The round key is just the leftmost 64 bits of this register. In the case of 128-bit keys, the round key register is 128 bits wide and *two* nibbles are passed through the S-box.

3.29.1 Cryptanalysis

PRESENT has received a lot of attention by the research community but, due to its sound design, has withstood cryptanalysis well.

All attacks disclosed so far are on reduced-round versions.

Linear cryptanalysis seems to have fared better in attacking PRESENT than differential cryptanalysis, as can be seen in Table 3.10, that reports the cryptanalysis done to date.

There are no significant biclique based results to date, all the attempts to date shaving off less than one bit of complexity from brute force attack [AFL+12, JKL+12].

3.29.2 Intellectual Property

We are aware of no patents encumbering PRESENT.
The cipher has been standardised in ISO/IEC 29192 Part 2: Block ciphers.

3.30 Threefish

The tweakable block cipher Threefish was introduced in 2009 as part of the Skein hash function, a SHA-3 competition candidate [FLS+10]. Threefish is an SPN with a block size of 256, 512, or 1024 bits, the key size being always equal to the block size, and the number of rounds being 72, 72 and 80 for the three block/key sizes, respectively. The tweak input is always 128 bits.

One of Threefish's most striking features is its simplicity: all fundamental operations work on

Figure 3.34: The Threefish MIX Function

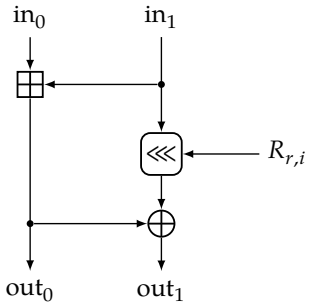
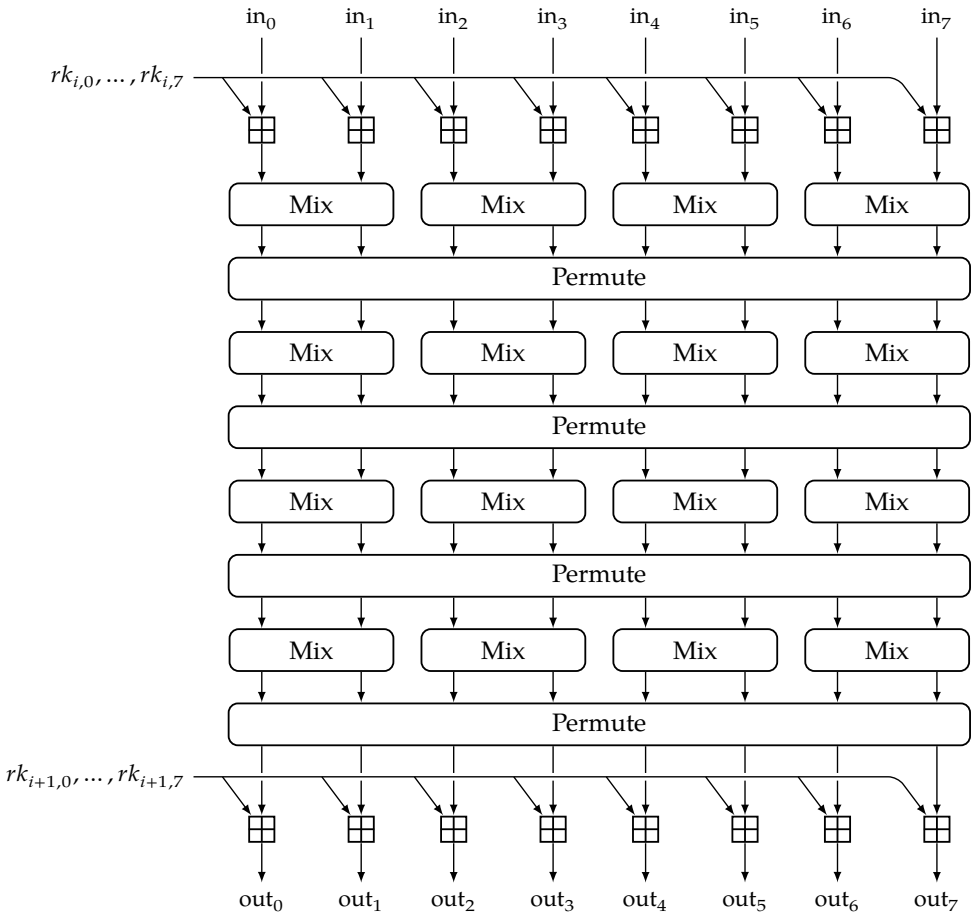


Figure 3.35: Four Rounds of Threefish-512



64-bit blocks, and non-linearity is based on a single, simple MIX function, alternated with a fixed permutation layer of 64-bit blocks.

The MIX function is represented in Figure 3.34, whereas four rounds of Threefish are depicted in Figure 3.35, in the case of Threefish-512.

The round rotation constants $R_{r,i}$ are taken from three tables (for the 256-, 512-, and 1024-bit versions). The key schedule is quite simple and involves modular additions, permutations of

the words of the round keys, and XOR with a fixed constant chosen, among randomly generated values (encrypting increasing values with AES-256 with a zero key), in order to defeat rotational attacks (see below).

Noteworthy aspects: the key mixing is done only each four rounds, an idea used later also in LED (Subsection 3.37.4 on page 227). The MIX function is extremely simple and has influenced later designs, such as SIMON (Section 3.36 on page 220). The permute operation is, literally, a permutation of the 64-bit words of the state, and is fixed.

3.30.1 Remarks

Threefish is quite fast in SW implementations: Threefish-512 encrypts data at 6.1 clock cycles per byte on a 64-bit Intel CPU.

3.30.2 Cryptanalysis

Using rotational cryptanalysis, Dmitry Khovratovich and Ivica Nikolić in 2010 presented the best cryptanalytic attacks at that time against a reduced-round Threefish cipher [KN10]. A follow-up attack from the same authors and Christian Rechberger [KNR10] breaks up to 53 of 72 rounds in Threefish-256, and 57 of 72 rounds in Threefish-512 – breaking also the corresponding collision resistance in Skein.

3.30.3 Intellectual Property

Bruce Schneier, on the Submitter Statement accompanying the submission of Skein to the SHA-3 context, wrote: *"I, Bruce Schneier, hereby declare that, to the best of my knowledge, the practice of the algorithm, reference implementation, and optimized implementations that I have submitted, known as Skein, may be covered by the following U.S. and/or foreign patents: none. I do hereby declare that I am aware of no patent applications that may cover the practice of my submitted algorithm, reference implementation or optimized implementations."*

Since Threefish is part of Skein, the same holds for Threefish.

3.31 The KATAN/KTANTAN Family

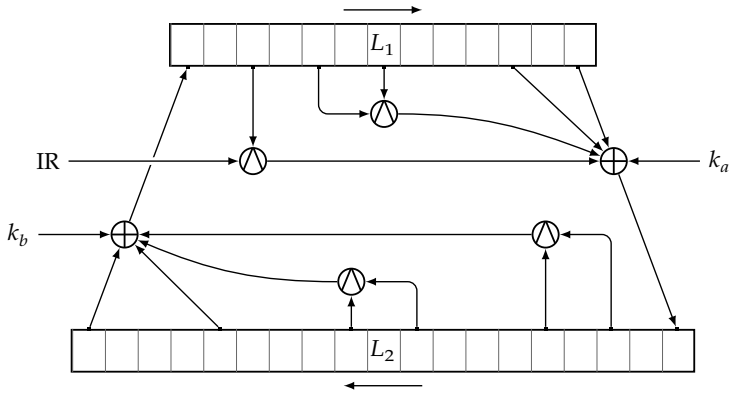
The KATAN/KTANTAN family [CDK09] contains six block ciphers divided into two *flavors*, called KATAN and KTANTAN. All block ciphers share the 80-bit key size and security level. Both flavors contain ciphers with 32, 48, or 64-bit block sizes. The ciphers in the KTANTAN flavor are more compact in hardware than their KATAN counterparts as the key is burnt into the device (and cannot be changed).

This cipher is profoundly different from most designs we have seen so far, in fact it resembles a stream cipher. In this, it follows the example of KeeLoq (Subsection 3.3.3 on page 136). The design of the KATAN/KTANTAN block cipher family is inspired by that of the stream cipher Trivium [CP05, CP08] – which in turn was designed to be analyzed with methods from the theory of block ciphers.

Figure 3.36 shows the structure of KATAN/KTANTAN.

The plaintext is first copied into the two registers L_1 and L_2, which have coprime lengths that add up to the block length (for instance, for KATAN-32 the lengths are 13 and 19 bits).

Figure 3.36: The Structure of KATAN/KTANTAN



Then at each clock the two registers function as two "interleaved" NLFSRs as described in the figure, where the feedback bits are fixed parameters of the system. A round consists in one clock of this circuit. At each round two key schedule bits are mixed in as depicted.

In the Figure *IR* stands for Irregular update Rule: at irregular intervals the bit *IR* determined whether a bit of L_1 is actually mixed in or not. *IR* is actually the most significant bit in a small 8-bit LFSR defined by polynomial $x^8 + x^7 + x^5 + x^3 + 1$ that begins with all ones and after 254 cycles will reach a predetermined state, at that point the cipher will stop. The use of *IR* will determine de facto an irregular sequence of two different update functions, and LFSR properties will guarantee that no function will be used more than seven times in a row, limiting attacks.

The two key bits k_a and k_b are derived from an 80-bit LFSR initialized with the secret key and clocked twice each round.

At the end of the 254 rounds the content of the registers L_1 and L_2 is the ciphertext.

Careful choice of the parameters makes the cipher easily invertible reusing the same hardware.

The cipher is extremely simple and is designed for low power low gate count applications.

Throughput relatively low but it is easy to increase it considerably at a relatively small price.

KTANTAN uses less resources since the secret key is hardwired in the circuit.

The designers of KATAN/KTANTAN did not patent any aspect of the cipher family.

3.31.1 Cryptanalysis

Simon Knellwolf, Willi Meier and María Naya-Plasencia [KMNP11] apply their technique of Conditional Differential Cryptanalysis [KMNP10] to mount attacks on KATAN in a related-key scenario and obtain practical key-recovery attacks for 120, 103 and 90 of 254 rounds of KATAN-32, KATAN-48 and KATAN-64, respectively. Similar results hold for KTANTAN.

Andrey Bogdanov and Christian Rechberger in [BR10] exploit some weaknesses in the key schedule of KTANTAN, i.e. some irregularities that cause some rounds to be influenced by some key bits and not by other bits, and mount a MITM attack with overlapping key subsets. They succeed to break full KTANTAN-32, resp. KTANTAN-48 using only three plaintext/ciphertext pairs and $2^{75.170}$, resp. $2^{75.044}$ encryptions, as well as full KTANTAN-64 using

2 plaintext/ciphertext pairs and $2^{75.584}$ encryptions.

3.31.2 Advantages

Very simple design. Easy to implement in HW with extremely reduced gate count. Uses very little power.

3.31.3 Disadvantages

Slow performance. Too many rounds, hence very hard to reduce latency. Key length and block size are small.

3.31.4 Intellectual Property

We are aware of no patents on KATAN/KTANTAN.

3.32 PRINTcipher

Lars Knudsen, Gregor Leander, Axel Poschmann, and Matthew Robshaw introduced PRINTcipher in 2010 [KLPR10] to cater to the needs of adding cryptography to printed IC (integrated circuits) and exploit the opportunities offered by this medium.

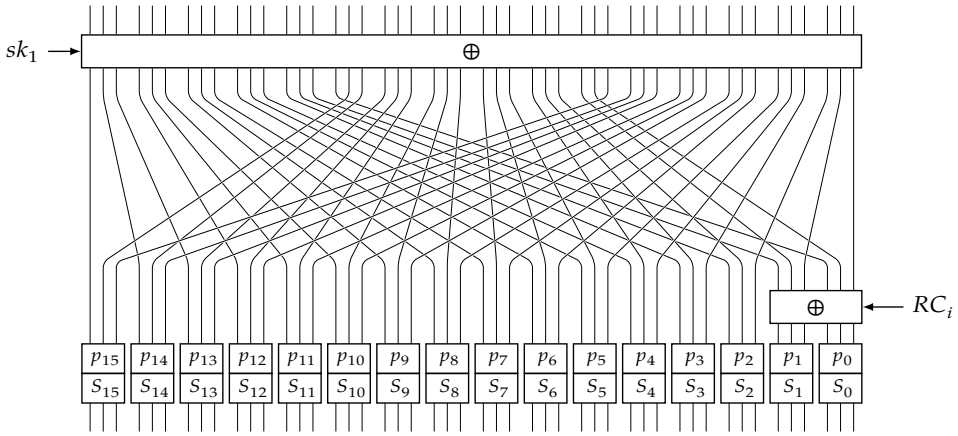
On one hand an IC printed onto a variety of mediums using silicon ink must meet heavy size constraints, whence the cryptographic components must use an extremely low gate count. On the other hand for most applications of this technology (such as RFID tags) the key can be fixed (and therefore it becomes an "identity" of the circuit) and therefore one can dispense with the traditional key schedule: IC printing makes hardwiring a different key onto a circuit very easy, whereas conventional IC manufacturing creates identical circuits and, whereas a specific tag can be personalised with a unique key, this represents an additional post-fabrication step (PUFs, i.e. physically unclonable functions, are an alternative, but not without their own drawbacks).

The PRINTcipher family was designed for this context. It is a $b = 48$ or 96-bit block cipher. The effective key size is $(5/3)b$. It is a SPN with $r = b$ rounds. A round (in the case of $b = 48$) is represented in Figure 3.37 on the next page. A fixed b-bit key is given "numerically" (but it is still hardwired in the system), whereas $(2/3)b$ further bits of algorithm variability are encoded in the circuit itself.

A round of encryption works as followed:

1. First, the state is xored with a round key sk_1.
2. Then, the cipher state is shuffled using a fixed linear diffusion layer. This layer sends bit i to bit $3i \bmod (b-1)$ for $i < b-1$ and bit $b-1$ to $b-1$.
3. The least significant 6 or 8 (for $b = 48, 96$ respectively) bits of the cipher state are xored with a round constant that is generated using an LFSR.
4. The state is transformed using a layer of key dependent permutations and an array of $b/3$ identical 3-bit S-boxes:
   (a) The input to each S-box is first permuted in a key-dependent permutation layer: Four possible permutations are defined, providing each the equivalent of two additional bits

Figure 3.37: A Round of PRINTcipher



of key material for each three bits of the state. For each S-box the same permutation – once chosen – is used in the same position in every round. The four permutations, for the choices (00), (01), (10) and (11) of key bits, send a bit triple $(c_2\, c_1\, c_0)$ to $(c_2\, c_1\, c_0)$, $(c_1\, c_2\, c_0)$, $(c_2\, c_0\, c_1)$, and $(c_0\, c_1\, c_2)$, respectively.

(b) Each bit triple is then transformed non linearly using the same S-box.

The $(5/3)b$-bit key k is considered as consisting of two sub keys sk_1 and sk_2 where sk_1 is b bits long and sk_2 is $(2/3)b$ bits long. The first subkey is used, unchanged, in the xor layer of each and every round, and each of the $b/3$ pairs of bits of the second sub key selects the permutation applies to the corresponding S-box.

The S-box is a 3-bit S-box, chosen because it can be realised with only 11 GE, whereas 4-bit S-boxes require on average 28 GE. This makes the whole substitution layer 50% less expensive, but also weaker – this fact is compensated by the high number of rounds.

x:	0	1	2	3	4	5	6	7
$S[x]$:	0	1	3	6	7	4	5	2

The combination of key dependent permutation and S-box results the facto in a keyed S-box, where, however, 0 is always mapped to 0 and 7 to 2. For each key bit pair the resulting combination has always exactly two fixed points.

The 3-bit S-box has been chosen to be optimal with respect to linear and differential properties, and to minimise the number of single-bit to single-bit differences (that is, there is exactly one). An interesting property (which could have crypto analytic implications) is the following: For any 3-to-3-bit permutation P, there exist constants c and d (depending on P) such that

$$S(P(x)) = P(S(x \oplus c)) \oplus d \qquad \text{for all } x \ .$$

3.32.1 Cryptanalysis

In [ALZ11] two differential attacks are mounted against 22 rounds of PRINTcipher-48, requiring the full code book and about 2^{48} computational steps. One attack works by first determining the (fixed) keyed permutations.

In [LAAZ11] the existence of invariant subspaces is proven for moderately sized classes of weak keys: 252 out of 2^{80} for PRINTcipher-48 and 2102 out of 2^{160} for PRINTcipher-96. The existence of invariant subspaces stems from the fact that there exist subsets of all S-boxes that are mapped onto themselves for specific sets of inputs and carefully chosen keys. The search for these subspaces is helped by the fact that the key mixing is the same in each round. The key remark is that in PRINTcipher the probability of truncated differential characteristics, the bias for statistical saturation attacks, and the bias of linear hulls are extremely key-dependent and that for a weak key, increasing the number of rounds up to the full number of rounds does not increase the security of the cipher with respect to these attacks. For example, when weak keys are used there is at least one linear approximation for PRINTcipher-48 with bias at least 2^{-17}, and truncated differential characteristics with probability 2^{-16}.

The approach in [LAAZ11] has been further investigated=, finding 64 families of weak keys for PRINTcipher-48 and as many as 115,669 for PRINTcipher-96, which have been found using Mixed Linear Integer Programming. At least 2^{45} weak keys for PRINTcipher-48 can be recovered in less than 20 minutes per key on a single PC using only a few chosen and one known plaintext(s).

Martin Ågren and Thomas Johansson [ÅJ11a, ÅJ11b] also investigate the strength of the cipher under weak keys. They find even more families for reduced round variants. For instance, 28 rounds can be broken for half of the keys, and 29 rounds for a fraction 2^{-5} of the keys.

3.32.2 Remarks

The combination of a key dependent bit permutation and a fixed S-box is equivalent to the choice among four S-boxes depending on key bits, a design strategy that goes back to the early history of block cipher design, cf. Lucifer (Section 3.1 on page 126).

3-bit S-boxes have been used previously in other ciphers, such as 3-way [DGV93] and the Scaleable Encryption Algorithm [SPGQ06] (see Section 3.26 on page 197) while key-dependent algorithm features have appeared in a variety of block ciphers including Blowfish (Section 3.9 on page 155), Twofish (Section 3.13 on page 162), and GOST (Section 3.4 on page 138).

The large classes of weak keys in PRINTcipher showcases the disadvantages of choosing key dependent permutations, which can partially undo some of the advantages of the chosen fixed permutation layers. This is exacerbated by the fact that the key schedule is trivial: The same key bit is always mixed at the same place, and in fact hardwired: in correspondence of a one in the key there is a NOT gate in the circuit. This is a consequence of the design rationale: the round must be iterated, and the round constant mixing is minimal.

On the other hand, the large number of rounds makes the mixing of a round constant only to the last six or eight significant bits of the state a wise choice: they are completely diffused after just two or three rounds (i.e. the round constant makes, in the case of PRINTcipher-48, two S-boxes active, six in the next round, and 16 in the round after that – a similar computation shows that all 32 S-boxes are active after three rounds in PRINTcipher-96).

3.33 Bel-T

Bel-T is a Bielorussian block cipher. The actual name is СТБ 34.101.31-2011. СТБ is an acronym for (Государственный) Стандарт, (Республики) Беларусь, i.e. *(State) Standard, (the Republic of) Belarus*. The information reported here has been taken from the original standards document in Russian, which is available from http://apmi.bsu.by/assets/files/std/belt-spec27.pdf. The current version was standardised in 2011.

The cipher has a block size of 128 bits and a key size of 128, 192, or 256 bits. Keys shorter than 256 bits are just padded to 256 bits.

It is an 8-round cipher, where the round structure is inspired by Feistel networks and the Lai-Massey design. A round is depicted in Figure 3.38 on the next page.

Bel-T is a four-branch structure, with 32-bit branches.

In each round the four branches – a, b, c and d – are considered as two pairs – the first consisting of a and b, the second comprising c and d – each of which is mixed in a three-round Feistel network. However, the two parallel Feistel network are mixed in the middle (after two rounds of the network of a and b, and after the first round of the network of c and d), by applying a Lai-Massey round to branches b and c. The Lai-Massey structure is additive (we mentioned this possibility in Section 1.5 on page 36, cf. Figure 1.6).

Nonlinearity is given by the use of incompatible algebraic operations (arithmetic modulo 2^{32} and bitwise XOR) and by three different 32-bit to 32-bit functions G_i (with $i = 5, 13, 21$). The functions G_i consists of parallel applications of a single 8-bit S-Box to all four bytes of the input followed by a cyclic rotation of the whole 32 bit word to the left by i bits – the rotation amounts are the largest Fibonacci numbers smaller than 32, skipping the 8 for obvious reasons.

The F-functions of both the Feistel components and of the Lai-Massey component consists in key mixing (addition modulo 32) and application of one of the functions G_i – in other words they are just half-size GOST-like F-functions. The output of the F-function in the Lai-Massey component is then further "tweaked" by the XOR of the round constant round number (from 1 to 8) in its binary representation, padded with zeros.

The whole round is terminated by a Nyberg-like branch permutation.

The key K is just split into eight 32-bit words $\theta_1, \ldots, \theta_8$ and the 56 round keys $k_1, \ldots, k_{56}$ just repeat the sequence $\theta_1, \ldots, \theta_8$ seven times. These 56 values are then used for the rounds $i = 1, 2, \ldots, 8$ as indicated in Figure 3.38 on the next page. This key schedule is reminiscent of that of NewDES (Subsection 3.3.2 on page 134).

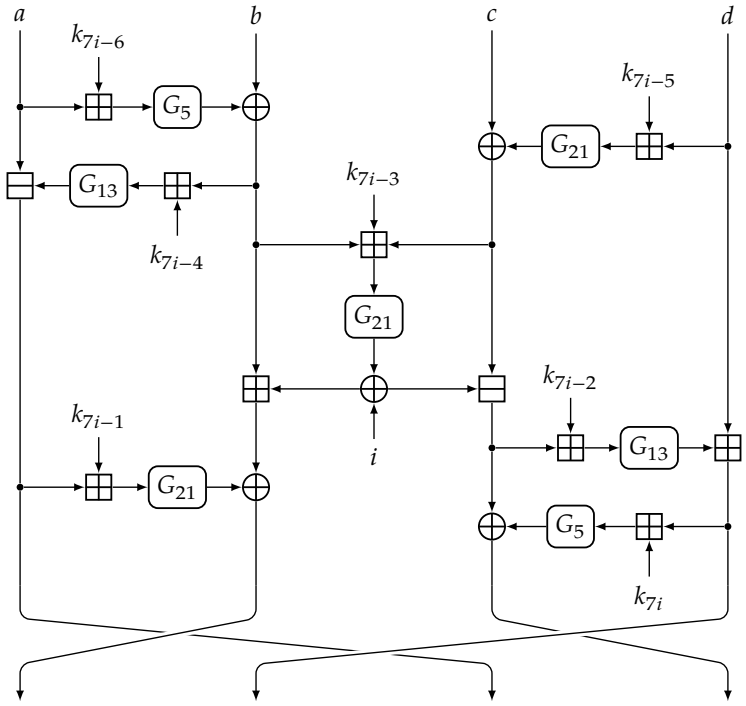
3.33.1 Remarks

There is no publicly disclosed cryptanalysis. We performed some tests on the Bel-T S-box and did not found significant biases, but it seems to be less optimal than the Rijndael S-box. The design rationale was not disclosed.

We do not know whether aspects of the cipher are patented.

This cipher is just one of many current efforts ongoing in the republics of the former Soviet Union to develop alternatives to western ciphers. We mention them briefly since they are just variations on the AES theme:

Figure 3.38: A Round of Bel-T



- Russia is developing a cipher called Кузнечик (Kuznechik, i.e. "Grasshopper" in Russian) which should be a 10-round AES-like SPN that uses a 16×16 MDS matrix over $\mathbb{F}_{2^8}$ in place of the 4×4 MDS matrix used in Rijndael. The Kuznechik key schedule is a Feistel network with constants as round keys, each round giving a round key for the main cipher.

- Ukraine is developing Калина (Kalyna, the ukrainian name of the national flower of Ukraine, i.e. the Guelder rose), an AES-like cipher with key and block sizes of 128, 256 and 512 bits, It uses 8×8 MDS matrices in the linear diffusion steps, and employs four non-CCZ-equivalent 8-bit S-boxes in place of a single one. The key schedule is itself an AES-like SPN. Kalyna is optimised for 64-bit processors and uses 64-bit modular addition for key whitening steps.

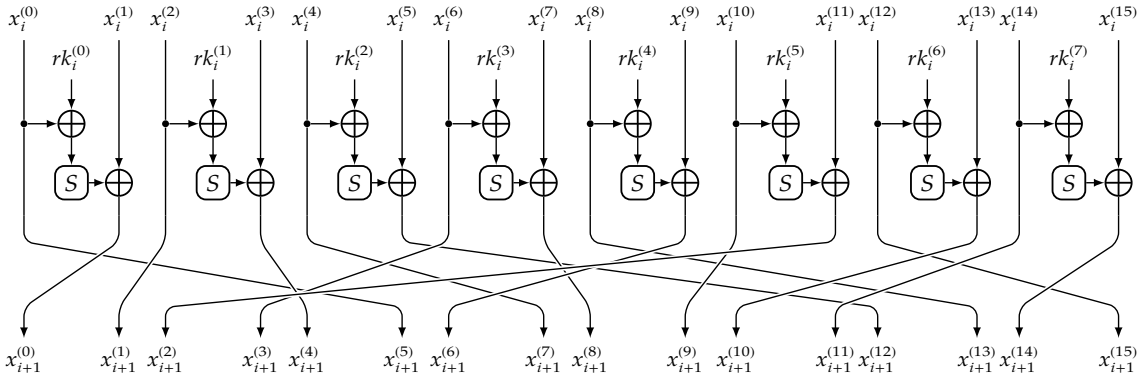
3.34 TWINE

TWINE was developed by NEC and presented in 2012 [SMMK12]. Its designers are Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. It is an interesting design building on previous theoretical work by Tomoyasu Suzaki and Kazuhiko Minematsu on improvements to multi-branch Feistel networks to reduce the number of rounds necessary to achieve full diffusion [SM10].

TWINE is a 64-bit block cipher with 80 and 128-bit key lenghts. It was designed for small hardware implementation and efficient embedded software realization.

It is a generalized 16-branch Type 2 Feistel-like network with a complex branch permutation in place of a simple rotation. Each branch is just 4 bits wide. A round is represented in Figure 3.39. The number of rounds is 36.

Figure 3.39: A Round of TWINE



The design is very interesting for a few reasons. The F-function is minimal: it consists of key mixing and a single S-box, this simplicity being compensated by the large number of branches. But, most importantly, it shows how the distinction between SPN and Feistel network is in fact blurred: if we group the branches two by two as a sequence of 8 bytes, the XORing of the keyed transform (F-function) of a nibble to the right can be seen as the combination of key mixing with an substitution layer on the bytes – "incomplete" in the sense that four of the eight bytes are left unchanged – followed by a bit (in fact, nibble) permutation layer. This interpretation could have been made for previous Type 2 Feistel designs as well, but TWINE makes it very clear.

The S-box was chosen to satisfy following requirements:

1. The maximum differential and linear probabilities are 2^{-2}, which is the theoretical minimum for an invertible S-box;
2. The algebraic degree is 3; and
3. The interpolation polynomial is as dense as possible and has degree 14.

The design of the S-box is inspired by the AES S-box, i.e. it is defined as $y = S(x) = f((x \oplus b)^{-1})$ where $b = 1$ is a 4-bit constant, the inversion is over the field $\mathbb{F}_{2^4} = \mathbb{F}[z]/(z^4 + z + 1)$ and f is an affine transformation. The resulting S-box is:

x:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$:	C	0	F	A	2	B	9	5	8	3	D	7	1	E	6	4

Since the deepest element in the cipher is the 4-Bit S-box and it is a Feistel cipher, the critical path traverses 18 S-boxes and key-mixing operations and we estimate that a critical path of less than 100 GE is achievable.

The design of the branch shuffle used in TWINE is explained in Subsection 1.8.3 on page 52.

3.34.1 Cryptanalysis

The cipher is very recent, so there is not much in the direction of cryptanalysis yet. The paper that presented TWINE contains considerable cryptanalitic arguments to justify the design.

There is some biclique cryptanalysis, shaving only fractions of a bit from the cipher's security [ÇKöB12, KDH13].

3.34.2 Intellectual Property

Japanese Patent Application PCT/JP2012/006314 covers aspects of TWINE, such as the branch permutation.

3.34.3 LBlock

The cipher LBlock [WZ11] despite superficial differences, is in fact quite similar to TWINE. LBlock is presented as a classic Feistel design, i.e. the 64-bit state is partitioned into two 32-bit halves. An array of different S-boxes operates nibble-wise after key mixing, then the 4-bit outputs are permuted before being XORed. If the structure is "unrolled" the "shuffle" in LBlock encryption is the same as the "shuffle" used in TWINE decryption. Hence, there are enough similarities that the two ciphers are often analyzed in the same paper, as the above cited cryptanalysis exemplifies.

3.35 PRINCE

Introduced in 2012 [BCG+12a], PRINCE is a 64-bit block cipher with a 128-bit key. It is an iterative cipher with heterogeneous rounds, designed with following goals in mind for a HW implementation:

1. The cipher can perform instantaneous encryption and decryption, within a single clock cycle. There is no warm-up phase. In particular there should be negligible key setup overheads for both encryption and decryption. This was achieved by dispensing with key expansion altogether.

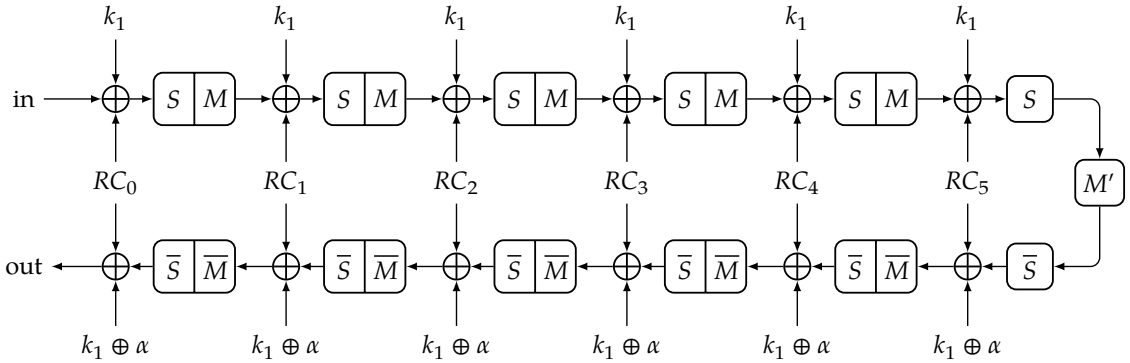
2. If implemented in modern chip technology, low delays resulting in moderately high clock rates can be achieved.

3. The hardware costs are moderate (i.e., considerably lower than fully unrolled versions of AES or PRESENT).

PRINCE has a completely symmetric structure: the encryption circuit is used unchanged for decryption. It has a property its designers call α-reflection, i.e. decryption with key k is performed by encrypting with key $k \oplus \alpha$ for a constant α that is hardwired in the design of the key schedule. This property prevents the cipher from being an involution.

This is achieved by using the idea of a *reflector*, introduced by Arthur Scherbius, the inventor of the ENIGMA machine: Encryption uses both a keyed function f_K and its inverse f_K^{-1}, and an involutory operation R called a *reflector*. Then $E_K = f_K^{-1} \circ R \circ f_K$. However, this makes the cipher an involution, and using results of Youssef, Tavares, and Heys at SAC 96 [YTH96] proved

Figure 3.40: The Structure of $\text{PRINCE}_{\text{core}}$



bounds on the number of fixed points of these operations, and in fact DES has weak keys that make the cipher involutory.

The concept is therefore generalized in PRINCE by masking the key after the reflection point: $E_K = f_{K+\alpha}^{-1} \circ R \circ f_K$. Such a cipher is called a **reflection cipher**, i.e. a cipher such that there is a permutation of the key space P with the property that $E_K^{-1} = E_{P(K)}$ (this definition is very general, for instance RSA with P = inversion modulo $(p-1)(q-1)$ is a reflection cipher).

Key whitening is used to further strengthen the cipher. The key whitening is non-symmetric in order to prevent variants of slide attacks [DKS12]: The 128-bit key k is split into two 64-bit halves k_0 and k_1. The key k_0 together with the derived value $k_0' = (k_0 \ggg 1) \oplus (k_0 \gg 63)$ are used for key whitening, whereas k_1 is used for a 12 round iterative core cipher, called $\text{PRINCE}_{\text{core}}$.

Figure 3.40 shows the structure of $\text{PRINCE}_{\text{core}}$. The first six rounds of $\text{PRINCE}_{\text{core}}$ consist of a key addition, the addition of a round constant, an S-box layer S, and a linear layer M. The last six rounds are the inverse of the first six rounds, and thus are composed of the inverse operations that form the initial rounds, that is: a linear layer $\overline{M}$ that inverts M, the inverse S-Box layer $\overline{S}$, and a key addition with the addition of a round constant. Between the first and the last six rounds there is an invertible linear transformation M': the final liner transformation of the sixth round and the initial liner transformation of the seventh round are subsumed into it. The middle transform M' is the reflector in the $\text{PRINCE}_{\text{core}}$ design. Note that $RC_{11-i} = RC_i \oplus \alpha$ and therefore $E_K^{-1} = E_{K \oplus \alpha}$.

3.35.1 The S-box

The S-box is defined as follows:

x:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$:	B	F	3	2	A	C	9	1	6	7	8	0	E	5	D	4

It was chosen in order to satisfy following requirements.

1. The maximal probability of a differential is 1/4
2. There are exactly 15 differentials with probability 1/4.

3. The maximal absolute bias of a linear approximation is $1/4$.
4. There are exactly 30 linear approximations with absolute bias $1/4$.
5. Each of the 15 non-zero component functions has algebraic degree three.

Up to affine equivalence there are only eight S-boxes fulfilling those criteria [LP07]. Since this S-box is not involutory, its inverse must be implemented as well.

3.35.2 The Linear Transformations

The linear transformations M and M' are defined by 64×64 matrices over $\mathbb{F}_2$ which have been designed with the aim to have the lowest possible density under the condition that at least 16 S-boxes are active in 4 rounds by a superbox argument. Trivially, this implies that each output bit of an S-box has to influence three S-boxes in the next round and therefore the minimum number of ones per row and column is three. If this can be achieved, HW implementation will be as inexpensive as possible.

The M'-layer is only used in the middle round, thus it is required to be an involution to ensure the α-reflection property. The designers then first defined M' and then put $M = \pi \circ M'$ where π is a permutation of the 16 nibbles of the state. Defining M in terms of M' further simplifies both HW and SW implementations.

3.35.2.1 The Nibble Permutation

The permutation π simply maps the i-th nibble to the $(5\,i \bmod 16)$-th position – and this is just wiring in hardware. This operation is in fact a `ShiftRows`: Indeed, if we put the 16 nibbles of the state in a square matrix, then π acts as follows on the state

15	11	7	3
14	10	6	2
13	9	5	1
12	8	4	0

$\mapsto$

3	15	11	7
6	2	14	10
9	5	1	13
12	8	4	0

,

where nibbles number 0 and 15 are the least and the most significant ones, respectively.

3.35.2.2 The Diffusion Matrix

The matrix M' is defined as follows. We first define 4×4 matrices

$$M_0 = \begin{pmatrix} 0&0&0&0\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&0&1 \end{pmatrix}, \quad M_1 = \begin{pmatrix} 1&0&0&0\\ 0&0&0&0\\ 0&0&1&0\\ 0&0&0&1 \end{pmatrix}, \quad M_2 = \begin{pmatrix} 1&0&0&0\\ 0&1&0&0\\ 0&0&0&0\\ 0&0&0&1 \end{pmatrix}, \quad M_3 = \begin{pmatrix} 1&0&0&0\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&0&0 \end{pmatrix}.$$

Then $\hat{M}^{(0)}$ is defined as the 16×16 anti-circulant block matrix whose first row of blocks is defined by (M_0, M_1, M_2, M_3), and $\hat{M}^{(1)}$ as the 16×16 anti-circulant block matrix whose first row

of blocks is defined by (M_1, M_2, M_3, M_0), explicitly:

$$\hat{M}^{(0)} = \begin{pmatrix} M_0 & M_1 & M_2 & M_3 \\ M_1 & M_2 & M_3 & M_0 \\ M_2 & M_3 & M_0 & M_1 \\ M_3 & M_0 & M_1 & M_2 \end{pmatrix}, \quad \hat{M}^{(1)} = \begin{pmatrix} M_1 & M_2 & M_3 & M_0 \\ M_2 & M_3 & M_0 & M_1 \\ M_3 & M_0 & M_1 & M_2 \\ M_0 & M_1 & M_2 & M_3 \end{pmatrix}.$$

Finally, M' is the 64×64 block diagonal matrix with $(\hat{M}^{(0)}, \hat{M}^{(1)}, \hat{M}^{(1)}, \hat{M}^{(0)})$ as diagonal block. The matrix M' is an involution with 2^{32} fixed points, which is roughly the expected average for a randomly chosen involution over a set of cardinality 2^{64}.

There is an equivalent way to represent the operation M' that makes the wide trails pedigree of the cipher explicit. First, put the bits of the state into a 4×16 matrix as follows:

63	62	61	60	47	46	45	44	31	30	29	28	15	14	13	12
59	58	57	56	43	42	41	40	27	26	25	24	11	10	9	8
55	54	53	52	39	38	37	36	23	22	21	20	7	6	5	4
51	50	49	48	35	34	33	32	19	18	17	16	3	2	1	0

The rows of each 4×4 square of bits are the nibbles of the state. The actual building block for M' is the following map on four bits of a column of this matrix:

$$x = \begin{pmatrix} x_3 \\ x_2 \\ x_1 \\ x_0 \end{pmatrix} \mapsto (x_0 + x_1 + x_2 + x_3) \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \oplus \mathrm{Rot}_{n_i} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

where Rot_{n_i} is a cyclic rotation by n_i places towards the top. For the 16 columns of the matrix above, read from left to right, the sequence of the n_i is: 3,2,1,0; 0,3,2,1; 0,3,2,1; 3,2,1,0. This second representation makes it clear that M' works on 4 subspaces of dimension 16 over $\mathbb{F}_2$ in parallel and that its purpose is to ensure diffusion into 16-bit superboxes. Hence, it is a `MixColumns` operation.

3.35.3 The Key Schedule

We note that there are no real round keys. At each round the same key k_1 is added, together with the round constant (so one could claim that the key schedule is just the addition different round constants to the secret key). The first round constant is zero, the constants RC_i for $i = 1, \ldots, 5$ and $i = 11$ are successive 64-bit chunks of the binary expansion of the fractional part of π, the remaining constants satisfy $RC_{11-i} = RC_i \oplus RC_{11}$.

This design and choice of the constants imply that $\mathrm{PRINCE}_{\mathrm{core}}$ possesses the α-reflection property. For any key k_1, decrypting with key k_1 is performed by encrypting with $k_1 \oplus \alpha$, where α = `c0ac29b7c97c50dd` $= RC_{11}$. Decryption reuses the same circuit for encryption. For the complete cipher PRINCE, the key whitening must be performed in reverse.

3.35.4 Cryptanalysis

Table 3.11 lists several publicly available cryptanalytic results pertaining to PRINCE. Note that the chosen α and related key attack techniques do not immediately apply to most applications. Also attacks limited to $\text{PRINCE}_{\text{core}}$ are not immediately applicable because practical applications will not use the core cipher in isolation.

Table 3.11: Cryptanalysis of PRINCE – Selected Published Results

Cipher	Rounds Attacked	Attack Complexity Time	Data	Memory	Technique	Reference
PRINCE	4/12	2^{64}	2^{4}	2^{4}	Integral	[JNP+13]
PRINCE	4/12	$2^{43.4}$	32 KP	$2^{26.7}$	Differential/Logic	[DP15]
PRINCE	4/12	5 sec.	2^{10}	$\lll 2^{27}$	MitM	[DP15]
PRINCE	4/12	$2^{23.9}$	48	48	Integral	[Mor17]
PRINCE	4/12	$2^{7.4}$	64	negl.	Integral	[RR16c]
PRINCE	5/12	$2^{21.4}$	32	32	Integral	[RR16c]
PRINCE	5/12	2^{13}	2^{13}	32	Integral	[RR16c]
PRINCE	6/12	2^{64}	2^{16}	2^{16}	Integral	[JNP+13]
PRINCE	6/12	$2^{32.9}$	$2^{14.9}$	$\lll 2^{27}$	Differential/Logic	[DP15]
PRINCE	6/12	$2^{86}+2^{86}$ MA	2 KP	$2^{24.6}$	Acc. Exh.	[RR16a]
PRINCE	6/12	$2^{24.6}$	2^{13}	2^{13}	Integral	[RR16c]
PRINCE	7/12	$2^{52.1}$	$2^{34.6}$	$2^{34.6}$	Higher Order Diff.	[Mor17]
PRINCE	7/12	$2^{44.3}$	2^{33}	2^{33}	Higher Order Diff.	[RR16c]
PRINCE	8/12	2^{124}	1 or 2	2^{20}	MitM	[CNPV13a, CNPV13b]
PRINCE	8/12	2^{60}	2^{53}	2^{30}	MitM	[LJW13]
PRINCE	8/12	$2^{50.7}$ *	2^{16}	$2^{84.9}$	MitM	[DP15]
PRINCE	8/12	$2^{65.7}$ *	2^{16}	$2^{68.9}$	MitM	[DP15]
PRINCE	9/12	$2^{51.21}$	$2^{46.89}$	$2^{52.21}$	Mult. diff.	[CFG+14]
PRINCE	10/12	2^{68}	2^{57}	2^{41}	MitM	[LJW13]
PRINCE	10/12	$2^{60.62}$	$2^{57.94}$	$2^{61.52}$	Mult. diff.	[CFG+14]
PRINCE	Full	2^{125}	2	negl.	Ad hoc, Single key	[JNP+13]
PRINCE	Full	2^{64}	2^{33}	2^{33}	Related key	[JNP+13]
PRINCE	8/12	$2^{122.7}$	2 KP	negl.	Optimised Search	[RR16b]
PRINCE	8/12	$2^{109.3}$	2 KP	2^{65}	MitM	[RR16b]
PRINCE	10/12	$2^{124.1}$	2 KP	negl.	Optimised Search	[RR16b]
PRINCE	10/12	$2^{122.2}$	2 KP	$2^{53.3}$	MitM	[RR16b]
PRINCE	Full	$2^{125.5}$	2 KP	negl.	Exhaustive Search	[JNP+13]
PRINCE	Full	$2^{125.1}$	2 KP	negl.	Optimised Search	[RR16b]
$\text{PRINCE}_{\text{core}}$	2/12	2^{32}	2^{32}	2^{32}	Differential	[ALL12]
$\text{PRINCE}_{\text{core}}$	4/12	2^{56}	2^{48}	2^{48}	Differential	[ALL12]
$\text{PRINCE}_{\text{core}}$	4/12	2^{8}	2^{4}	2^{4}	Integral	[JNP+13]
$\text{PRINCE}_{\text{core}}$	5/12	2^{21}	$5 \cdot 2^{4}$	2^{8}	Integral	[JNP+13]
$\text{PRINCE}_{\text{core}}$	6/12	2^{30}	2^{16}	2^{16}	Integral	[JNP+13]
$\text{PRINCE}_{\text{core}}$	8/12	2^{53}	2^{53}	2^{28}	MitM	[LJW13]
$\text{PRINCE}_{\text{core}}$	Full	2^{63}	2^{40}	2^{8}	Biclique	[ALL12]
$\text{PRINCE}_{\text{core}}$	Full	2^{39}	2^{39}	2^{39}	Related key bmrg.	[JNP+13]
$\text{PRINCE}_{\text{core}}$	Full	2^{41}	2^{41}	negl.	Single key bmrg. (chosen α)	[JNP+13]
Modified S-box	10/12	$2^{53.61}$	$2^{50.42}$	$2^{54.00}$	Mult. diff.	[CFG+14]
Modified S-box	11/12	$2^{62.43}$	$2^{59.81}$	$2^{63.39}$	Mult. diff.	[CFG+14]

*: Online time, negl.: negligible, Acc. Exh.: Accelerated Exhaustive Search

However, there are several attacks that have come quite close to significantly reducing the security of the full cipher. This means that the security margin is perhaps too small.

3.36 SIMON and SPECK

On February 12th, 2013, a team of cryptologists with the NSA released performance data for two new families of block ciphers, called SIMON and SPECK. This was followed on June 19th, 2013 by the release of the specifications [BSS+13], and on August 22, 2013 Ray Beaulieu, one of the designers, gave a public presentation at the CHES 2013 Rump Session in Santa Barbara.

Both ciphers are designed to perform well in lightweight applications, but SIMON is tuned for optimal performance in hardware, and SPECK for optimal performance in software, especially on microcontrollers. Here, lightweight does not imply low latency.

For values $n = 16, 24, 32, 48$ and 64 of a parameter called *word size*, the block size is always $2n$ and the key size can be mn where $m = 2, 3$ or 4 - but not all block size/key size combinations are possible: the ten admissible combinations are given in Table 3.12.

Table 3.12: Some Parameters for SIMON and SPECK

				SIMON		SPECK	
Block Size	Key Size(s)	n	m	Rounds	Sequences	Rounds	(α, β)
32	64	16	4	32	z_0	22	$(7, 2)$
48	72, 96	24	3, 4	36, 36	z_0, z_1	22, 23	$(8, 3)$
64	96, 128	32	3, 4	42, 44	z_2, z_3	26, 27	$(8, 3)$
96	96, 144	48	2, 3	52, 54	z_2, z_3	28, 29	$(8, 3)$
128	128, 192, 256	64	2, 3, 4	68, 69, 72	z_2, z_3, z_4	32, 33, 34	$(8, 3)$

The notation SIMON-xx/yy, resp. SPECK-xx/yy, denotes SIMON, resp. SPECK, with xx-bit blocks and yy-bit keys.

At the moment of this writing there is no cryptanalysis of these two designs. They can both be defined as Feistel networks with nonlinearity provided by the use of algebraic diverse structures and not by the use of substitution layers.

One interesting aspect of the paper [BSS+13] is that it contains gate counts for several implementations of each cipher and each allowed key size/block size combination, for several possible gate count/performance tradeoffs. As many as 10 different implementations are reported, for instance for Simon-96/96, going from a 1160 GE implementation requiring 2743 cycles to process one block to a 1790 GE implementation requiring just 57 cycles. The fact that a 50% increase in gate count leads to a nearly 50-fold increase in throughput is one of the most extreme documented examples in the literature.

The designers of SIMON and SPECK declare in [BSS+13] *"the algorithms presented are free from any intellectual property restrictions. This release does not constitute an endorsement of these algorithms for official use."*

Figure 3.41: A Round of SIMON

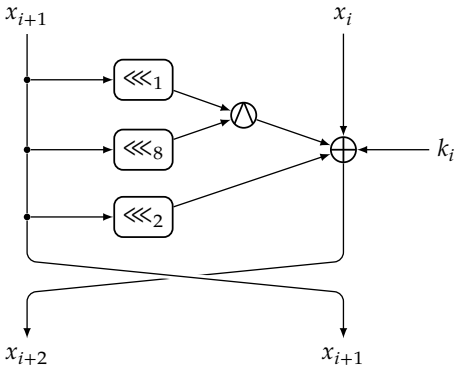


Figure 3.42: The SIMON Key Schedule for $m = 2$

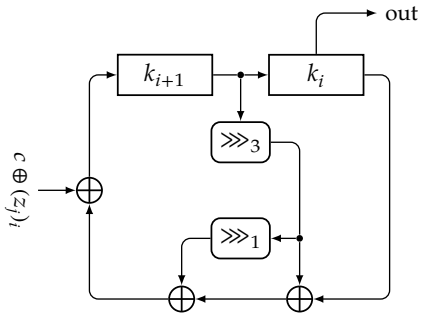


Figure 3.43: The SIMON Key Schedule for $m = 3$

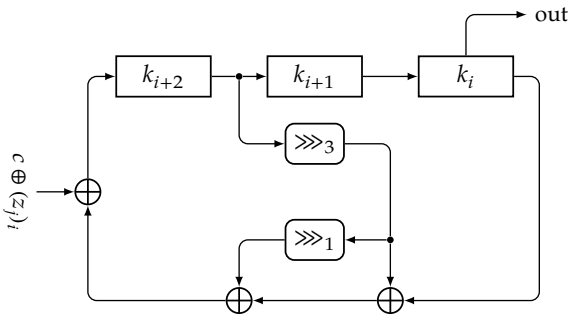
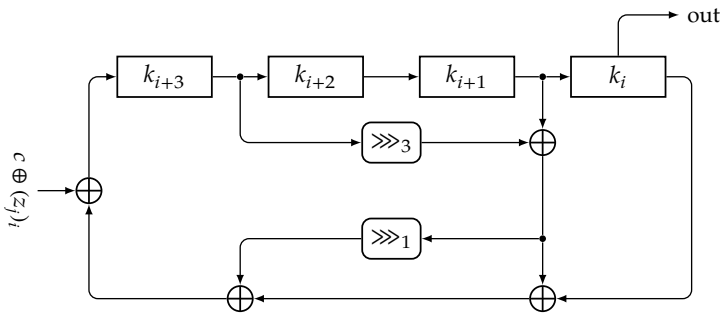


Figure 3.44: The SIMON Key Schedule for $m = 4$



3.36.1 SIMON

SIMON is a classic Feistel network. It makes use of following operations on n-bit words: $\oplus$, the bitwise XOR; $\wedge$, the bitwise AND; and left circular shifts. SIMON is a Feistel network with the following keyed round function

$$\begin{aligned} R_k : \mathbb{F}_2^n \times \mathbb{F}_2^n &\to \mathbb{F}_2^n \times \mathbb{F}_2^n \\ (x, y) &\mapsto (y \oplus f(x) \oplus k, x) \end{aligned}$$

where

$$f(x) = ((x \lll 1) \wedge (x \lll 8)) \oplus (x \lll 2)$$

and k is the round key. The number of rounds ranges from 32 for SIMON-32/64 to 72 for SIMON-128/256, and is given in detail in Table 3.12 on the facing page.

The designers of SIMON decided against including plaintext and ciphertext whitening steps, as inclusion of such operations can adversely affect circuit size.

The key schedules for $m = 2, 3, 4$ i.e. key lengths $2n, 3n$ and $4n$, respectively, are depicted in Figures 3.42, 3.43 and 3.44, where all paths are n bits wide. The key schedules are simple FSRs with input based on simple XORs and circular shifts.

One bit round constants are XORed in specifically for the purpose of eliminating slide proper-

ties and circular shift symmetries. These constants come from five periodic sequences $z_0, \ldots, z_4$, of periods 31 or 62. Sequences z_0 and z_1 have period 31, are generated by 5-bit LFSRs, and are used for block sizes of 32 and 48 bits:

$$z_0 = 11111010001001010110000111001 10\ldots$$
$$z_1 = 10001110111110010011000010110 10\ldots$$

Sequences z_2, z_3 and z_4 are for larger blocks. They are obtained by XORing the sequences z_0, z_1 and the period 31 sequence

$$10001110111110010011000010110 10\ldots$$

respectively with the period two sequence 01010101 ..., and thus have period 62. Which sequence is used for each block size/key size combination is detailed in Table 3.12. Details about the definition of the three period 31 sequence are found in the paper [BSS+13]. The constant c is the number $2^n - 4 = \texttt{ff...fc}_x$.

3.36.1.1 Remarks

The key schedule is a fascinating aspect of SIMON and, as we shall see, also of SPECK. In fact Bruce Schneier commented thusly on his blog: "*I was most impressed with their key schedule. I am always impressed with how the NSA does key schedules*." It is very simple and uses little resources, yet it does slowly, relentlessly and irregularly confuse its output, even though it is essentially a linear operation.

In fact, all the non-linearity in SIMON comes from a single AND operation in the round function. And the two inputs are the same input rotated in a different way. Therefore, on average, a bias of the Hamming way of this operation towards the low values is to be expected. This is a drastic simplification of more classical approaches where the results of two AND operations are combined (where, for instance, one operand of an AND operation is the bitwise negation of an operand to the other AND operation, i.e., a 2-to-1 multiplexer). Furthermore, if we exclude the purely linear XOR with $x \lll 2$ in the round function, we easily see that $x \mapsto (x \lll 1) \wedge (x \lll 8)$ is not surjective (since the set of positions of zeros in the result is the union of the sets of positions of the zeros in $x \lll 1$ and in the rotation of the latter, no results with just a single zero bit are possible), and non surjective round functions have led to weaknesses in the past [RPW97].

3.36.1.2 Cryptanalysis

SIMON is a very recent cipher, and there only a few cryptanalytic results so far. In Table 3.13 on the facing page we report the best attack so far. Some more research has been recently reported, namely [ALLW13b] and [ABG+13], but the results therein are not included here because they are worse. The paper [AL13] contains results obtained with impossible differentials cryptanalysis, but these are consistently much worse than the results obtained via classical differentials cryptanalysis, so we omit them and refer to the paper for the actual results.

3.36.2 SPECK

SPECK makes use of the following operations on n-bit words: $\oplus$, the bitwise XOR; $+$, integer addition modulo 2^n; and left and right circular shifts.

Table 3.13: Cryptanalysis of SIMON – Published Results

Cipher	Rounds Attacked	Technique	Attack Complexity Time	Data	Memory	Reference
SIMON 32/64	19/32	Differential	2^{32}	2^{31}	–	[BRV14]
SIMON 48/72	18/36	Differential	$2^{43.3}$	$2^{46.4}$	2^{24}	[AL13]
SIMON 48/72	20/36	Differential	2^{52}	2^{46}	–	[BRV14]
SIMON 48/96	18/36	Differential	$2^{43.3}$	$2^{46.4}$	2^{24}	[AL13]
SIMON 48/96	20/36	Differential	2^{75}	$2^{46.4}$	–	[BRV14]
SIMON 64/96	24/42	Differential	$2^{58.4}$	$2^{62.0}$	2^{32}	[AL13]
SIMON 64/96	26/42	Differential	$2^{63.9}$	$2^{63.0}$	2^{31}	[ALWL14]
SIMON 64/128	24/44	Differential	$2^{58.4}$	$2^{62.0}$	2^{32}	[AL13]
SIMON 64/128	26/44	Differential	2^{94}	$2^{63.0}$	2^{31}	[ALWL14]
SIMON 96/96	35/52	Differential	$2^{93.3}$	$2^{93.2}$	$2^{37.8}$	[ALWL14]
SIMON 96/144	29/54	Differential	$2^{83.7}$	$2^{87.5}$	2^{48}	[AL13]
SIMON 96/96	35/54	Differential	$2^{101.1}$	$2^{93.2}$	$2^{37.8}$	[ALWL14]
SIMON 128/128	40/68	Differential	$2^{120.5}$	$2^{124.8}$	2^{64}	[AL13]
SIMON 128/128	46/68	Differential	$2^{125.7}$	$2^{125.6}$	$2^{40.6}$	[ALWL14]
SIMON 128/192	40/69	Differential	$2^{120.5}$	$2^{124.8}$	2^{64}	[AL13]
SIMON 128/192	46/69	Differential	2^{142}	$2^{125.6}$	$2^{40.6}$	[ALWL14]
SIMON 128/256	40/72	Differential	$2^{120.5}$	$2^{124.8}$	2^{64}	[AL13]
SIMON 128/256	46/72	Differential	2^{206}	$2^{125.6}$	$2^{40.6}$	[ALWL14]

SPECK is an iterated cipher with following keyed round function

$$R_k : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n \times \mathbb{F}_2^n$$
$$(x, y) \mapsto (((x \ggg \alpha) + y) \oplus k, (y \lll \beta) \oplus ((x \ggg \alpha) + y) \oplus k)$$

with rotation amounts $\alpha = 7$ and $\beta = 2$ for the 32-bit block size and $\alpha = 8$ and $\beta = 3$ in all other cases. A graphical representation of a round of SPECK is given in Figure 3.45 on the next page. The number of rounds ranges from 22 for SPECK-32/64 to 32 for SPECK-128/256. The exact values for each block size/key size combination are given in Table 3.12 on page 220.

The designers acknowledge that the SPECK round function shares similarities to the mixing function of the Threefish block cipher (cf. Figure 3.34 on page 206 in Section 3.30), while at the same pointing out some significant differences (for instance, rotation amounts are fixed, there are no word permutations). Also, it is immediately seen that each round of SPECK can be split into two heterogeneous Feistel rounds which are performed in alternating fashion, namely,

$$R^{(1)}(x, y) = (y, ((x \ggg \alpha) + y) \oplus k)$$
$$R^{(2)}(x, y) = (y, (x \lll \beta) \oplus y) \ .$$

The key schedule, represented in Figure 3.46 on the following page, is very clever. It reuses the round function R. The secret key is written as $K = \ell_{m-2} \| \ldots \| \ell_0 \| k_0$ as a concatenation of n-bit words, where $\ell_0, \ell_1, \ldots, \ell_{m-2}$ are copied into a circular buffer and k_0 in a separate register. Then both k_0 and the last word of the circular buffer are fed to the round function R using the round

Figure 3.45: A Round of SPECK

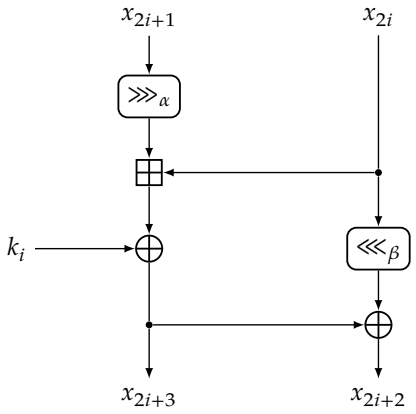
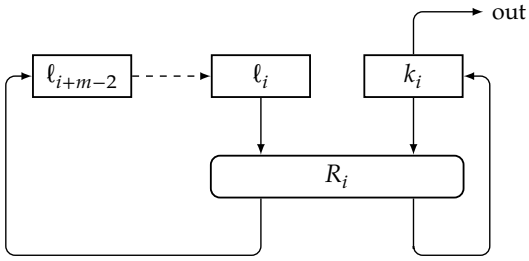


Figure 3.46: The SPECK Key Schedule



number i as the key:

$$\begin{aligned} \ell_{i+m-1} &= (k_i + (\ell_i \ggg \alpha)) \oplus i \\ k_{i+1} &= (k_i \lll \beta) \oplus \ell_{i+m-1} \end{aligned} .$$

One of the two outputs of the round function is placed in the circular buffer, the other output is the successive round key.

Similarly to SIMON, also SPECK does not include plaintext and ciphertext whitening steps.

3.36.2.1 Remarks

With respect to SIMON, the round function also reveals a minimalistic source of non-linearity: a single arithmetic addition, which, for inputs of low density, is well approximable with linear functions. The key schedule, on the other hand, is non linear, and this may be an improvement.

3.36.2.2 Cryptanalysis

SPECK is a very recent cipher, and there only a few cryptanalytic results so far. In Table 3.14 on the next page we report the complexities of some recent attacks. The paper [ALWL14] contains also several results obtained via rectangle attacks: these always worse than the differential cryptanalysis and therefore we omitted them (with one exception). However we refer the reader to the paper for the technique and more the detailed results.

3.37 A Miscellanea of Recent Lightweight Designs

3.37.1 KLEIN

KLEIN (which means "small" in German), proposed in 2010 by Zheng Gong, Svetla Nikova and Yee-Wei Law [GNL11] is another "mini AES." It is a SPN with a 64-bit block size and key sizes of 64, 80 and 96 bits, with 12, 16 or 20 rounds, respectively. The 64-bit state is treated as a vector of 16 nibbles. The operations on the state are:

1. Adding a round key.

Table 3.14: Cryptanalysis of SPECK – Published Results

Cipher	Rounds Attacked	Technique	Attack Complexity Time	Data	Memory	Referemnce
SPECK 32/64	10/22	Differential	$2^{29.2}$	2^{29}	2^{16}	[ALWL14]
SPECK 32/64	11/22	Differential	2^{46}	2^{14}	2^{22}	[Din14]
SPECK 32/64	12/22	Differential	2^{51}	2^{19}	2^{22}	[Din14]
SPECK 32/64	13/22	Differential	2^{57}	2^{25}	2^{22}	[Din14]
SPECK 32/64	14/22	Differential	2^{63}	2^{31}	2^{22}	[Din14]
SPECK 48/72	12/22	Differential	2^{43}	$2^{43.2}$	–	[BRV14]
SPECK 48/72	14/22	Differential	2^{65}	2^{41}	2^{22}	[Din14]
SPECK 48/96	12/23	Differential	2^{43}	$2^{43.2}$	–	[BRV14]
SPECK 48/96	15/23	Differential	2^{89}	2^{41}	2^{22}	[Din14]
SPECK 64/96	15/26	Differential	$2^{61.1}$	2^{61}	2^{32}	[ALWL14]
SPECK 64/96	16/26	Differential	2^{63}	2^{63}	–	[BRV14]
SPECK 64/96	18/26	Differential	2^{83}	2^{61}	2^{22}	[Din14]
SPECK 64/128	16/27	Differential	2^{63}	2^{63}	–	[BRV14]
SPECK 64/128	19/27	Differential	2^{125}	2^{61}	2^{22}	[Din14]
SPECK 96/96	15/28	Differential	$2^{89.1}$	2^{89}	2^{48}	[ALWL14]
SPECK 96/96	16/28	Differential	2^{85}	2^{84}	2^{22}	[Din14]
SPECK 96/144	15/29	Differential	$2^{89.1}$	2^{89}	2^{48}	[ALWL14]
SPECK 96/144	17/29	Differential	2^{133}	2^{85}	2^{22}	[Din14]
SPECK 128/128	16/32	Differential	$2^{111.1}$	2^{116}	2^{64}	[ALWL14]
SPECK 128/128	17/32	Differential	$2^{122.1}$	2^{122}	2^{64}	[AL13]
SPECK 128/128	17/32	Differential	2^{113}	2^{113}	2^{22}	[Din14]
SPECK 128/192	16/33	Differential	$2^{111.1}$	2^{116}	2^{64}	[ALWL14]
SPECK 128/192	18/33	Differential	2^{177}	2^{113}	2^{22}	[Din14]
SPECK 128/256	16/34	Differential	$2^{111.1}$	2^{116}	2^{64}	[ALWL14]
SPECK 128/256	18/34	Rectangle	$2^{182.7}$	$2^{125.9}$	$2^{121.9}$	[ALWL14]
SPECK 128/256	19/34	Differential	2^{241}	2^{113}	2^{22}	[Din14]

2. Transforming all nibbles through a 4-bit S-box.

3. Permuting the nibbles (using a cyclic rotation).

4. A linear combination of the nibbles (actually done in parallel on two groups of 8).

The S-box was chosen to satisfy conditions similar to the PRESENT (Section 3.29 on page 204) S-box, but since it is involutive some conditions had to be slightly relaxed.

The key schedule is a kind of Feistel network. It is agile even if keys are frequently changed and is designed to avoid potential related-key attacks.

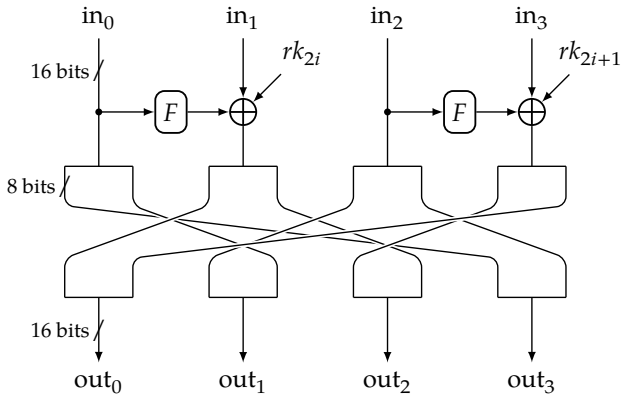
We are aware of no patents encumbering KLEIN.

A survey of the current attacks on KLEIN is given in Table 3.15 on the next page.

Table 3.15: Cryptanalysis of KLEIN – Published Results

Version	Rounds Attacked	Technique	Attack Complexity Time	Data	Memory	Reference
64	7/12	Integral	$2^{45.5}$	$2^{34.3}$	2^{32}	[YWLZ11]
64	8/12	Truncated diff.	$2^{46.8}$	2^{32}	2^{16}	[YWLZ11]
64	8/12	Differential	2^{35}	2^{35}	–	[ANPS11]
64	10/12	Parallel-Cut MITM	2^{62}	1	2^{60}	[NWW13b]
64	Full	Biclique	$2^{62.84}$	2^{39}	$2^{4.5}$	[ASA13]
64	Full	Differential	$2^{57.07}$	$2^{54.5}$	2^{16}	[LNP14]
80	8/16	Integral	$2^{77.5}$	$2^{34.3}$	2^{32}	[YWLZ11]
80	11/16	Parallel-Cut MITM	2^{74}	2	2^{74}	[NWW13b]
80	13/16	Differential	2^{76}	2^{52}	2^{16}	[LNP14]
80	Full	Biclique	2^{79}	2^{48}	2^{60}	[AFL+12]
96	13/20	Parallel-Cut MITM	2^{94}	2	2^{82}	[NWW13b]
96	14/20	Differential	$2^{89.2}$	$2^{58.4}$	2^{16}	[LNP14]
96	Full	Biclique	$2^{95.18}$	2^{32}	2^{60}	[AFL+12]

Figure 3.47: A Round of Piccolo



3.37.2 Piccolo

Piccolo (which means "small" in Italian), was introduced by SONY in 2011 [SIH+11]. Its design is similar to a Type 2 4-branch Feistel network with an extremely compact 4-bit S-box and simple key schedule.

Key lengths of 80 and 128 bits are supported, with respectively 25 and 31 rounds. A round is represented in Figure 3.47. The four branches are 16 bits wide, but in place of a simple circular permutation of the branches there is a more complex byte permutation – hence the four branches are each split in two.

The F-function is simple: first the S-box is applied in parallel to all four nibbles, then these are interpreted as a vector of four elements of $\mathbb{F}_{2^4}$ and are multiplied by an MDS matrix, then there is another S-box application. This is the SPS F-function type The S-box is very simple, and can be implemented with only 12 GE. Key scheduling is just bit extraction from the secret key.

Adding decryption support to a Piccolo encryption implementation requires little additional

area or code.

We are not aware of attacks that break the full cipher. The best cryptanalytic results we are aware of are:

- The biclique cryptanalysis in [WWY12] breaks full round Piccolo-80 without post whitening keys, resp. 28-round Piccolo-128 without pre whitening keys, with time complexity of $2^{78.95}$, resp. $2^{126.79}$ encryptions.
- In [JKL+12] attacks are described on the full cipher Piccolo-80/128 with computational complexities of $2^{79.13}$ and $2^{127.35}$, respectively.

U.S. Patent Application 2014/0003603 covers, among other things, various types of Feistel Networks and Lai-Massey networks where the F-function is of type SPS. As described in Subsection 1.8.4 on page 53, this is a line of research pursued by Andrey Bogdanov and Kyoji Shibutani [BS13b]. In particular, F-functions of SPS type are covered which alternate two different S-Boxes in each S layer. The idea of alternating two different types of S-boxes in the S layer of a SP F-function is already found in CLEFIA (Section 3.28 on page 201), also a SONY cipher.

3.37.3 MIBS

The MIBS cipher was introduced at CANS 2009 [ISSK09] by Maryam Izadi, Babak Sadeghiyan, Seyed Saeed Sadeghian, and Hossein Arabnezhad Khanooki. It was designed for resource-constrained devices, such as low-cost RFID tags, with a small gate count in hardware implementations.

It is a 64-bit block cipher using keys of 64 or 80 bits. The structure is a simple balanced 2-branch Feistel Network design, with 32 rounds for both key sizes.

All internal operations in MIBS are nibble-wise, that is, on 4-bit words. The round function of MIBS has an SPN structure consisting of an XOR layer with a round subkey, a substitution layer consisting of eight instances of the same 4-bit S-box, and a linear transformation layer (with branch number five), in this order.

The key schedule is a variant of the key schedule of PRESENT (Section 3.29 on page 204).

Asli Bay, Jorge Nakahara and Serge Vaudenay mount several types of attacks on reduced round MIBS [HWG10]. They break up to 18 rounds faster than exhaustive search.

3.37.4 LED

The LED cipher is a 64-bit cipher with key sizes of 64 to 128 bits. The first version was presented at CHES 2011 [GPPR11], and a revision was disclosed later [GPPR12]. The cipher is an ongoing project with its own web page. LED is an acronym that stands for Light Encrypting Device, but it is also a pun on the fact that LEDs produce *light* using little energy.

The cipher aims to be as small as PRESENT (Section 3.29 on page 204), and roughly similarly performing (in fact, it is faster in SW, slower in HW), while avoiding some of the attacks that have affected other ultra lightweight ciphers in the recent past. It is also designed to be realised with a very compact serial hardware architecture.

It is an interesting twist on the "mini AES" idea because key mixing is not performed at each

round. Round keys are added every four rounds, reducing latency. Each round is composed of (i) XOR of fixed constants, (ii) parallel application of the PRESENT S-box to all 16 nibbles in the state, (iii) `ShiftRows` and (iv) `MixColumns` using an MDS matrix. The constants are added to only a fixed half of the state. A group of four rounds is called a *step*. The number of rounds varies from 32 to 48 for 64 to 128-bit keys.

Key schedule is almost nonexistent: With a 64-bit key, there is only one sub key which is the secret key itself. With 128-bit keys, the secret key is split into two 64-bit parts which are used as subkeys in alternating fashion. Slide attacks (Section 2.6.2 on page 107) do not apply because different round constants are used in each step.

The paper [GPPR12] contains also extensive HW area/performance comparisons.

3.38 Ciphers that are Efficient when Masked

A recent development direction is the search for ciphers that are more efficient than existing designs when implemented to be higher order side-channel analysis resistant – even if, as a result, unprotected implementations are somewhat inefficient.

3.38.1 PICARO

The block cipher PICARO was introduced by Gilles Piret, Claude Carlet and Thomas Roche at ACNS 2012 [PRC12]. It is a balanced 2-branch Feistel network, with a block size of 128 bits and a key size of 128 bits.

The starting point for the design of PICARO are 8-bit S-boxes which are meant to be evaluated as efficiently as possible in a higher order side-channel analysis resistant implementation – this is contrast to the usual approach of taking an existing design and hardening its implementation. The method for hardening the implementation is a Boolean masking scheme for software implementations described by Matthieu Rivain and Emmanuel Prouff [RP10], based on Yuval Ishai, Amit Sahai and David Wagner's hardware-oriented masking scheme [ISW03], and further developed in [CGP+12, PR13].

A d^{th} order masking scheme as presented in [RP10] is an algorithmic countermeasure that thwarts d^{th} DSCA (differential side channel analysis); the data processed by the cipher is randomised in such a way that there exists no set of d processed side channel trails that together depend on the secret. In order to achieve this, each elementary boolean or arithmetic variable x in a domain D is replaced by a vector consisting of $d+1$ **shares** $x_0, x_1, \ldots, x_d$ that satisfy a relation $x_0 * x_1 * \cdots * x_d = x$, where $*$ is a group operation in D or a superset thereof – this is thus related to the concept of secret sharing as understood in MPC (multi party computation). The actual values of the shares are randomised, and each operation on one or more variables is then replaced by parallel or serial related operations on the vectors of shares: for instance,if $*$ is the XOR, operations such as shifts and rotations are replaced by $d+1$ shifts and rotations. There are several masking schemes, and the paper [PRC12] contains a little survey thereof. Masking schemes make implementations considerably more expensive, and therefore the search for building blocks for ciphers that lend themselves to a "lighter" masked realisation is a very interesting research problem.

Because of this research goal, the choice fell on a S-box with relatively poor (i.e. high) nonlinearity. Also, the S-box is non-injective, causing some input differences to produce vanishing

Table 3.16: Performance Comparisons of AES and PICARO

Number of Kcycles for encryption		
Version	AES	PICARO
Unprotected	2	26
Masked order 1	129	94
Masked order 2	271	160
Masked order 3	470	253

output differences. These features would make the cipher weak (or require too many rounds) if used in a F-function constructed as a simple SPN (i.e. key mixing, an array of 8 S-boxes, a diffusion layer).

These suboptimal cryptanalytic properties are countered by adopting an approach that recalls the DES: the input to the F-function is first expanded from 64 to 112 bits, then it is XORed to a 112-bit round key, and finally fed to an array of 14 S-boxes before being compressed to 64 bits.

The key schedule has been designed to be easy to implement; the round keys can be derived on-the-fly in both encryption and decryption mode. The round keys are obtained from the master key and a key derived from it by simple XORs of its parts by simple cyclic rotation and bit extraction.

If the master key is zero, all round keys are equal to zero as well. In fact there are four weak keys (one is all zeros, the second is all ones, the other two consist of alternating zeros and ones).

In Table 3.16, taken from [PRC12], the performance of PICARO is compared to that of the AES in variously masked implementations on a 8-bit micro controller. PICARO is much slower than the AES in an unprotected implementation. However, implementations masked of order one, two, or three are considerably faster than the corresponding implementations of the AES.

3.38.2 ZORRO

ZORRO is a block cipher presented at CHES 2013 [GGNPS13] by Benoît Gérard, Vincent Grosso, María Naya-Plasencia and François-Xavier Standaert. The design of ZORRO has the same goals as PICARO – i.e. producing a cipher that can be realised efficiently in a higher-order boolean masked implementation – but with the aim of recycling existing designs lessons as much as possible such as the AES rounds (which have been well studied for hardening against physical attacks), keeping bijective S-boxes to obtain better SCA-resistance, and reducing the total number of S-boxes by taking advantage of strong diffusion. Because of this, ZORRO aims to be the "masked hero," as its literary counterpart.

The cipher has a block size and key size of 128 bits. It is a SPN with 24 rounds, grouped into six *steps* of four rounds each, and key mixing is applied only at before the first and after each step – with the same key used unchanged every time. The state is represented as the AES-128 state as a 4-by-4 state matrix of 8-bit entries.

After comparing several different designs for composite S-boxes which are easy to mask, the choice fell on a 4-round Feistel network with a linear mixing layer using a 4-bit S-box as the F-function.

Table 3.17: Performance Comparisons of AES, Noekeon, PICARO, Zorro, Robin, and Fantomas in Masked Implementations: Encryption time for a 128-bit block on an Atmel AtMega644p.

Number of cycles for encryption						
Version	AES	Noekeon	PICARO	Zorro	Robin	Fantomas
Unprotected	5,100	8,606	37,515	6,578	8,877	6,673
Masked order 1	209,165	65,736	177,573	97,867	60,988	49,060
Masked order 2	395,546	120,965	316,047	196,999	124,026	97,458
Masked order 3	616,136	177,943	503,707	321,057	187,420	143,440

Each round is a composition of four transforms: a modified `SubBytes` transform where, to minimise the number of S-boxes, the S-boxes are applied only to the first row of the state matrix; an `AddConstants` transform, in which the vector $[i, i, i, i \ll 3]$ is added to the first state rows where i is the round index; then `ShiftRows` and `MixColumns` follow.

On AtmelAtMega644p the cipher is significantly faster than PICARO for both unprotected and masked implementations.

An important observation stems from the fact that the S-boxes are applied only to the first row of the state matrix and not to the whole matrix. This allows part of the cipher to be represented as a kind of Matsui-like Feistel Network with linear mixing layers in place of branch permutations.

ZORRO is not patented [Sta14].

3.38.3 Robin and Fantomas

At FSE 2014, Vincent Grosso, Gaëtan Leurent, Fran cois-Xavier Standaert, and Kerem Varici presented design approaches for ciphers allowing for lightweight masked implementations that are based in a fundamental way on bit-slicing [GLFV14].

In particular they use small S-boxes that, while offering less than optimal linear and differential properties, can be implemented using very small circuits, hence with a short program consisting of logic operations. Such S-boxes will then require a higher number of rounds than optimal ones to achieve a comparable level of security: the secret to an efficient design is then to strike the right balance. Diffusion layers are table based.

As an example of the application of their design ideas, in [GLFV14] two ciphers are presented, called Robin and Fantomas.

Robin is involutional. It uses an 8-bit S-box obtained from a 4-bit one via a 3-round Feistel network (a similar approach was used in MISTY). The linear layer is involutional. It has 16 rounds.

Fantomas is not involutional. Its 8-bit S-box obtained is constructed as an unbalanced Feistel networks built from 3- and 5-bit S-boxes similarly to MISTY, to improve differential properties – as a result, Fantomas uses only 12 rounds instead of 16 and its designers claim that to achieve a comparable level of security.

Therefore, Fantomas is faster, but requires separate code paths for encryption and decryption. Robin is slightly slower, but encryption and decryption are unified, only the key schedule having to be run in the inverse order.

Performance numbers comparing Robin and Fantomas to AES, Noekeon and the other masked-lightweight designs described in this section are given in Table 3.17 on the facing page (the paper contains a chart, but we obtained the exact numbers from the authors).

Robin and Fantomas are not patented [Sta14].

Chapter 4

Comparisons

Whereas most public key primitives are based on long integer arithmetic (ECC, RSA, ElGamal, Diffie Hellman), binary field arithmetic (ECC, Diffie Hellman) or matrix arithmetic (McEliece, Niederreiter), which are roughly all implemented at the same level of efficiency in SW and HW, the situation is rather different for block ciphers and symmetric cryptography in general. One of the main reasons is the presence of operations such as bit permutation layers. On one hand a complex bit permutation layer is almost free in HW, whereas is can be extremely expensive in SW – on the other hand a bit sliced implementation in SW can be very competitive, but only when the mode of operation allows several instances of the cipher to run in parallel.

This means that primitives that heavily rely on these operations are expensive in SW and can be inexpensive in HW. Since general purpose CPUs are commonplace, one may want to design primitives that are very efficient on such platforms. This explains the success of AES and other ciphers which are essentially byte oriented.

Comparison of HW performance on FPGAs is very important in the literature, presumably for applications such as high bandwidth links or memory encryption (where inexpensive and fast HW is required), or Internet of Everything applications (where small gate counts lead to financial savings). In several contexts, inexpensive sensor nodes (Internet of Everything) with 8- or 16-bit CPUs are the second most important type of platform. Only after this come general purpose CPUs. In reporting comparisons we follow the same order, and the different weights of the sections reflect the weight in the literature.

4.1 Hardware Performance

We report here some performance results of keyed lightweight block ciphers in hardware. Table 4.1 on page 235 is obtained by merging several sources. As it is now common in the literature, throughput has been normalized at 100kHz in order to get a common baseline, In many cases it is not properly indicated whether the gate counts for the implementations contain only the encryption operation or also the decryption – this is a common problem. In fact, even the very thorough authors of [BSS+13] state: *It is important to note the difficulties inherent in the sort of comparison we are doing. Different authors implement their algorithms under differing assumptions: various cell libraries are used for hardware implementations, and a variety of assumptions are made for software implementations. In addition, it is not always clear what a particular author means, for example, by code size (is the decryption algorithm implemented or not?) or gate count (is the key schedule included?). All of this can make attempts at a fair comparison problematic.*

Only a few papers report power consumption, and even when they do, the power figures are often difficult to analyse and properly compare, a problem acknowledged by the Ecrypt II report on lightweight cryptographic primitives [BBG+10]. It is a well-known fact that at low fre-

quencies, as typical for low-cost applications, the power consumption is dominated by its static part, which is proportional to the amount of transistors involved. As frequencies increase the relationship becomes more complicated. Furthermore, the power consumption strongly depends on the used technology and, if estimated, it greatly varies with the simulation method. To address these issues and to reflect the time-area-power trade-off inherent in any hardware implementation a new figure of merit (FOM) was proposed in [BDN+10], to measure the efficiency of a cipher. The FOM is computed as $\frac{\text{bits per block} \times 10^9}{\text{cycles per block} \times \text{GE}^2}$, where GE is *gate equivalents*. We note that it is a figure that favors low gate designs. Hence, in Table 4.1 we omit the power values but, besides cycles per block, throughput at 100 KHz (in kb/sec), the area requirements (in GE), we report the FOM. The size of the manufacturing process of the FPGA used for the implementation is also given.

Some implementations are marked with an asterisk (*), other with a dagger (†):

- Results marked with an asterisk (*) usually are single clock (from 0.5 to 2.1 cycles per block) for fully pipelined, highly parallel implementations of the cipher to be used in non feedback modes of operation. The latency in these cases is equal (or very close) to the number of rounds and if the cipher is used in modes of operation with feedback the throughput has to be divided accordingly. These results are reported only to show how the hardware requirements increase in case one desires maximal throughput, i.e. of the order of magnitude of one block per clock cycle – the papers report the corresponding latency only in very rare cases, but it can be assumed (as a rule of thumb) that it is roughly corresponding to the clock count for the other, non-parallel implementations. (These caveats does not apply to all other implementations.)

- Results marked with a dagger (†) are also fully unrolled, but to maximize performance and reduce latency to 1 cycle – rounds could be intermixed and these implementations are in general not pipelined.

Under the column labeled Enc./Dec. we report whether only encryption is implemented or decryption as well. A lone question mark ("?") means we could not ascertain the information, whereas "E?" means we lean towards only encryption was implemented. For ARMADILLO we enter N/A as it is a multipurpose cryptographic primitive out of which one can build other primitives such as encryption, hashing and so on (at essentially no additional cost).

Ciphers are grouped by block size, and within each group they are ordered alphabetically.

Despite the high throughput per clock cycle, these two types of implementations usually have a very long critical path, which limits the top frequency. It is still not unusual that these implementations achieve throughputs of several Gb/s. This is the case, for instance, of the results reported in [EYCP00, SRQL02, KSGK04, BCG+12b], peaking with 17.4Gb/s and 33.25Gb/s implementations of Camellia [DID04, DID05].

In many cases implementations come in obvious pairs, for instance there are two for PRESENT-80 and Piccolo-80: these represent two different approaches to cipher implementation, where the slower implementation is cyclic and round-based, and the faster implementation is serialized. In general implementations whose latency is similar to the number of rounds (or a very small multiple thereof) are serialized – and there may be several approaches in between. For the details we refer to the papers.

Finally, GOST-FB and GOST-PS denote GOST with eight different S-Boxes as used by the Central Bank of Russian Federation, and GOST with only the PRESENT S-Box, respectively.

We note that precise clock information is not available for an abundance of block cipher FPGA implementations in the literature (for instance in [GC01]), so we had no baseline to scale the results and did not include them.

Table 4.1: Performance of Block Ciphers in HW

Algorithm	Block size (bits)	Key size (bits)	Enc./ Dec.	Cycles per block	Speed (kbps@ 100kHz)	Tech Process (µm)	Area (#GE)	FOM $\frac{bits \times 10^9}{cyc/blk \times GE^2}$	Ref.
KATAN32	32	80	E	255	12.5	0.13	802	194	[CDK09]
SIMON	32	64	E+D	271	11.8	0.13	535	413	[BSS+13]
SIMON	32	64	E+D	21.5	97	0.13	722	2,855	[BSS+13]
SPECK	32	64	E+D	385	8.3	0.13	642	159	[BSS+13]
SPECK	32	64	E+D	23	139.1	0.13	850	1,926	[BSS+13]
KATAN48	48	80	E	255	18.8	0.13	927	219	[CDK09]
SIMON	48	96	E+D	304	15.8	0.13	763	271	[BSS+13]
SIMON	48	96	E+D	37	129.7	0.13	1,062	1,150	[BSS+13]
SPECK	48	96	E+D	400	12	0.13	884	154	[BSS+13]
SPECK	48	96	E+D	24	200	0.13	1,254	1,272	[BSS+13]
CAST-128	64	128	E+D	16	400	0.04	2,600	592	[KSGK04]
CAST-128*	64	128	E+D	1	6400	0.04	24,200	110	[KSGK04]
DES	64	56	E	144	44.4	0.18	2,300	84	[LPPS07]
DES	64	56	E+D	28	228.57	3	3,000	254	[VHVM88]
DESL	64	56	E	144	44.4	0.18	1,848	130	[LPPS07]
DESXL	64	184	E	144	44.4	0.18	2,168	95	[LPPS07]
GOST-FB	64	256	E	264	24.24	0.18	800	379	[PLW10]
GOST-FB	64	256	E	32	200	0.18	1,000	2,000	[PLW10]
GOST-PS	64	256	E	264	24.24	0.18	651	572	[PLW10]
GOST-PS	64	256	E	32	200	0.18	1,017	1,934	[PLW10]
HIGHT	64	128	E	34	188	0.25	3,048	203	[HSH+06]
HIGHT	64	128	E+D	34	188	0.35	2,608	259	[LLYC09]
IDEA	64	128	E+D	9	711	0.04	1,852	2,073	[KSGK04]
IDEA*	64	128	E+D	1	6400	0.04	11,700	468	[KSGK04]
KASUMI	64	128	?		?		6,000		[ECR]
KATAN64	64	80	E	255	25.1	0.13	1,054	226	[CDK09]
KATAN64	64	80	E	85	75.3	0.13	1,269	468	[CDK09]
KHAZAD	64	128	E+D	9	711	0.04	2,250	1,405	[KSGK04]
KHAZAD*	64	128	E	1	6400	0.04	7,872	1,033	[SRQL02]
KLEIN-64	64	64	E+D	207	30.9	0.18	1,220	208	[GNL11]
KLEIN-80	64	80	E+D	271	23.6	0.18	1,478	108	[GNL11]
KLEIN-80	64	80	E+D	17	376	0.18	2,629	108	[SMMK12]
KLEIN-96	64	96	E+D	335	19.1	0.18	1,528	82	[GNL11]

Performance of Block Ciphers in HW (continued)

Algorithm	Block	Key	E+D	Cycles	T'put	Process	Gates	FOM	Ref.
LED-64	64	64	E?	1,248	5.1	0.18	966	55	[GPPR12]
LED-64	64	64	E?	32	200	0.18	2,695	275	web page
LED-80	64	80	E?	1,872	3.4	0.18	1,040	32	[GPPR12]
LED-80	64	80	E?	48	133.3	0.18	2,780	173	web page
LED-96	64	96	E?	1,872	3.4	0.18	1,116	27	[GPPR12]
LED-96	64	96	E?	48	133.3	0.18	2,866	162	web page
LED-128	64	128	E?	1,872	3.4	0.18	1,265	21	[GPPR12]
LED-128	64	128	E?	48	133.3	0.18	3,036	145	web page
LED-128	64	128	E+D	49	130.6	0.045	3,309	119	[BCG+12b]
LED-128†	64	128	E+D	1	6400	0.045	109,811	5	[BCG+12b]
mCrypton	64	64	E	13	492.3	0.13	2,420	841	[LK05]
mCrypton	64	64	E+D	13	492.3	0.13	3,473	408	[LK05]
mCrypton	64	96	E	13	492.3	0.13	2,681	685	[LK05]
mCrypton	64	96	E+D	13	492.3	0.13	3,789	343	[LK05]
mCrypton	64	128	E	13	492.3	0.13	2,949	566	[LK05]
mCrypton	64	128	E+D	13	492.3	0.13	4,108	292	[LK05]
MISTY-1	64	128	E+D	9	711	0.04	4,820	306	[KSGK04]
MISTY-1*	64	128	E	1	6400	0.04	8,386	910	[EYCP00]
MISTY-1*	64	128	E+D	0.5	12800	0.04	13,080	748	[KSGK04]
Piccolo-80	64	80	E+D	432	14.8	0.18	1,034	139	[SIH+11]
Piccolo-80	64	80	E+D	27	237	0.18	1,496	1,059	[SIH+11]
Piccolo-128	64	128	E+D	528	12.1	0.18	1,334	68	[SIH+11]
Piccolo-128	64	128	E+D	33	194	0.18	1,773	617	[SIH+11]
PRESENT-80	64	80	E	547	11.7	0.18	1,075	101	[RPLP08]
PRESENT-80	64	80	E	32	200	0.18	1,570	811	[BKL+07]
PRESENT-80†	64	80	E+D	1	6400	0.045	63,942	16	[BCG+12b]
PRESENT-128	64	128	E	559	11.4	0.18	1,391	59	[Pos09]
PRESENT-128	64	128	E	32	200	0.18	1,886	562	[BKL+07]
PRESENT-128	64	128	E+D	32	200	0.045	3,707	146	[BCG+12b]
PRESENT-128†	64	128	E+D	1	6400	0.045	68,908	13	[BCG+12b]
PRINCE	64	128	E+D	12	533.3	0.045	3,779	373	[BCG+12b]
PRINCE*	64	128	E+D	1	6400	0.045	8,260	938	[BCG+12b]
SIMON	64	96	E+D	720	8.9	0.13	815	134	[BSS+13]
SIMON	64	96	E+D	45	142.2	0.13	1,216	962	[BSS+13]
SIMON	64	128	E+D	768	8.3	0.13	968	89	[BSS+13]
SIMON	64	128	E+D	48	133.3	0.13	1,417	664	[BSS+13]
SPECK	64	96	E+D	876	7.3	0.13	918	87	[BSS+13]
SPECK	64	96	E+D	29	220.7	0.13	1,522	953	[BSS+13]
SPECK	64	128	E+D	930	6.9	0.13	1,058	61	[BSS+13]
SPECK	64	128	E+D	31	206.5	0.13	1,658	751	[BSS+13]
Triple DES	64	184	E+D	48	133.7	0.04	431	7,178	[KSGK04]
Triple DES*	64	184	E+D	1	6400	0.04	14.240	316	[KSGK04]
TWINE	64	80	E	393	16.2	0.18	1,011	159	[SMMK12]
TWINE	64	80	E	36	178	0.18	1,503	787	[SMMK12]
XTEA	64	128	?	112	57.1	0.13	3,490	47	[ECR]
SEA	96	96	E+D	50(?)	192(?)	0.13	449		[MSQ07]
SEA	96	96	E+D	93	103	0.13	3,758	73	[MSQ07]
SIMON	96	96	E+D	2,743	3.7	0.13	955	38	[BSS+13]
SIMON	96	96	E+D	216	44.4	0.13	1,088	375	[BSS+13]
SIMON	96	96	E+D	54	177.8	0.13	1,580	712	[BSS+13]
SPECK	96	96	E+D	2,824	3.4	0.13	1,012	33	[BSS+13]
SPECK	96	96	E+D	232	41.4	0.13	1,134	322	[BSS+13]
SPECK	96	96	E+D	30	320	0.13	2,058	756	[BSS+13]

Performance of Block Ciphers in HW (continued)

Algorithm	Block	Key	E+D	Cycles	T'put	Process	Gates	FOM	Ref.
AES	128	128	?	160	80	0.13	3,100	83	[ECR]
AES	128	128	E	226	56.6	0.13	2,400	98	[MPL+11]
AES*	128	128	E	2.1	6095	0.09	10,992	504	[EYCP00]
AES	128	128	E+D	10	1280	0.18	18,300	39	[SHAS07]
AES	128	128	E+D	5.5	1164	0.045	15,880	92	[BCG+12b]
AES†	128	128	E+D	1	12800	0.045	135,051	7	[BCG+12b]
ARMADILLO2-A	128	80	N/A	176	72	0.18	2,923	85	[BDN+10]
ARMADILLO2-A	128	80	N/A	44	291	0.18	4,030	179	[BDN+10]
Camellia	128	128	E+D	20	640	0.35	11,350	50	[AIK+00]
Camellia*	128	128	E+D	1	12800	0.12	5,368	4,442	[DID04]
Camellia*	128	128	E+D	0.5	25600	0.12	11,287	2,009	[DID05]
CLEFIA	128	128	E+D	18	711.1	0.09	5,979	199	[SSA+07]
CLEFIA	128	128	E+D	36	355.6	0.09	4,950	145	[SSA+07]
RC-6	128	128	E?	20	640	0.12	2,902	760	[Beu03]
RC-6*	128	128	E?	1	12800	0.12	8,554	1,749	[Beu03]
SEED	128	128	E+D	16	800	0.18	23,500	15	[SHAS07]
Serpent*	128	128	E	1	12800	0.09	9,004	1,579	[EYCP00]
SIMON	128	128	E+D	4,410	2.9	0.13	1,234	19	[BSS+13]
SIMON	128	128	E+D	560	22.9	0.13	1,317	132	[BSS+13]
SIMON	128	128	E+D	70	182.9	0.13	2,378	323	[BSS+13]
SPECK	128	128	E+D	4,250	3	0.13	1,280	18	[BSS+13]
SPECK	128	128	E+D	1,054	12.1	0.13	1,396	62	[BSS+13]
SPECK	128	128	E+D	34	376.5	0.13	3,012	415	[BSS+13]
ARMADILLO2-B	192	128	N/A	64	300	0.18	6,025	82	[BDN+10]
ARMADILLO2-B	192	128	N/A	256	75	0.18	4,353	40	[BDN+10]

It is not immediate to draw conclusions from such a wealth of data points. However, some general considerations can be done:

- AES actually achieves quite decent performance at relatively modest gate counts.
- SIMON and SPECK are among the most compact ciphers, however it seems to be difficult to produce single clock latency versions due to the number of rounds.
- PRINCE provides good performance at low gate counts. Reduced round versions seem to retain good security, making it suitable for applications such as memory encryption.
- Besides PRINCE, five ciphers, namely KHAZAD, MISTY, Camellia, Serpent and IDEA can be implemented in single cycle with a surprisingly low gate count: these designs must be studied further in order to understand what exactly in the design allows this property. In particular, the security achieved despite simple (in some cases very small) S-Boxes and low round counts are choices to be analyzed carefully. Similar considerations apply to RC-6, but the use of variable shifts can be a problem in some contexts.
- LED, Piccolo and PRESENT are very interesting for sensor applications, as they can be very compact both in HW and SW – however in order to get low latency the gate count (and critical path length) explodes. So they do not seem interesting for memory encryption.

Table 4.2: Performance of Block Ciphers in SW on a 8-bit ATMEL ATtiny45 [EGG+12]

Algorithm	Block Size	Code Size	RAM Use	Enc. cycles	Dec. cycles	energy νJ
DESXL	64	820	48	84,600	84,600	348.8
AES	128	1,659	33	4,600	7,010	19.2
HIGHT	64	402	32	19,500	20,100	79.8
IDEA	64	836	232	8,300	22,700	34.3
KASUMI	64	1,264	24	11,900	11,900	47.6
NOEKEON	128	364	32	23,500	23,500	95.9
TEA	64	648	24	7,400	7,540	30.3
mCrypton-64	64	1,076	28	16,500	22,700	68.0
SEA	96	426	24	41,600	40,900	173.7
KATAN64	64	338	18	72,000	88,500	289.2
KLEIN-80	64	1,268	18	6,100	7,700	25.1
PRESENT-80	64	1,000	18	11,300	13,600	45.3

4.2 Software Performance: 8 and 16 bit Microcontrollers

Comparing performance in software is even more difficult, since there are so many possible variations in software environments. Furthermore, the advances in CPU performance and the evolution of the various ISAs makes alignment of different benchmark results in software arguably even more difficult than in hardware.

However, there are some recent comparisons on microcontrollers which are relatively extensive and therefore can provide some possible baselines, such as [EKP+07, EGG+12, CMM13].

In [EGG+12] twelve block ciphers are implemented on an ATMEL ATtiny45 device, and the results are compared with those from [EKP+07]. The ATMEL ATtiny45 is a low power 8-bit AVR RISC-based microcontroller with 32 general purpose registers, and integrates 4KB of flash memory, a 256-Byte EEPROM, and 256 bytes of SRAM. We report the benchmarks on the ATMEL ATtiny45 in Table 4.2.

The authors of SIMON and SPECK have compared the software performance of their algorithms to other popular lightweight block ciphers in [BSS+13]. The 8-bit microcontroller of choice was the ATMEL ATmega128 running at 16 MHz, an architecture similar to the chip used by [EGG+12]. For both SIMON and SPECK, and for each block size/key size combination, three implementations are provided: one that minimizes RAM usage (apart from the state, which is not counted because processing is assumed to be done in place), one that minimizes code size (FLASH), and one that maximizes throughput thus consuming less energy, but accepting higher RAM and FLASH usage. A summary of their performance measurement, including results from [OBSC10], is given in Table 4.3.

One further notable recent comparison is [CMM13], where about twenty recent lightweight block ciphers have been implemented on the MSP 430, a 16-bit microcontroller from TI which is used in wireless sensor nodes. This is a completely different architecture than the Atmel chips in the other comparisons. We report their results in Table 4.4 on page 240.

NOEKEON is tested in both direct-key and indirect-key mode. The cipher MIBS is described in [ISSK09]. The authors of [CMM13] made three implementations of LED. The first one (LEDxx)

Table 4.3: Performance of Block Ciphers on a 8-bit ATMEL ATmega128 [BSS+13, OBSC10]

Algorithm	Block Size	Key Size	Code Size	RAM Use	Enc. cycles/byte
TWINE	64	80	1,304	414	271
PRESENT-80	64	80	487	0	1,333
KATAN64	64	80	272	18	9,143
KLEIN-80	64	80	766	18	762
SIMON (low RAM)	64	96	274	0	239
SIMON (low FLASH)	64	96	198	168	218
SIMON (low energy)	64	96	530	168	205
SPECK (low RAM)	64	96	182	0	144
SPECK (low FLASH)	64	96	152	108	154
SPECK (low energy)	64	96	556	104	114
KLEIN-96	64	96	766	18	955
SIMON (low RAM)	64	128	282	0	250
SIMON (low FLASH)	64	128	208	176	228
SIMON (low energy)	64	128	404	176	217
SPECK (low RAM)	64	128	186	0	150
SPECK (low FLASH)	64	128	160	112	160
SPECK (low energy)	64	128	596	108	118
PRESENT-128	64	128	487	0	1,333
SIMON (low RAM)	128	128	732	0	376
SIMON (low FLASH)	128	128	328	544	342
SIMON (low energy)	128	128	446	544	333
SPECK (low RAM)	128	128	396	0	167
SPECK (low FLASH)	128	128	214	264	165
SPECK (low energy)	128	128	388	256	139
AES	128	128	943	33	288
AES	128	128	1,912	432	125
AES	128	128	1,912	176	135

is a standard implementation that does not use tables. The others use 8 lookup tables for the S-Box, ShiftRows and MixColumns steps. In the implementations marked as "tdur", these tables are pre-computed and the cost of building them is not included. In the implementations marked as "tcalc", these tables are computed before encryption and decryption and these costs are included in the benchmark.

The performance results always include key schedule, so they do not give asymptotic throughput in modes where the key is constant. These results include the memory consumption, both for internal state (RAM) and code size (which includes static tables).

From the comparisons given in Tables 4.2, 4.4 and 4.3 we see that some ciphers can be implemented in a particularly compact way: TEA, XTEA, SIMON and SPECK are among the smallest, followed by NOEKEON, LED mCrypton, Piccolo, SEA and TWINE. KATAN is very small in [EGG+12] whereas KATAN has a big memory footprint in [CMM13] – a fact which the authors attribute to the flexibility of block sizes supported by their implementation.

TEA, XTEA and the AES are generally fast, as are SIMON and SPECK – the latter usually obtaining comparable speed with a much smaller memory footprint. KLEIN gives decent perfor-

Table 4.4: Performance of Block Ciphers on a 16-bit TI MSP 430 [CMM13]

Algorithm	Block size (bits)	code size (bytes)	RAM usage (bytes)	Encryption (incl. Key Schedule) (cycles/byte)	Decryption (cycles/byte)
AES	128	4,460	19	1,891	2,406
CLEFIA-128	128	4,780	180	6,134	6,365
CLEFIA-192	128	5,010	268	9,394	7,708
CLEFIA-256	128	4,924	268	9,728	9,080
DESXL	64	16,816	112	3,256	8,364
HIGHT	64	3,130	18	4,046	4,077
IDEA	64	3,140	82	3,925	20,422
KATAN32	32	5,816	1881	186,069	179,264
KATAN48	48	7,076	1969	187,878	175,613
KATAN64	64	8,348	1953	189,798	174,740
KLEIN-64	64	5,486	36	3,689	12,575
KLEIN-80	64	5,676	38	5,034	16,921
KLEIN-96	64	5,862	39	6,437	21,348
LBlock	64	3,568	13	5,369	2,750
LED-128	64	2,648	41	167,686	168,144
LED-128_tcalc	64	2,948	41	33,590	34,369
LED-128_tdur	64	2,264	41	21,382	21,729
LED-64	64	2,670	41	111,835	112,169
LED-64_tcalc	64	2,498	41	26,551	27,175
LED-64_tdur	64	2,264	41	14,359	14,535
mCrypton-64	64	2,726	18	13,475	27,483
mCrypton-96	64	2,834	20	13,562	27,540
mCrypton-128	64	3,108	24	13,551	27,571
MIBS64	64	3,184	29	6,132	6,611
MIBS80	64	3,138	16	7,336	4,980
NOEKEON (dir)	128	2,710	34	1,643	1,695
NOEKEON (ind)	128	2,784	34	3,285	3,339
PRESENT (size)	64	4,964	142	61,450	61,226
PRESENT (speed)	64	4,814	142	45,573	46,091
Piccolo-128	64	2,510	91	4,562	4,950
Piccolo-80	64	2,434	79	4,013	4,328
SEA	96	2,804	24	9,954	10,013
SKIPJACK	64	6,628	19	10,615	15,421
TEA	64	1,354	13	1,098	1,141
TWINE	128	2,216	23	5,125	3,808
XTEA	64	1,394	11	1,160	1,203

mance as well. IDEA has good encryption performance but decryption is very slow because of the inefficiencies of decryption key scheduling in very constrained environments – this limits its applications. KATAN, LED and mCrypton tend to be slow.

Table 4.5: Six device/server use cases for lightweight encryption from [BGLP13]. For the practical measurements given in Table 4.6 on the next page, the notation "big/small" refers to more/less than 10 on average – concretely, the values 1000 and 1 were used.

	D	B	op. mode	Example
1	small	small	–	authentication / access control / secure traceability (industrial assembly line)
2	small	big	parallel	secure streaming communication (medical device sending continuously sensitive data to a server, tracking data, etc.)
3	small	big	serial	secure serial communication
4	big	small	–	multi-user authentication / secure traceability (parallel industrial assembly lines)
5	big	big	parallel	multi-user secure streaming communication / cloud computing / smart meters server / sensors network / Internet of Things
6	big	big	serial	multi-user secure serial communication

4.3 Software Performance: Desktop CPUs

There is no lack of block cipher comparisons for general purpose CPUs, however they are very diverse, often referring to radically different implementations of the same ISAs, and encompass extremely different implementation techniques.

However, recently, a comparison of the performance of LED, PRESENT and Piccolo has been published [BGLP13] that tries to assess the performance of these lightweight ciphers on three different implementations of the x86-64 instruction set (a Core i3-2367M, a XEON X5650, and a Core 2 Duo P8600) using four different implementation techniques for the substitution layer: a table-lookup based implementation, the use of vperm on 16-byte vectors to make a quick lookup for 4-bit S-Boxes, and two different bit sliced implementations. Several different parameter choices are taken into consideration to define six use cases:

- A server communicates with D devices, all using distinct keys. For each device, the server has to encipher/decipher B 64-bit blocks of data. The cases where the ciphertext comes from a parallel mode of operation (like CTR) or a serial one (like CBC) are distinguished.
- t_E is the time required by the implementation one encryption without the key schedule and without the packing/unpacking of the input/output data – i.e. the conversion to the format used by the vperm or bitslice based implementations.
- t_{KS} is the time taken by the key schedule (without the packing of the key data).

The six use cases are described in Table 4.5 and the performance results in Table 4.6 on the next page (abridged from [BGLP13]). There are often two bit sliced implementations, that differ on the number of blocks processed simultaneously.

Table 4.6: Implementation results for LED, PRESENT and Piccolo on various x86 architectures.

Algorithm	Implementation Type (blocks)	t_E (cycles)	t_{KS} (cycles)	use cases speed (cycles/byte) 1	2	3	4	5	6
Implementation results for LED, PRESENT and Piccolo on Core i3-2367M @ 1.4GHz									
LED-64	table	608	0	76.0	76.0	76.0	76.0	76.0	76.0
	vperm	560	0	71.5	36.0	71.0	36.5	36.0	36.0
	bitslice (16)	2,443	0	307.5	20.7	306.8	21.3	20.5	20.7
	bitslice (32)	2,604	0	328.0	12.2	327.3	12.9	11.9	12.2
LED-128	table	906	0	113.3	113.3	113.3	113.3	113.3	113.3
	vperm	858	0	109.4	54.6	108.3	55.8	54.6	54.6
	bitslice (16)	3,671	0	461.6	30.4	460.3	31.6	30.1	30.4
	bitslice (32)	3,967	0	499.1	17.6	497.7	19.1	17.3	17.6
Piccolo-80	table	671	38	88.6	83.9	83.9	88.6	83.9	83.9
	vperm	512	55	73.1	33.3	65.3	37.7	33.3	33.3
	bitslice (16)	977	0	125.5	9.2	123.7	11.1	9.2	9.2
Piccolo-128	table	829	57	110.8	103.6	103.6	110.8	103.6	103.6
	vperm	644	55	89.6	41.6	81.9	45.9	41.6	41.6
	bitslice (16)	1,196	0	153.9	10.9	151.0	13.8	10.8	10.9
PRESENT-80	table	580	408	123.5	72.6	72.6	123.5	72.6	72.6
	vperm	540	350	123.4	35.1	68.8	67.8	35.0	35.0
	bitslice (8)	1,333	706	259.0	23.1	168.9	36.0	23.0	23.0
	bitslice (16)	2,038	1,100	394.5	17.5	256.2	27.0	17.3	17.4
PRESENT-128	table	580	334	114.3	72.5	72.5	114.3	72.5	72.5
	vperm	540	296	118.6	35.0	68.8	66.4	35.0	35.0
	bitslice (8)	1,313	738	260.6	22.8	166.5	36.3	22.8	22.8
	bitslice (16)	2,221	1,286	440.9	19.0	279.2	30.2	18.7	18.9
Implementation results for LED, PRESENT and Piccolo on XEON X5650 @ 2.67GHz									
LED-64	table	567	0	70.9	70.9	70.9	70.9	70.9	70.9
	vperm	749	0	96.1	48.1	94.9	49.3	48.1	48.1
	bitslice (16)	2,445	0	307.7	20.7	307.0	21.3	20.5	20.7
	bitslice (32)	2,846	0	358.2	13.1	357.5	13.9	12.9	13.1
LED-128	table	847	0	105.9	105.9	105.9	105.9	105.9	105.9
	vperm	1,058	0	135.8	67.4	133.5	69.7	67.4	67.4
	bitslice (16)	3,674	0	461.9	30.3	460.7	31.6	30.1	30.3
	bitslice (32)	4,306	0	541.5	19.0	540.0	20.5	18.6	19.0
Piccolo-80	table	568	39	75.9	71.0	71.0	75.9	71.0	71.0
	vperm	580	47	81.3	37.4	73.7	42.1	37.4	37.4
	bitslice (16)	1,038	0	133.1	9.7	131.3	11.5	9.6	9.7
Piccolo-128	table	700	62	95.3	87.5	87.5	95.3	87.5	87.5
	vperm	724	77	103.3	47.4	92.7	53.3	47.4	47.4
	bitslice (16)	1,400	0	179.3	12.5	176.5	15.4	12.4	12.5
PRESENT-80	table	525	398	115.4	65.7	65.7	115.4	65.7	65.7
	vperm	650	441	150.3	42.1	82.8	82.1	42.1	42.1
	bitslice (8)	1,360	600	249.3	23.6	172.4	34.9	23.6	23.6
	bitslice (16)	2,453	1,437	488.7	20.9	308.2	33.1	20.6	20.7
PRESENT-128	table	525	304	103.6	65.7	65.7	103.6	65.7	65.7
	vperm	650	408	152.8	42.1	82.8	86.6	42.1	42.1
	bitslice (8)	1,389	674	262.1	24.0	175.9	36.5	23.9	23.9
	bitslice (16)	2,882	1,888	598.8	24.3	361.9	40.2	23.9	24.1
Implementation results for LED, PRESENT and Piccolo on Core 2 Duo P8600 @ 2.4GHz									
LED-64	table	502	0	62.8	62.8	62.8	62.8	62.8	62.8
	vperm	751	0	94.9	47.4	94.4	47.9	47.4	47.4
	bitslice (16)	2,880	0	362.8	24.7	362.0	25.5	24.5	24.7
	bitslice (32)	3,029	0	381.5	14.2	380.7	15.0	13.9	14.2

Implementation results for LED, PRESENT and Piccolo on various x86 architectures (continued)

Algorithm	Implementation Type (blocks)	t_E (cycles)	t_{KS} (cycles)	Use cases speed (cycles/byte) 1	2	3	4	5	6
LED-128	table	748	0	93.5	93.5	93.5	93.5	93.5	93.5
	vperm	1,091	0	140.0	68.7	136.9	71.8	68.7	68.7
	bitslice (16)	4,219	0	531.0	35.2	529.4	36.8	35.0	35.2
	bitslice (32)	4,521	0	568.9	20.2	567.3	21.9	19.8	20.2
Piccolo-80	table	537	41	72.3	67.1	67.1	72.3	67.1	67.1
	vperm	594	44	82.8	38.3	75.4	42.9	38.3	38.3
	bitslice (16)	1,100	0	141.7	10.7	139.6	12.8	10.7	10.7
Piccolo-128	table	669	65	91.8	83.6	83.6	91.8	83.6	83.6
	vperm	739	73	103.5	47.2	93.4	52.8	47.2	47.2
	bitslice (16)	1,400	0	180.2	13.0	177.0	16.2	12.9	13.0
PRESENT-80	table	476	359	104.4	59.5	59.5	104.4	59.5	59.5
	vperm	651	384	144.1	42.3	83.0	79.4	42.3	42.3
	bitslice (8)	1,446	731	277.1	25.5	183.6	39.0	25.4	25.4
	bitslice (16)	2,438	1,250	464.1	21.2	306.7	32.1	20.9	21.0
PRESENT-128	table	476	285	95.1	59.5	59.5	95.1	59.5	59.5
	vperm	652	386	146.8	42.4	83.1	81.9	42.4	42.4
	bitslice (8)	1,472	812	290.5	25.8	186.8	40.6	25.7	25.7
	bitslice (16)	2,830	1,631	560.8	24.3	355.7	38.3	23.9	24.1

Even though the ciphers analyzed in [BGLP13] are probably not relevant for the applications we are interested in, the results signal that taking into account the possibility of using bit slicing in the (server side) implementation and non-feedback modes can be an important design consideration. Applications such as high-bandwith encryption for "1000×" networks would profit from this.

Bibliography

[3GP09] 3GPP. TS 35.202 V11.0.0; Technical Specification 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification (Release 11), September 2009.
Available from: http://www.3gpp.org/DynaReport/35202.htm.
One citation on page 175.

[AB96] Ross J. Anderson and Eli Biham. Two Practical and Provably Secure Block Ciphers: BEAR and LION. In Gollmann [Gol96], pages 113–120.
Available from: http://dx.doi.org/10.1007/3-540-60865-6_48.
2 citations on pages 33 and 68.

[Abe10] Masayuki Abe, editor. *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*. Springer, 2010.
Available from: https://doi.org/10.1007/978-3-642-17373-8,
DOI: doi:10.1007/978-3-642-17373-8.
3 citations on pages 262, 267, and 274.

[ABG+13] Javad Alizadeh, Nasour Bagheri, Praveen Gauravaram, Abhishek Kumar, and Somitra Kumar Sanadhya. Linear Cryptanalysis of Round Reduced SIMON. *IACR Cryptology ePrint Archive*, 2013:663, 2013.
Available from: http://eprint.iacr.org/2013/663.
One citation on page 222.

[ABK00] Ross J. Anderson, Eli Biham, and Lars R. Knudsen. The Case for Serpent. In *Proceedings of the Third AES Candidate Conference*, pages 349–354, 2000.
One citation on page 168.

[AC09] Martin R. Albrecht and Carlos Cid. Algebraic Techniques in Differential Cryptanalysis. In Dunkelman [Dun09], pages 193–208.
Available from: http://dx.doi.org/10.1007/978-3-642-03317-9_12.
2 citations on pages 75 and 205.

[Ada90] Carlisle Michael Adams. *A Formal and Practical Design Procedure for Substitution-Permutation Network Cryptosystems*. PhD thesis, Department of Electrical Engineering, Queen's University at Kingston, 1990. University Microfilms order no. UMI00281681.
2 citations on pages 27 and 146.

[Ada94] Carlisle Michael Adams. Provably Secure and Efficient Block Ciphers. In *Selected Areas in Cryptography 1994*, pages 129–133, 1994.
Available from: http://www.sacconference.org/SAC-history.html.
One citation on page 146.

[Ada97] Carlisle Michael Adams. Constructing Symmetric Ciphers Using the CAST Design Procedure. *Designs, Codes and Cryptography*, 12(3):283–316, 1997.
Available from: http://dx.doi.org/10.1023/A:1008229029587.
2 citations on pages 27 and 146.

[Ada98] Carlisle Michael Adams. CAST-256, A Submission for the Advanced Encryption Standard. In *First AES Candidate Conference*. National Institute of Standard and Technology, 1998.
Available from: http://csrc.nist.gov/archive/aes/round1/conf1/cast-slides.pdf.
2 citations on pages 146 and 147.

[AdlGMP96] Gonzalo Álvarez, Dolores de la Guia, Fausto Montoya, and Alberto Peinado. Akelarre: a new Block Cipher Algorithm. In Henk Meijer and Stafford Tavares, editors, *Selected Areas in Cryptography 1996*, page 14. Selected Areas in Cryptography Organizing Board, 1996.
Available from: http://www.sacconference.org/SAC-history.html.
One citation on page 37.

[ADMS09] Jean-Philippe Aumasson, Itai Dinur, Willi Meier, and Adi Shamir. Cube Testers and Key Recovery Attacks on Reduced-Round MD6 and Trivium. In Dunkelman [Dun09], pages 1–22.
Available from: http://dx.doi.org/10.1007/978-3-642-03317-9_1.
One citation on page 82.

[AF13a] Daniel Augot and Matthieu Finiasz. Exhaustive search for small dimension recursive MDS diffusion layers for block ciphers and hash functions. In *ISIT*, pages 1551–1555. IEEE, 2013.
Available from: http://dx.doi.org/10.1109/ISIT.2013.6620487.
One citation on page 51.

[AF13b] Daniel Augot and Matthieu Finiasz. Exhaustive Search for Small Dimension Recursive MDS Diffusion Layers for Block Ciphers and Hash Functions. *CoRR*, abs/1305.3396, 2013.
Available from: http://arxiv.org/abs/1305.3396.
One citation on page 51.

[AF14] Daniel Augot and Matthieu Finiasz. Direct Construction of Recursive MDS Diffusion Layers using Shortened BCH Codes. Preproceedings of FSE 2014, March 2014.
Available from: https://www.rocq.inria.fr/secret/Matthieu.Finiasz/research/2014/augot-finiasz-fse14.pdf.
One citation on page 51.

[AFI+04] Gwénolé Ars, Jean-Charles Faugère, Hideki Imai, Mitsuru Kawazoe, and Makoto Sugita. Comparison Between XL and Gröbner Basis Algorithms. In Lee [Lee04], pages 338–353.
Available from: http://dx.doi.org/10.1007/978-3-540-30539-2_24.
One citation on page 121.

[AFL+12] Farzaneh Abed, Christian Forler, Eik List, Stefan Lucks, and Jakob Wenzel. Biclique Cryptanalysis Of PRESENT, LED, And KLEIN. Cryptology ePrint Archive, Report 2012/591, 2012.
Available from: http://eprint.iacr.org/.
2 citations on pages 205 and 226.

[AGM+09] Kazumaro Aoki, Jian Guo, Krystian Matusiewicz, Yu Sasaki, and Lei Wang. Preimages for Step-Reduced SHA-2. In Matsui [Mat09], pages 578–597.
Available from: http://dx.doi.org/10.1007/978-3-642-10366-7_34.
One citation on page 95.

[AIK+00] Kazumaro Aoki, Tetsuya Ichikawa, Masayuki Kanda, Mitsuru Matsui, Shiho Moriai, Junko Nakajima, and Toshio Tokita. Camellia: A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis. In Stinson and Tavares [ST01], pages 39–56.
Available from: http://dx.doi.org/10.1007/3-540-44983-3_4.
2 citations on pages 170 and 237.

[ÅJ11a] Martin Ågren and Thomas Johansson. Linear Cryptanalysis of PRINTcipher - Trails and Samples Everywhere. *IACR Cryptology ePrint Archive*, 2011:423, 2011.
Available from: http://eprint.iacr.org/2011/423.
One citation on page 211.

[ÅJ11b] Martin Ågren and Thomas Johansson. Linear Cryptanalysis of PRINTcipher - Trails and Samples Everywhere. In Bernstein and Chatterjee [BC11], pages 114–133.
Available from: http://dx.doi.org/10.1007/978-3-642-25578-6_10.
One citation on page 211.

[AL12] Martin R. Albrecht and Gregor Leander. An All-In-One Approach to Differential Cryptanalysis for Small Block Ciphers. In Knudsen and Wu [KW13], pages 1–15.
Available from: http://dx.doi.org/10.1007/978-3-642-35999-6_1.
4 citations on pages 44, 85, 86, and 87.

[AL13] Hoda AlKhzaimi and Martin M. Lauridsen. Cryptanalysis of the SIMON Family of Block Ciphers. *IACR Cryptology ePrint Archive*, 2013:543, 2013.
Available from: http://eprint.iacr.org/2013/543.
3 citations on pages 222, 223, and 225.

[ALL12] Farzaneh Abed, Eik List, and Stefan Lucks. On the Security of the Core of PRINCE Against Biclique and Differential Cryptanalysis. Cryptology ePrint Archive, Report 2012/712, 2012.
Available from: http://eprint.iacr.org/2012/712.
One citation on page 219.

[ALLW13a] Farzaneh Abed, Eik List, Stefan Lucks, and Jakob Wenzel. Cryptanalysis of the Speck Family of Block Ciphers. *IACR Cryptology ePrint Archive*, 2013:568, 2013. Merged with [ALLW13b] into [ALWL14].
Available from: http://eprint.iacr.org/2013/568.
One citation on page 247.

[ALLW13b] Farzaneh Abed, Eik List, Stefan Lucks, and Jakob Wenzel. Differential Cryptanalysis of Reduced-Round Simon. *IACR Cryptology ePrint Archive*, 2013:526, 2013. Merged with [ALLW13a] into [ALWL14].
Available from: http://eprint.iacr.org/2013/526.
2 citations on pages 222 and 247.

[ALWL14] Farzaneh Abed, Eik List, Jakob Wenzel, and Stefan Lucks. Differential Cryptanalysis of round-reduced Simon and Speck. Preproceedings of FSE 2014, March 2014. Merged and updated version of [ALLW13b] and [ALLW13a].
4 citations on pages 223, 224, 225, and 247.

[ALZ11] Mohamed Ahmed Abdelraheem, Gregor Leander, and Erik Zenner. Differential Cryptanalysis of Round-Reduced PRINTcipher: Computing Roots of Permutations. In Joux [Jou11], pages 1–17.
Available from: http://dx.doi.org/10.1007/978-3-642-21702-9_1.
One citation on page 211.

[And94] Ross J. Anderson, editor. *Fast Software Encryption, Cambridge Security Workshop, Cambridge, UK, December 9-11, 1993, Proceedings*, volume 809 of *Lecture Notes in Computer Science*. Springer, 1994.
3 citations on pages 261, 280, and 290.

[ANPS11] Jean-Philippe Aumasson, María Naya-Plasencia, and Markku-Juhani Olavi Saarinen. Practical Attack on 8 Rounds of the Lightweight Block Cipher KLEIN. In Bernstein and Chatterjee [BC11], pages 134–145.
Available from: http://dx.doi.org/10.1007/978-3-642-25578-6_11.
One citation on page 226.

[AS08] Kazumaro Aoki and Yu Sasaki. Preimage Attacks on One-Block MD4, 63-Step MD5 and More. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *SAC 2008*, volume 5381 of *Lecture Notes in Computer Science*, pages 103–119. Springer, 2008.
Available from: http://dx.doi.org/10.1007/978-3-642-04159-4_7.
2 citations on pages 95 and 101.

[AS09] Kazumaro Aoki and Yu Sasaki. Meet-in-the-Middle Preimage Attacks Against Reduced SHA-0 and SHA-1. In Halevi [Hal09], pages 70–89.
Available from: http://dx.doi.org/10.1007/978-3-642-03356-8_5.
One citation on page 101.

[ASA13] Zahra Ahmadian, Mahmoud Salmasizadeh, and Mohammad Reza Aref. Biclique Cryptanalysis of the Full-Round KLEIN Block Cipher. *IACR Cryptology ePrint Archive*, 2013:97, 2013.
Available from: http://eprint.iacr.org/2013/097.
One citation on page 226.

[AT89] Carlisle Michael Adams and Stafford E. Tavares. Good S-Boxes Are Easy To Find. In Brassard [Bra90], pages 612–615.
Available from: http://dx.doi.org/10.1007/0-387-34805-0_56.
One citation on page 146.

[AT90a] Carlisle Michael Adams and Stafford E. Tavares. The Structured Design of Cryptographically Good S-Boxes. *Journal of Cryptology*, 3(1):27–41, 1990.
Available from: http://dx.doi.org/10.1007/BF00203967.
One citation on page 146.

[AT90b] Carlisle Michael Adams and Stafford E. Tavares. Technical Report TR 90-013: The Use of Bent Sequences to Achieve Higher-Order Strict Avalanche Criterion in S-Box Design. Technical report, Department of Electrical Engineering, Queen's University, Kingston, Ontario, January 1990.
2 citations on pages 60 and 146.

[AT93] Carlisle Michael Adams and Stafford E. Tavares. Designing S-Boxes For Ciphers Resistant To Differential Cryptanalysis. In *Proceedings Of The 3Rd Symposium On State And Progress Of Research In Cryptography*, pages 181–190, 1993.
Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.40.2536.
2 citations on pages 27 and 146.

[AV03] Kazumaro Aoki and Serge Vaudenay. On the Use of GF-Inversion as a Cryptographic Primitive. In Mitsuru Matsui and Robert J. Zuccherato, editors, *Selected Areas in Cryptography 2003*, volume 3006 of *Lecture Notes in Computer Science*, pages 234–247. Springer, 2003.
Available from: http://dx.doi.org/10.1007/978-3-540-24654-1_17.
One citation on page 66.

[BAB93] Ishai Ben-Aroya and Eli Biham. Differential Cryptanalysis of Lucifer. In Stinson [Sti94], pages 187–199.
Available from: http://dx.doi.org/10.1007/3-540-48329-2_17.
One citation on page 127.

[BAB96] Ishai Ben-Aroya and Eli Biham. Differential Cryptanalysis of Lucifer. *J. Cryptology*, 9(1):21–34, 1996.
Available from: http://dx.doi.org/10.1007/BF02254790.
One citation on page 127.

[BAK98] Eli Biham, Ross J. Anderson, and Lars R. Knudsen. Serpent: A New Block Cipher Proposal. In Vaudenay [Vau98a], pages 222–238.
Available from: http://dx.doi.org/10.1007/3-540-69710-1_15.
One citation on page 168.

[BB94] Eli Biham and Alex Biryukov. How to Strengthen DES Using Existing Hardware. In Pieprzyk and Safavi-Naini [PSN95], pages 398–412.
Available from: http://dx.doi.org/10.1007/BFb0000451.
One citation on page 163.

[BB97] Eli Biham and Alex Biryukov. An Improvement of Davies' Attack on DES. *Journal of Cryptology*, 10(3):195–206, 1997.
Available from: http://dx.doi.org/10.1007/s001459900027.
3 citations on pages 119, 120, and 130.

[BBD+98] Eli Biham, Alex Biryukov, Orr Dunkelman, Eran Richardson, and Adi Shamir. Initial Observations on Skipjack: Cryptanalysis of Skipjack-3XOR. In Stafford E. Tavares and Henk Meijer, editors, *Selected Areas in Cryptography*, volume 1556 of *Lecture Notes in Computer Science*, pages 362–376. Springer, 1998.
Available from: http://dx.doi.org/10.1007/3-540-48892-8_27.
One citation on page 165.

[BBF+99] Eli Biham, Alex Biryukov, Niels Ferguson, Lars R. Knudsen, Bruce Schneier, and Adi Shamir. Cryptanalysis of Magenta. In *Proceedings of the Second AES Candidate Conference*, March 1999.
Available from: http://www.schneier.com/paper-magenta.html.
3 citations on pages 101, 179, and 180.

[BBG+10] Tor E. Björstad, Andrey Bogdanov, Henri Gilbert, Kota Ideguchi, Sebastiaan Indesteege, özgül Küçük, Gregor Leander, Nicky Mouha, Jorge Nakahara, Jr., Axel Poschmann, Christian Rechberger, Vincent Rijmen, Gautham Sekar, Kyoji Shibutani, Martin Schläffer, François-Xavier Standaert, Elmar Tischhauser, Vesselin Velichkov, and Ivan Visconti. Ecrypt II Deliverable D.SYM.5 – WG2 Lightweight Cryptographic Algorithms. Technical report, ECRYPT II, 2010.
One citation on page 233.

[BBS86] Lenore Blum, Manuel Blum, and Mike Shub. A Simple Unpredictable Pseudo-Random Number Generator. *SIAM Journal Computing*, 15(2):364–383, 1986.
Available from: http://dx.doi.org/10.1137/0215025.
One citation on page 65.

[BBS99a] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In Stern [Ste99], pages 12–23.
Available from: http://dx.doi.org/10.1007/3-540-48910-X_2.
2 citations on pages 85 and 165.

[BBS99b] Eli Biham, Alex Biryukov, and Adi Shamir. Miss in the Middle Attacks on IDEA and Khufu. In Knudsen [Knu99], pages 124–138.
Available from: http://dx.doi.org/10.1007/3-540-48519-8_10.
One citation on page 85.

[BBS05] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. *Journal of Cryptology*, 18(4):291–311, 2005.
Available from: http://dx.doi.org/10.1007/s00145-005-0129-3.
One citation on page 85.

[BC04] Eli Biham and Rafi Chen. Near-Collisions of SHA-0. In Franklin [Fra04], pages 290–305.
Available from: http://dx.doi.org/10.1007/978-3-540-28628-8_18.
One citation on page 104.

[BC11] Daniel J. Bernstein and Sanjit Chatterjee, editors. *Progress in Cryptology - INDOCRYPT 2011 - 12th International Conference on Cryptology in India, Chennai, India, December 11-14, 2011. Proceedings*, volume 7107 of *Lecture Notes in Computer Science*. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-25578-6.
2 citations on pages 246 and 247.

[BCD+98] Carolynn Burwick, Don Coppersmith, Edward D'Avignon, Rosario Gennaro, Shai Halevi, Charanjit Jutla Stephen M. Matyas, Jr., Mohammad Peyravian, Luke O'Connor, David Safford, and Nevenko Zunic. MARS – a candidate cipher for AES. In *Proceedings of the First AES Candidate Conference*, pages 337–342. National Institute of Standard and Technology, 1998.
Available from: http://csrc.nist.gov/archive/aes/index.html.
One citation on page 167.

[BCD03a] Alex Biryukov, Christophe De Cannière, and Gustaf Dellkrantz. Cryptanalysis of SAFER++. In Boneh [Bon03], pages 195–211.
Available from: http://dx.doi.org/10.1007/978-3-540-45146-4_12.
2 citations on pages 153 and 154.

[BCD03b] Alex Biryukov, Christophe De Cannière, and Gustaf Dellkrantz. Crytanalysis of SAFER++. *IACR Cryptology ePrint Archive*, 2003:109, 2003.
Available from: http://eprint.iacr.org/2003/109.
2 citations on pages 153 and 154.

[BCG+12a] Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knezevic, 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 Wang and Sako [WS12], pages 208–225.
Available from: http://dx.doi.org/10.1007/978-3-642-34961-4_14.
One citation on page 215.

[BCG+12b] Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knezevic, 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 (Full version). *IACR Cryptology ePrint Archive*, 2012:529, 2012.
Available from: http://eprint.iacr.org/2012/529.
3 citations on pages 234, 236, and 237.

[BCN+10] Gregory V. Bard, Nicolas Courtois, Jorge Nakahara, Pouyan Sepehrdad, and Bingsheng Zhang. Algebraic, AIDA/Cube and Side Channel Analysis of KATAN Family of Block Ciphers. In Gong and Gupta [GG10], pages 176–196.
Available from: http://dx.doi.org/10.1007/978-3-642-17401-8_14,
DOI: doi:10.1007/978-3-642-17401-8.
One citation on page 82.

[BCQ04] Alex Biryukov, Christophe De Cannière, and Michaël Quisquater. On Multiple Linear Approximations. In Franklin [Fra04], pages 1–22.
Available from: http://dx.doi.org/10.1007/978-3-540-28628-8_1.
One citation on page 91.

[BD10] Eli Biham and Orr Dunkelman. The SHAvite-3 Hash Function, 2009-2010.
Available from: http://www.cs.technion.ac.il/~orrd/SHAvite-3/.
One citation on page 55.

[BDD+12] Charles Bouillaguet, Patrick Derbez, Orr Dunkelman, Pierre-Alain Fouque, Nathan Keller, and Vincent Rijmen. Low-Data Complexity Attacks on AES. *IEEE Trans. Information Theory*, 58(11):7002–7017, 2012.
Available from: https://doi.org/10.1109/TIT.2012.2207880,
DOI: doi:10.1109/TIT.2012.2207880.
One citation on page 186.

[BDF11] Charles Bouillaguet, Patrick Derbez, and Pierre-Alain Fouque. Automatic Search of Attacks on Round-Reduced AES and Applications. In Rogaway [Rog11], pages 169–187.
Available from: https://doi.org/10.1007/978-3-642-22792-9_10,
DOI: doi:10.1007/978-3-642-22792-9_10.
One citation on page 186.

[BDFL11] Charles Bouillaguet, Orr Dunkelman, Pierre-Alain Fouque, and Gaëtan Leurent. New Insights on Impossible Differential Cryptanalysis. In Miri and Vaudenay [MV12a], pages 243–259.
Available from: http://dx.doi.org/10.1007/978-3-642-28496-0_15.
One citation on page 89.

[BDK01a] Eli Biham, Orr Dunkelman, and Nathan Keller. Linear Cryptanalysis of Reduced Round Serpent. In Matsui [Mat02], pages 16–27.
Available from: http://dx.doi.org/10.1007/3-540-45473-X_2.
One citation on page 169.

[BDK01b] Eli Biham, Orr Dunkelman, and Nathan Keller. The Rectangle Attack - Rectangling the Serpent. *IACR Cryptology ePrint Archive*, 2001:21, 2001.
Available from: http://eprint.iacr.org/2001/021.
One citation on page 84.

[BDK01c] Eli Biham, Orr Dunkelman, and Nathan Keller. The Rectangle Attack - Rectangling the Serpent. In Pfitzmann [Pfi01], pages 340–357.
Available from: http://dx.doi.org/10.1007/3-540-44987-6_21.
One citation on page 84.

[BDK02a] Eli Biham, Orr Dunkelman, and Nathan Keller. Enhancing Differential-Linear Cryptanalysis. In Zheng [Zhe02], pages 254–266.
Available from: http://dx.doi.org/10.1007/3-540-36178-2_16,
DOI: doi:10.1007/3-540-36178-2.
One citation on page 94.

[BDK02b] Eli Biham, Orr Dunkelman, and Nathan Keller. New Results on Boomerang and Rectangle Attacks. In Daemen and Rijmen [DR02c], pages 1–16.
Available from: http://dx.doi.org/10.1007/3-540-45661-9_1.
One citation on page 169.

[BDK03] Eli Biham, Orr Dunkelman, and Nathan Keller. Differential-Linear Cryptanalysis of Serpent. In Johansson [Joh03], pages 9–21.
Available from: http://dx.doi.org/10.1007/978-3-540-39887-5_2.
One citation on page 169.

[BDK05] Eli Biham, Orr Dunkelman, and Nathan Keller. Related-Key Boomerang and Rectangle Attacks. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 507–525. Springer, 2005.
Available from: https://doi.org/10.1007/11426639_30,
DOI: doi:10.1007/11426639_30.
One citation on page 185.

[BDK07a] Eli Biham, Orr Dunkelman, and Nathan Keller. A New Attack on 6-Round IDEA. In Biryukov [Bir07], pages 211–224.
Available from: http://dx.doi.org/10.1007/978-3-540-74619-5_14,
DOI: doi:10.1007/978-3-540-74619-5.
One citation on page 144.

[BDK07b] Eli Biham, Orr Dunkelman, and Nathan Keller. A Simple Related-Key Attack on the Full SHACAL-1. In Masayuki Abe, editor, *CT-RSA*, volume 4377 of *Lecture Notes in Computer Science*, pages 20–30. Springer, 2007.
Available from: http://dx.doi.org/10.1007/11967668_2.
One citation on page 187.

[BDK+10] Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, and Adi Shamir. Key Recovery Attacks of Practical Complexity on AES-256 Variants with up to 10 Rounds. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 299–319. Springer, 2010.
Available from: https://doi.org/10.1007/978-3-642-13190-5_15,
DOI: doi:10.1007/978-3-642-13190-5_15.
One citation on page 185.

[BDKS11] Eli Biham, Orr Dunkelman, Nathan Keller, and Adi Shamir. New Data-Efficient Attacks on Reduced-Round IDEA. *IACR Cryptology ePrint Archive*, 2011:417, 2011.
Available from: http://eprint.iacr.org/2011/417.
One citation on page 144.

[BDLF10] Charles Bouillaguet, Orr Dunkelman, Gaëtan Leurent, and Pierre-Alain Fouque. Another Look at Complementation Properties. In Hong and Iwata [HI10], pages 347–364.
Available from: http://dx.doi.org/10.1007/978-3-642-13858-4_20.
One citation on page 161.

[BDMW10] Kim A. Browning, John F. Dillon, Michael T. McQuistan, and Adam J. Wolfe. An APN permutation in dimension six. McGuire, Gary (ed.) et al., Finite fields. Theory and applications. Proceedings of the 9th international conference on finite fields and applications, Dublin, Ireland, July 13–17, 2009. Providence, RI: American Mathematical Society (AMS). Contemporary Mathematics 518, 33-42 (2010)., 2010.
One citation on page 61.

[BDN+10] Stéphane Badel, Nilay Dagtekin, Jorge Nakahara, Khaled Ouafi, Nicolas Reffé, Pouyan Sepehrdad, Petr Susil, and Serge Vaudenay. ARMADILLO: A Multi-purpose Cryptographic Primitive Dedicated to Hardware. In Mangard and Standaert [MS10], pages 398–412.
Available from: http://dx.doi.org/10.1007/978-3-642-15031-9_27.
2 citations on pages 234 and 237.

[Bel11] Steven M. Bellovin. Frank Miller: Inventor of the One-Time Pad. *Cryptologia*, 35(3):203–222, 2011.
Available from: http://dx.doi.org/10.1080/01611194.2011.583711.
One citation on page 24.

[Beu03] Jean-Luc Beuchat. FPGA Implementations of the RC6 Block Cipher. In Peter Y. K. Cheung, George A. Constantinides, and José T. de Sousa, editors, *FPL 2003*, volume 2778 of *Lecture Notes in Computer Science*, pages 101–110. Springer, 2003.
Available from: http://dx.doi.org/10.1007/978-3-540-45234-8_11.
One citation on page 237.

[BF00a] Steve Babbage and Laurent Frisch. On MISTY1 Higher Order Differential Cryptanalysis. In Dongho Won, editor, *ICISC*, volume 2015 of *Lecture Notes in Computer Science*, pages 22–36. Springer, 2000.
Available from: http://dx.doi.org/10.1007/3-540-45247-8_3.
One citation on page 175.

[BF00b] Eli Biham and Vladimir Furman. Impossible Differential on 8-Round MARS' Core. In *Proceedings of the Third AES Candidate Conference*, pages 186–194, 2000.
One citation on page 168.

[BF06a] Thomas Baignères and Matthieu Finiasz. Dial C for Cipher – Le chiffrement était presque parfait. In Eli Biham and Amr M. Youssef, editors, *Selected Areas in Cryptography 2006*, volume 4356 of *Lecture Notes in Computer Science*, pages 76–95. Springer, 2006.
Available from: http://dx.doi.org/10.1007/978-3-540-74462-7_7.
3 citations on pages 40, 65, and 66.

[BF06b] Thomas Baignères and Matthieu Finiasz. KFC - The Krazy Feistel Cipher. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 380–395. Springer, 2006.
Available from: http://dx.doi.org/10.1007/11935230_25.
2 citations on pages 40 and 65.

[BG10] Celine Blondeau and Benoit Gérard. Links between theoretical and effective differential probabilities: Experiments on PRESENT. ECrypt II Workshop on Tools for Cryptanalysis, 2010.
Available from: http://perso.uclouvain.be/fstandae/tools_2010_proceedings.pdf.
2 citations on pages 44 and 86.

[BG11a] Céline Blondeau and Benoît Gérard. Multiple Differential Cryptanalysis: Theory and Practice. In Joux [Jou11], pages 35–54.
Available from: http://dx.doi.org/10.1007/978-3-642-21702-9_3.
2 citations on pages 80 and 205.

[BG11b] Céline Blondeau and Benoît Gérard. Multiple Differential Cryptanalysis: Theory and Practice (Corrected). *IACR Cryptology ePrint Archive*, 2011:115, 2011.
Available from: http://eprint.iacr.org/2011/115.
2 citations on pages 80 and 205.

[BGLP13] Ryad Benadjila, Jian Guo, Victor Lomné, and Thomas Peyrin. Implementing Lightweight Block Ciphers on x86 Architectures. In Lange et al. [LLL13], page 25 pages.
2 citations on pages 241 and 243.

[BGN12a] Céline Blondeau, Benoît Gérard, and Kaisa Nyberg. Multiple Differential Cryptanalysis using `LLR` and χ^2 Statistics. In Ivan Visconti and Roberto De Prisco, editors, *SCN 2012*, volume 7485 of *Lecture Notes in Computer Science*, pages 343–360. Springer, 2012. IACR preprint: [BGN12b].
Available from: http://dx.doi.org/10.1007/978-3-642-32928-9_19.
4 citations on pages 79, 87, 88, and 252.

[BGN12b] Céline Blondeau, Benoît Gérard, and Kaisa Nyberg. Multiple Differential Cryptanalysis using `LLR` and χ^2 Statistics. *IACR Cryptology ePrint Archive*, 2012:360, 2012. Final version: [BGN12a].
Available from: http://eprint.iacr.org/2012/360.
One citation on page 252.

[BGS11] Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors. *Selected Areas in Cryptography - 17th International Workshop, SAC 2010, Waterloo, Ontario, Canada, August 12-13, 2010, Revised Selected Papers*, volume 6544 of *Lecture Notes in Computer Science*. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-19574-7.
2 citations on pages 255 and 291.

[BGW+13] Andrey Bogdanov, Huizheng Geng, Meiqin Wang, Long Wen, and Baudoin Collard. Zero-Correlation Linear Cryptanalysis with FFT and Improved Attacks on ISO Standards Camellia and CLEFIA. In Lange et al. [LLL13], page 18 pages.
2 citations on pages 91 and 203.

[Bih93] Eli Biham. New Types of Cryptoanalytic Attacks Using related Keys (Extended Abstract). In Helleseth [Hel94], pages 398–409.
Available from: http://dx.doi.org/10.1007/3-540-48285-7_34.
One citation on page 106.

[Bih94a] Eli Biham. Cryptanalysis of Multiple Modes of Operation. In Pieprzyk and Safavi-Naini [PSN95], pages 278–292.
Available from: http://dx.doi.org/10.1007/BFb0000441.
One citation on page 132.

[Bih94b] Eli Biham. New Types of Cryptanalytic Attacks Using Related Keys. *J. Cryptology*, 7(4):229–246, 1994.
Available from: http://dx.doi.org/10.1007/BF00203965.
One citation on page 106.

[Bih97a] Eli Biham. Cryptanalysis of Ladder-DES. In *FSE 1997* [Bih97c], pages 134–138.
Available from: http://dx.doi.org/10.1007/BFb0052341.
One citation on page 177.

[Bih97b] Eli Biham. A Fast New DES Implementation in Software. In *FSE* [Bih97c], pages 260–272.
Available from: http://dx.doi.org/10.1007/BFb0052352.
2 citations on pages 131 and 195.

[Bih97c] Eli Biham, editor. *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*, volume 1267 of *Lecture Notes in Computer Science*. Springer, 1997.
5 citations on pages 252, 253, 262, 271, and 280.

[Bih03] Eli Biham, editor. *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*. Springer, 2003.
2 citations on pages 258 and 272.

[Bir03] Alex Biryukov. Analysis of Involutional Ciphers: Khazad and Anubis. In Johansson [Joh03], pages 45–53.
Available from: http://dx.doi.org/10.1007/978-3-540-39887-5_5.
One citation on page 190.

[Bir07] Alex Biryukov, editor. *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers*, volume 4593 of *Lecture Notes in Computer Science*. Springer, 2007.
Available from: https://doi.org/10.1007/978-3-540-74619-5,
DOI: doi:10.1007/978-3-540-74619-5.
6 citations on pages 250, 267, 273, 278, 279, and 292.

[BK98a] Alex Biryukov and Eyal Kushilevitz. From Differential Cryptoanalysis to Ciphertext-Only Attacks. In Hugo Krawczyk, editor, *CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 72–88. Springer, 1998.
Available from: http://dx.doi.org/10.1007/BFb0055721.
2 citations on pages 94 and 134.

[BK98b] Alex Biryukov and Eyal Kushilevitz. Improved Cryptanalysis of RC5. In Nyberg [Nyb98], pages 85–99.
Available from: http://dx.doi.org/10.1007/BFb0054119.
One citation on page 158.

[BK09] Alex Biryukov and Dmitry Khovratovich. Related-Key Cryptanalysis of the Full AES-192 and AES-256. In Matsui [Mat09], pages 1–18.
Available from: http://dx.doi.org/10.1007/978-3-642-10366-7_1.
3 citations on pages 103, 183, and 185.

[BK10] Alex Biryukov and Dmitry Khovratovich. Feasible Attack on the 13-round AES-256. *IACR Cryptology ePrint Archive*, 2010:257, 2010.
Available from: http://eprint.iacr.org/2010/257.
One citation on page 185.

[BKL+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 Pascal Paillier and Ingrid Verbauwhede, editors, *CHES 2007*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.
Available from: http://dx.doi.org/10.1007/978-3-540-74735-2_31.
2 citations on pages 204 and 236.

[BKN09] Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolić. Distinguisher and Related-Key Attack on the Full AES-256. In Halevi [Hal09], pages 231–249.
Available from: http://dx.doi.org/10.1007/978-3-642-03356-8_14.
2 citations on pages 183 and 185.

[BKR97] Johan Borst, Lars R. Knudsen, and Vincent Rijmen. Two Attacks on Reduced IDEA. In Walter Fumy, editor, *EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 1997.
Available from: http://dx.doi.org/10.1007/3-540-69053-0_1.
2 citations on pages 84 and 94.

[BKR11a] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique Cryptanalysis of the Full AES. *IACR Cryptology ePrint Archive*, 2011:449, 2011.
Available from: http://eprint.iacr.org/2011/449.
4 citations on pages 96, 182, 184, and 185.

[BKR11b] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique Cryptanalysis of the Full AES. In Lurl and Wang [LW11], pages 344–371.
Available from: http://dx.doi.org/10.1007/978-3-642-25385-0_19.
2 citations on pages 96 and 182.

[BKS10] Yongjin Yeom Bonwook Koo and Junghwan Song. Related-Key Boomerang Attack on Block Cipher SQUARE. Cryptology ePrint Archive, Report 2010/073, 2010.
Available from: http://eprint.iacr.org/2010/073.
One citation on page 159.

[BLN14] Céline Blondeau, Gregor Leander, and Kaisa Nyberg. Differential-Linear Cryptanalysis Revisited. Preproceedings of FSE 2014, March 2014.
One citation on page 94.

[BLNW12] Andrey Bogdanov, Gregor Leander, Kaisa Nyberg, and Meiqin Wang. Integral and Multidimensional Linear Distinguishers with Correlation Zero. In Wang and Sako [WS12], pages 244–261.
Available from: http://dx.doi.org/10.1007/978-3-642-34961-4_16.
3 citations on pages 91, 92, and 147.

[Blo13] Cèline Blondeau. Improbable Differential from Impossible Differential: On the Validity of the Model. In Goutam Paul and Serge Vaudenay, editors, *Indocrypt 2013*, volume 8250 of *Lecture Notes in Computer Science*, page 18 pages. Springer, 2013.
Available from: https://doi.org/10.1007/978-3-319-03515-4,
DOI: doi:10.1007/978-3-319-03515-4.
2 citations on pages 85 and 203.

[BM95] James Bamford and Wayne Madsen. *The Puzzle Palace, Second Edition*. Information security and cryptography. Penguin Books, 1995.
One citation on page 151.

[BN13] Céline Blondeau and Kaisa Nyberg. New Links between Differential and Linear Cryptanalysis. In Johansson and Nguyen [JN13], pages 388–404.
Available from: http://dx.doi.org/10.1007/978-3-642-38348-9_24,
DOI: doi:10.1007/978-3-642-38348-9.
One citation on page 205.

[BNPV02] Alex Biryukov, Jorge Nakahara, Jr., Bart Preneel, and Joos Vandewalle. New Weak-Key Classes of IDEA. In Robert H. Deng, Sihan Qing, Feng Bao, and Jianying Zhou, editors, *ICICS 2002*, volume 2513 of *Lecture Notes in Computer Science*, pages 315–326. Springer, 2002.
Available from: http://dx.doi.org/10.1007/3-540-36159-6_27.
One citation on page 144.

[Bog07a] Andrey Bogdanov. Cryptanalysis of the KeeLoq block cipher. *IACR Cryptology ePrint Archive*, 2007:55, 2007.
Available from: http://eprint.iacr.org/2007/055.
One citation on page 137.

[Bog07b] Andrey Bogdanov. Linear Slide Attacks on the KeeLoq Block Cipher. In Dingyi Pei, Moti Yung, Dongdai Lin, and Chuankun Wu, editors, *Inscrypt*, volume 4990 of *Lecture Notes in Computer Science*, pages 66–80. Springer, 2007.
Available from: http://dx.doi.org/10.1007/978-3-540-79499-8_7.
One citation on page 137.

[Bog11] Andrey Bogdanov. On unbalanced Feistel networks with contracting MDS diffusion. *Designs, Codes and Cryptography*, 59(1-3):35–58, 2011.
Available from: http://dx.doi.org/10.1007/s10623-010-9462-0.
One citation on page 55.

[Bon03] Dan Boneh, editor. *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*. Springer, 2003.
3 citations on pages 249, 259, and 271.

[Bor99] Johan Borst. Weak keys of CRYPTON. In *Proceedings of the Second AES Candidate Conference*, March 1999.
Available from: http://csrc.nist.gov/archive/aes/round1/comments/980828-jborst.pdf.
One citation on page 197.

[Bor00] Johan Borst. The Block Cipher: Gran Cru, October 2000.
Available from: http://www.cosic.esat.kuleuven.be/nessie/workshop/submissions.html.
One citation on page 187.

[BP98] Lawrie Brown and Josef Pieprzyk. Introducing the new LOKI97 Block Cipher, June 1998.
Available from: http://seit.unsw.adfa.edu.au/staff/sites/lpb/research/loki97/.
One citation on page 178.

[BPW06] Johannes Buchmann, Andrei Pyshkin, and Ralf-Philipp Weinmann. Block Ciphers Sensitive to Gröbner Basis Attacks. In David Pointcheval, editor, *CT-RSA*, volume 3860 of *Lecture Notes in Computer Science*, pages 313–331. Springer, 2006.
Available from: http://dx.doi.org/10.1007/11605805_20.
One citation on page 122.

[BR02] Paulo S.L.M. Barreto and Vincent Rijmen. The KHAZAD Legacy-Level Block Cipher. In *First NESSIE Workshop*. NESSIE, November 2002.
Available from: http://www.larc.usp.br/~pbarreto/KhazadPage.html.
One citation on page 187.

[BR10] Andrey Bogdanov and Christian Rechberger. A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN. In Biryukov et al. [BGS11], pages 229–240.
Available from: http://dx.doi.org/10.1007/978-3-642-19574-7_16.
3 citations on pages 95, 98, and 208.

[BR11a] Paulo S. L. M. Barreto and Vincent Rijmen. Whirlpool. In Henk C. A. van Tilborg and Sushil Jajodia, editors, *Encyclopedia of Cryptography and Security (2nd Ed.)*, pages 1384–1385. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-1-4419-5906-5_626.
2 citations on pages 46 and 67.

[BR11b] Andrey Bogdanov and Vincent Rijmen. Linear Hulls with Correlation Zero and Linear Cryptanalysis of Block Ciphers. *IACR Cryptology ePrint Archive*, 2011:123, 2011.
Available from: http://eprint.iacr.org/2011/123.
2 citations on pages 91 and 92.

[Bra90] Gilles Brassard, editor. *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*. Springer, 1990.
2 citations on pages 247 and 298.

[BRV14] Alex Biryukov, Arnab Roy, and Vesselin Velichkov. Differential Analysis of Block Ciphers SIMON and SPECK. Preproceedings of FSE 2014, March 2014.
2 citations on pages 223 and 225.

[BS90] Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In Menezes and Vanstone [MV91], pages 2–21.
Available from: http://dx.doi.org/10.1007/3-540-38424-3_1.
3 citations on pages 71, 79, and 133.

[BS91a] Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991.
Available from: http://dx.doi.org/10.1007/BF00630563.
4 citations on pages 71, 79, 80, and 133.

[BS91b] Eli Biham and Adi Shamir. Differential Cryptanalysis of Snefru, Khafre, REDOC-II, LOKI and Lucifer. In Feigenbaum [Fei92], pages 156–171.
Available from: http://dx.doi.org/10.1007/3-540-46766-1_11.
One citation on page 127.

[BS91c] Eli Biham and Adi Shamir. Differential Cryptoanalysis of Feal and N-Hash. In Davies [Dav91], pages 1–16.
Available from: http://dx.doi.org/10.1007/3-540-46416-6_1.
2 citations on pages 71 and 137.

[BS92] Eli Biham and Adi Shamir. Differential Cryptanalysis of the Full 16-Round DES. In Ernest F. Brickell, editor, *CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 487–496. Springer, 1992.
Available from: http://dx.doi.org/10.1007/3-540-48071-4_34.
2 citations on pages 71 and 74.

[BS93] Eli Biham and Adi Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag, London, UK, 1993.
One citation on page 40.

[BS94] Matt Blaze and Bruce Schneier. The MacGuffin Block Cipher Algorithm. In Preneel [Pre95], pages 97–110.
Available from: http://dx.doi.org/10.1007/3-540-60590-8_8.
One citation on page 32.

[BS11] Andrey Bogdanov and Kyoji Shibutani. Analysis of 3-line generalized Feistel networks with double SD-functions. *Inf. Process. Lett.*, 111(13):656–660, 2011.
Available from: http://dx.doi.org/10.1016/j.ipl.2011.04.002.
One citation on page 55.

[BS13a] Andrey Bogdanov and Kyoji Shibutani. Generalized Feistel networks revisited. *Designs, Codes and Cryptography*, 66(1-3):75–97, 2013.
Available from: http://dx.doi.org/10.1007/s10623-012-9660-z.
One citation on page 55.

[BS13b] Andrey Bogdanov and Kyoji Shibutani. Towards the optimality of Feistel ciphers with substitution-permutation functions. In *The International Workshop on Coding and Cryptography WCC 2013, April 15–19, 2013, Bergen (Norway), Preproceedings*, pages 112–123, 2013.
Available from: http://www.selmer.uib.no/WCC2013/PreProceedings.pdf.
2 citations on pages 55 and 227.

[BSS+13] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK Families of Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2013/404, 2013.
Available from: http://eprint.iacr.org/2013/404.
8 citations on pages 220, 222, 233, 235, 236, 237, 238, and 239.

[BT13] Andrey Bogdanov and Elmar Tischhauser. On the Wrong Key Randomisation and Key Equivalence Hypotheses in Matsui's Algorithm 2. In *Fast Software Encryption 2013*. Springer-Verlag, 2013.
Available from: https://lirias.kuleuven.be/handle/123456789/333158.
2 citations on pages 79 and 90.

[BV05] Thomas Baignères and Serge Vaudenay. Proving the Security of AES Substitution-Permutation Network. In Bart Preneel and Stafford E. Tavares, editors, *Selected Areas in Cryptography 2005*, volume 3897 of *Lecture Notes in Computer Science*, pages 65–81. Springer, 2005.
Available from: http://dx.doi.org/10.1007/11693383_5.
One citation on page 66.

[BW99] Alex Biryukov and David Wagner. Slide Attacks. In Knudsen [Knu99], pages 245–259.
Available from: http://dx.doi.org/10.1007/3-540-48519-8_18.
2 citations on pages 107 and 112.

[BW00] Alex Biryukov and David Wagner. Advanced Slide Attacks. In Preneel [Pre00], pages 589–606.
Available from: http://dx.doi.org/10.1007/3-540-45539-6_41.
6 citations on pages 107, 110, 111, 112, 114, and 132.

[BW12] Andrey Bogdanov and Meiqin Wang. Zero Correlation Linear Cryptanalysis with Reduced Data Complexity. In Canteaut [Can12], pages 29–48.
Available from: http://dx.doi.org/10.1007/978-3-642-34047-5_3.
3 citations on pages 91, 92, and 160.

[Can12] Anne Canteaut, editor. *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-34047-5.
5 citations on pages 257, 261, 276, 280, and 290.

[Car10a] Claude Carlet. Boolean Functions for Cryptography and Error-Correcting Codes. In Crama and Hammer [CH10], pages 257–397.
Available from: http://www.math.univ-paris13.fr/~carlet/chap-fcts-Bool-corr.pdf.
3 citations on pages 56, 59, and 62.

[Car10b] Claude Carlet. Vectorial Boolean Functions for Cryptography. In Crama and Hammer [CH10], pages 398–472.
Available from: http://www.math.univ-paris13.fr/~carlet/chap-vectorial-fcts-corr.pdf.
4 citations on pages 56, 57, 59, and 61.

[CB06] Nicolas Courtois and Gregory V. Bard. Algebraic Cryptanalysis of the Data Encryption Standard. *IACR Cryptology ePrint Archive*, 2006:402, 2006.
Available from: http://eprint.iacr.org/2006/402.
One citation on page 121.

[CB07] Nicolas Courtois and Gregory V. Bard. Algebraic Cryptanalysis of the Data Encryption Standard. In Steven D. Galbraith, editor, *IMA Int. Conf.*, volume 4887 of *Lecture Notes in Computer Science*, pages 152–169. Springer, 2007.
Available from: http://dx.doi.org/10.1007/978-3-540-77272-9_10.
One citation on page 121.

[CBW08] Nicolas Courtois, Gregory V. Bard, and David Wagner. Algebraic and Slide Attacks on KeeLoq. In Nyberg [Nyb08], pages 97–115.
Available from: http://dx.doi.org/10.1007/978-3-540-71039-4_6.
2 citations on pages 122 and 137.

[CCZ98] Claude Carlet, Pascale Charpin, and Victor Zinoviev. Codes, Bent Functions and Permutations Suitable For DES-like Cryptosystems. *Des. Codes Cryptography*, 15(2):125–156, 1998.
Available from: http://dx.doi.org/10.1023/A:1008344232130.
One citation on page 60.

[CD98] Matt Curtin and Justin Dolske. A Brute Force Search of DES Keyspace. `;login:`, 23(3), May 1998.
Available from: http://www.interhack.net/pubs/des-key-crack/des-key-crack.ps.
One citation on page 130.

[CDK09] Christophe De Cannière, Orr Dunkelman, and Miroslav Knezevic. KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers. In Christophe Clavier and Kris Gaj, editors, *CHES 2009*, volume 5747 of *Lecture Notes in Computer Science*, pages 272–288. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-04138-9_20.
2 citations on pages 207 and 235.

[CE85] David Chaum and Jan-Hendrik Evertse. Crytanalysis of DES with a Reduced Number of Rounds: Sequences of Linear Factors in Block Ciphers. In Williams [Wil86], pages 192–211.
Available from: http://dx.doi.org/10.1007/3-540-39799-X_16.
One citation on page 95.

[CFG+14] Anne Canteaut, Thomas Fuhr, Henri Gilbert, María Naya-Plasencia, and Jean-René Reinhard. Multiple differential cryptanalysis of round-reduced PRINCE. Preproceedings of FSE 2014, March 2014.
Available from: http://eprint.iacr.org/2014/089.pdf.
One citation on page 219.

[CGP+12] Claude Carlet, Louis Goubin, Emmanuel Prouff, Michaël Quisquater, and Matthieu Rivain. Higher-Order Masking Schemes for S-Boxes. In Canteaut [Can12], pages 366–384.
Available from: http://dx.doi.org/10.1007/978-3-642-34047-5_21.
One citation on page 228.

[CH10] Yves Crama and Peter L. Hammer, editors. *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, volume 134 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, June 2010.
Available from: http://www.cambridge.org/.
2 citations on pages 56 and 257.

[CHN08] Joo Yeon Cho, Miia Hermelin, and Kaisa Nyberg. A New Technique for Multidimensional Linear Cryptanalysis with Applications on Reduced Round Serpent. In Pil Joong Lurl and Jung Hurl Cheon, editors, *ICISC*, volume 5461 of *Lecture Notes in Computer Science*, pages 383–398. Springer, 2008.
Available from: http://dx.doi.org/10.1007/978-3-642-00730-9_24.
One citation on page 91.

[Cho09] Joo Yeon Cho. Linear Cryptanalysis of Reduced-Round PRESENT. *IACR Cryptology ePrint Archive*, 2009:397, 2009.
Available from: http://eprint.iacr.org/2009/397.
One citation on page 205.

[Cho10] Joo Yeon Cho. Linear Cryptanalysis of Reduced-Round PRESENT. In Pieprzyk [Pie10], pages 302–317.
Available from: http://dx.doi.org/10.1007/978-3-642-11925-5_21.
One citation on page 205.

[CKM97] Stéphane Collart, Michael Kalkbrener, and Daniel Mall. Converting Bases with the Gröbner Walk. *J. Symb. Comput.*, 24(3/4):465–469, 1997.
Available from: http://dx.doi.org/10.1006/jsco.1996.0145.
One citation on page 122.

[ÇKöB12] Mustafa Çoban, Ferhat Karakoç, and özkan Boztas. Biclique Cryptanalysis of TWINE. In Josef Pieprzyk, Ahmad-Reza Sadeghi, and Mark Manulis, editors, *CANS 2012*, volume 7712 of *Lecture Notes in Computer Science*, pages 43–55. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-35404-5_5.
One citation on page 215.

[CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In Preneel [Pre00], pages 392–407.
Available from: http://dx.doi.org/10.1007/3-540-45539-6_27.
One citation on page 121.

[CL05] Carlos Cid and Gaëtan Leurent. An Analysis of the XSL Algorithm. In Roy [Roy05], pages 333–352.
Available from: https://doi.org/10.1007/11593447_18,
DOI: doi:10.1007/11593447_18.
One citation on page 183.

[CLLL00] Dong Hyeon Cheon, Sangjin Lurl, Jong In Lim, and Sung Jae Lee. New Block Cipher DONUT Using Pairwise Perfect Decorrelation. In Bimal K. Roy and Eiji Okamoto, editors, *INDOCRYPT 2000*, volume 1977 of *Lecture Notes in Computer Science*, pages 262–270. Springer, 2000.
Available from: http://dx.doi.org/10.1007/3-540-44495-5_23.
One citation on page 65.

[CM03] Nicolas Courtois and Willi Meier. Algebraic Attacks on Stream Ciphers with Linear Feedback. In Biham [Bih03], pages 345–359.
Available from: http://dx.doi.org/10.1007/3-540-39200-9_21.
2 citations on pages 58 and 121.

[CMM13] Mickaël Cazorla, Kevin Marquet, and Marine Minier. Survey and Benchmark of Lightweight Block Ciphers for Wireless Sensor Networks. *IACR Cryptology ePrint Archive*, 2013:295, 2013.
Available from: http://eprint.iacr.org/2013/295.
4 citations on pages 200, 238, 239, and 240.

[CNPV13a] Anne Canteaut, María Naya-Plasencia, and Bastien Vayssière. Sieve-in-the-Middle: Improved MITM Attacks. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013 (1)*, volume 8042 of *Lecture Notes in Computer Science*, pages 222–240. Springer, 2013.
Available from: http://dx.doi.org/10.1007/978-3-642-40041-4_13.
3 citations on pages 98, 99, and 219.

[CNPV13b] Anne Canteaut, María Naya-Plasencia, and Bastien Vayssière. Sieve-in-the-Middle: Improved MITM Attacks (Full Version). Cryptology ePrint Archive, Report 2013/324, 2013.
Available from: http://eprint.iacr.org/2013/324.
One citation on page 219.

[Coh87] Fred Cohen. *Introductory Information Protection*. Self-published, 1987.
Available from: http://all.net/books/IP/index.html.
One citation on page 21.

[Cop85] Don Coppersmith. The Real Reason for Rivest's Phenomenon. In Williams [Wil86], pages 535–536.
Available from: http://dx.doi.org/10.1007/3-540-39799-X_42.
One citation on page 146.

[Cop94] Don Coppersmith. The Data Encryption Standard (DES) and its strength against attacks. *IBM Journal of Research and Development*, 38(3):243–250, 1994.
Available from: http://dx.doi.org/10.1147/rd.383.0243.
One citation on page 71.

[Cou03] Nicolas Courtois. Fast Algebraic Attacks on Stream Ciphers with Linear Feedback. In Boneh [Bon03], pages 176–194.
Available from: http://dx.doi.org/10.1007/978-3-540-45146-4_11.
One citation on page 121.

[Cou12a] Nicolas T. Courtois. An Improved Differential Attack on Full GOST. *IACR Cryptology ePrint Archive*, 2012:138, 2012.
Available from: http://eprint.iacr.org/2012/138.
One citation on page 140.

[Cou12b] Nicolas T. Courtois. Security Evaluation of GOST 28147-89 in View of International Standardisation. *Cryptologia*, 36(1):2–13, 2012.
Available from: http://dx.doi.org/10.1080/01611194.2011.632807.
One citation on page 140.

[Cou13] Nicolas T. Courtois. Low-Complexity Key Recovery Attacks on GOST Block Cipher. *Cryptologia*, 37(1):1–10, 2013.
Available from: http://dx.doi.org/10.1080/01611194.2012.739587.
One citation on page 140.

[CP88] David Chaum and Wyn L. Price, editors. *Advances in Cryptology - EUROCRYPT '87, Workshop on the Theory and Application of of Cryptographic Techniques, Amsterdam, The Netherlands, April 13-15, 1987, Proceedings*, volume 304 of *Lecture Notes in Computer Science*. Springer, 1988.
2 citations on pages 264 and 291.

[CP02a] Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In Zheng [Zhe02], pages 267–287.
Available from: http://dx.doi.org/10.1007/3-540-36178-2_17,
DOI: doi:10.1007/3-540-36178-2.
2 citations on pages 121 and 183.

[CP02b] Nicolas T. Courtois and Josef Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. *IACR Cryptology ePrint Archive*, 2001:044, 2002.
Available from: http://eprint.iacr.org/2002/038.
One citation on page 140.

[CP05] Christophe De Cannière and Bart Preneel. Trivium Specifications, eSTREAM submission. stream project submissions, eStream, 2005.
Available from: http://www.ecrypt.eu.org/stream/triviump3.html.
One citation on page 207.

[CP08] Christophe De Cannière and Bart Preneel. Trivium. In Matthew J. B. Robshaw and Olivier Billet, editors, *The eSTREAM Finalists*, volume 4986 of *Lecture Notes in Computer Science*, pages 244–266. Springer, 2008.
Available from: http://dx.doi.org/10.1007/978-3-540-68351-3_18.
One citation on page 207.

[CRD08] Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors. *Progress in Cryptology - INDOCRYPT 2008, 9th International Conference on Cryptology in India, Kharagpur, India, December 14-17, 2008. Proceedings*, volume 5365 of *Lecture Notes in Computer Science*. Springer, 2008.
Available from: https://doi.org/10.1007/978-3-540-89754-5,
DOI: doi:10.1007/978-3-540-89754-5.
2 citations on pages 272 and 277.

[CRY01] CRYPTREC. Detailed Evaluation Report on the Security of RC2. CRYPTREC Technical Report No. 0036, 2001.
Available from: http://www.cryptrec.go.jp/estimation/rep_ID0036.pdf.
One citation on page 141.

[CRY03] CRYPTREC. Analysis of RC2. CRYPTREC Technical Report No. 1042, March 2003.
Available from: https://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1042_rc2.pdf.
One citation on page 141.

[CS09] Baudoin Collard and François-Xavier Standaert. A Statistical Saturation Attack against the Block Cipher PRESENT. In Marc Fischlin, editor, *CT-RSA 2009*, volume 5473 of *Lecture Notes in Computer Science*, pages 195–210. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-00862-7_13.
One citation on page 205.

[CSQ08] Baudoin Collard, François-Xavier Standaert, and Jean-Jacques Quisquater. Experiments on the Multiple Linear Cryptanalysis of Reduced Round Serpent. In Nyberg [Nyb08], pages 382–397.
Available from: http://dx.doi.org/10.1007/978-3-540-71039-4_24.
One citation on page 91.

[CV94] Florent Chabaud and Serge Vaudenay. Links Between Differential and Linear Cryptoanalysis. In Santis [San95], pages 356–365.
Available from: http://dx.doi.org/10.1007/BFb0053450.
2 citations on pages 59 and 94.

[CV02] Anne Canteaut and Marion Videau. Degree of Composition of Highly Nonlinear Functions and Applications to Higher Order Differential Cryptanalysis. In Knudsen [Knu02], pages 518–533.
Available from: http://dx.doi.org/10.1007/3-540-46035-7_34.
One citation on page 175.

[CWP12] Jiazhe Chen, Meiqin Wang, and Bart Preneel. Impossible Differential Cryptanalysis of the Lightweight Block Ciphers TEA, XTEA and HIGHT. In Mitrokotsa and Vaudenay [MV12b], pages 117–137.
Available from: http://dx.doi.org/10.1007/978-3-642-31410-0_8.
2 citations on pages 160 and 201.

[Dae91] Joan Daemen. Limitations of the Even-Mansour Construction. In Imai et al. [IRM93], pages 495–498.
Available from: http://dx.doi.org/10.1007/3-540-57332-1_46.
One citation on page 38.

[Dae95] Joan Daemen. *Cipher and hash function design strategies based on linear and differential cryptanalysis*. PhD thesis, K.U. Leuven, March 1995.
2 citations on pages 46 and 191.

[Dam91] Ivan Damgård, editor. *Advances in Cryptology - EUROCRYPT '90, Workshop on the Theory and Application of of Cryptographic Techniques, Aarhus, Denmark, May 21-24, 1990, Proceedings*, volume 473 of *Lecture Notes in Computer Science*. Springer, 1991.
4 citations on pages 278, 285, 286, and 295.

[Dav91] Donald W. Davies, editor. *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*. Springer, 1991.
4 citations on pages 256, 279, 285, and 286.

[DDKS12] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Efficient Dissection of Composite Problems, with Applications to Cryptanalysis, Knapsacks, and Combinatorial Search Problems. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 719–740. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-32009-5_42.
One citation on page 99.

[DDKS13a] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Improved Linear Sieving Techniques with Applications to Step-Reduced LED-64. *IACR Cryptology ePrint Archive*, 2013:634, 2013.
Available from: http://eprint.iacr.org/2013/634.
One citation on page 101.

[DDKS13b] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Key Recovery Attacks on 3-round Even-Mansour, 8-step LED-128, and Full AES^2. *IACR Cryptology ePrint Archive*, 2013:391, 2013.
Available from: http://eprint.iacr.org/2013/391.
3 citations on pages 116, 117, and 118.

[DDKS13c] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Key Recovery Attacks on 3-round Even-Mansour, 8-step LED-128, and Full AES^2. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013 (Part I)*, volume 8269 of *Lecture Notes in Computer Science*, pages 337–356. Springer, 2013.
Available from: http://dx.doi.org/10.1007/978-3-642-42033-7_18.
3 citations on pages 116, 117, and 118.

[DDKS13d] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Key Recovery Attacks on 3-round Even-Mansour, 8-step LED-128, and Full AES^2, Full Version, 2013. Submitted.
One citation on page 118.

[Den82] Dorothy Denning. *Cryptography and Data Security*. Addison-Wesley, Boston, Massachusetts, USA, June 1982.
One citation on page 97.

[Des94] Yvo Desmedt, editor. *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *Lecture Notes in Computer Science*. Springer, 1994.
3 citations on pages 275, 276, and 278.

[DF13] Patrick Derbez and Pierre-Alain Fouque. Exhausting Demirci-Selçuk Meet-in-the-Middle Attacks Against Reduced-Round AES. In Shiho Moriai, editor, *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers*, volume 8424 of *Lecture Notes in Computer Science*, pages 541–560. Springer, 2013.
Available from: https://doi.org/10.1007/978-3-662-43933-3_28,
DOI: doi:10.1007/978-3-662-43933-3_28.
2 citations on pages 184 and 185.

[DF15] Patrick Derbez and Pierre-Alain Fouque. Exhausting Demirci-Selçuk Meet-in-the-Middle Attacks against Reduced-Round AES. *IACR Cryptology ePrint Archive*, 2015:259, 2015.
Available from: http://eprint.iacr.org/2015/259.
2 citations on pages 184 and 185.

[DFJ13] Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting. In Johansson and Nguyen [JN13], pages 371–387.
Available from: https://doi.org/10.1007/978-3-642-38348-9_23,
DOI: doi:10.1007/978-3-642-38348-9_23.
2 citations on pages 184 and 185.

[DGPW12] Alexandre Duc, Jian Guo, Thomas Peyrin, and Lei Wei. Unaligned Rebound Attack: Application to Keccak. In Canteaut [Can12], pages 402–421.
Available from: http://dx.doi.org/10.1007/978-3-642-34047-5_23.
One citation on page 123.

[DGV93] Joan Daemen, René Govaerts, and Joos Vandewalle. A New Approach to Block Cipher Design. In Anderson [And94], pages 18–32.
Available from: http://dx.doi.org/10.1007/3-540-58108-1_2.
One citation on page 211.

[DH77] Whitfield Diffie and Martin E. Hellman. Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard. *Computer*, 10(6):74–84, June 1977.
Available from: http://dx.doi.org/10.1109/C-M.1977.217750,
DOI: doi:10.1109/C-M.1977.217750.
One citation on page 95.

[DID04] Daniel Denning, James Irvine, and Malachy Devlin. A Key Agile 17.4 Gbit/sec Camellia Implementation. In Jürgen Becker, Marco Platzner, and Serge Vernalde, editors, *FPL 2004*, volume 3203 of *Lecture Notes in Computer Science*, pages 546–554. Springer, 2004.
Available from: http://dx.doi.org/10.1007/978-3-540-30117-2_56.
2 citations on pages 234 and 237.

[DID05] Daniel Denning, James Irvine, and Malachy Devlin. A High-Throughput FPGA Camellia Implementation. In *Research in Microelectronics and Electronics*, volume 1, pages 137–140, 2005.
Available from: https://info.isl.ntt.co.jp/crypt/camellia/dl/reference/PRIME05.pdf,
DOI: doi:10.1109/RME.2005.1543022.
2 citations on pages 234 and 237.

[Din14] Itai Dinur. Improved Differential Cryptanalysis of Round-Reduced Speck. Cryptology ePrint Archive, Report 2014/320, May 2014.
Available from: http://eprint.iacr.org/.
One citation on page 225.

[DKR97] Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. The Block Cipher Square. In Biham [Bih97c], pages 149–165.
Available from: http://dx.doi.org/10.1007/BFb0052343.
4 citations on pages 46, 81, 158, and 186.

[DKS10a] Orr Dunkelman, Nathan Keller, and Adi Shamir. Improved Single-Key Attacks on 8-Round AES-192 and AES-256. In Abe [Abe10], pages 158–176.
Available from: https://doi.org/10.1007/978-3-642-17373-8_10,
DOI: doi:10.1007/978-3-642-17373-8_10.
2 citations on pages 184 and 185.

[DKS10b] Orr Dunkelman, Nathan Keller, and Adi Shamir. A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony. In Rabin [Rab10], pages 393–410.
Available from: http://dx.doi.org/10.1007/978-3-642-14623-7_21.
3 citations on pages 103, 106, and 175.

[DKS12] Orr Dunkelman, Nathan Keller, and Adi Shamir. Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In Pointcheval and Johansson [PJ12], pages 336–354.
Available from: http://dx.doi.org/10.1007/978-3-642-29011-4_21.
4 citations on pages 38, 115, 118, and 216.

[DKS15] Orr Dunkelman, Nathan Keller, and Adi Shamir. Improved Single-Key Attacks on 8-Round AES-192 and AES-256. *J. Cryptology*, 28(3):397–422, 2015.
Available from: https://doi.org/10.1007/s00145-013-9159-4,
DOI: doi:10.1007/s00145-013-9159-4.
2 citations on pages 184 and 185.

[DL96] Kenneth Dam and Herbert Lin, editors. *Cryptography's Role in Securing the Information Society*. Committurl to Study National Cryptography Policy; Computer Science and Telecommunications Board; Commission on Physical Sciences, Mathematics, and Applications; National Research Council; National Academy Press, Washington, D.C., May 1996.
Available from: http://cryptome.org/nrcindex.htm.
One citation on page 165.

[DM95] Donald W. Davies and Sean Murphy. Pairs and Triplets of DES S-Boxes. *J. Cryptology*, 8(1):1–25, 1995.
Available from: http://dx.doi.org/10.1007/BF00204799.
One citation on page 119.

[DP84] Donald W. Davies and Wyn L. Price. Digital signatures, an update. In *Proc. 5th International Conference on Computer Communication*, pages 845–849, October 1984.
One citation on page 67.

[DP15] Patrick Derbez and Léo Perrin. Meet-in-the-Middle Attacks and Structural Analysis of Round-Reduced PRINCE. In Gregor Leander, editor, *Fast Software Encryption - 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers*, volume 9054 of *Lecture Notes in Computer Science*, pages 190–216. Springer, 2015.
Available from: http://dx.doi.org/10.1007/978-3-662-48116-5_10,
DOI: doi:10.1007/978-3-662-48116-5_10.
One citation on page 219.

[DPAR] Joan Daemen, Michaël Peeters, Gilles Van Assche, and Vincent Rijmen. NOEKEON. Web site.
Available from: http://gro.noekeon.org.
One citation on page 192.

[DR01] Joan Daemen and Vincent Rijmen. The Wide Trail Design Strategy. In Bahram Honary, editor, *8th IMA International Conference on Cryptography and Coding*, volume 2260 of *Lecture Notes in Computer Science*, pages 222–238. Springer, 2001.
Available from: http://dx.doi.org/10.1007/3-540-45325-3_20.
2 citations on pages 33 and 35.

[DR02a] Joan Daemen and Vincent Rijmen. AES and the Wide Trail Design Strategy. In Knudsen [Knu02], pages 108–109.
Available from: http://dx.doi.org/10.1007/3-540-46035-7_7.
2 citations on pages 33 and 180.

[DR02b] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.
Available from: http://dx.doi.org/10.1007/978-3-662-04722-4,
DOI: doi:10.1007/978-3-662-04722-4.
2 citations on pages 39 and 180.

[DR02c] Joan Daemen and Vincent Rijmen, editors. *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers*, volume 2365 of *Lecture Notes in Computer Science*. Springer, 2002.
4 citations on pages 250, 277, 293, and 298.

[DR05] Joan Daemen and Vincent Rijmen. Probability distributions of Correlation and Differentials in Block Ciphers. *IACR Cryptology ePrint Archive*, 2005:212, 2005.
Available from: http://eprint.iacr.org/2005/212.
One citation on page 86.

[DR07] Joan Daemen and Vincent Rijmen. Probability distributions of correlation and differentials in block ciphers. *J. Mathematical Cryptology*, 1(3):221–242, 2007.
Available from: http://dx.doi.org/10.1515/JMC.2007.011.
One citation on page 86.

[DS08a] Hüseyin Demirci and Ali Aydin Selçuk. A Meet-in-the-Middle Attack on 8-Round AES. In Nyberg [Nyb08], pages 116–126.
Available from: https://doi.org/10.1007/978-3-540-71039-4_7,
DOI: doi:10.1007/978-3-540-71039-4_7.
3 citations on pages 182, 184, and 185.

[DS08b] Itai Dinur and Adi Shamir. Cube Attacks on Tweakable Black Box Polynomials. *IACR Cryptology ePrint Archive*, 2008:385, 2008.
Available from: http://eprint.iacr.org/2008/385.
One citation on page 82.

[DS09] Itai Dinur and Adi Shamir. Side Channel Cube Attacks on Block Ciphers. *IACR Cryptology ePrint Archive*, 2009:127, 2009.
Available from: http://eprint.iacr.org/2009/127.
One citation on page 82.

[DSP07] Orr Dunkelman, Gautham Sekar, and Bart Preneel. Improved Meet-in-the-Middle Attacks on Reduced-Round DES. In K. Srinathan, C. Pandu Rangan, and Moti Yung, editors, *INDOCRYPT 2007*, volume 4859 of *Lecture Notes in Computer Science*, pages 86–100. Springer, 2007.
Available from: http://dx.doi.org/10.1007/978-3-540-77026-8_8.
One citation on page 95.

[Dun09] Orr Dunkelman, editor. *Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers*, volume 5665 of *Lecture Notes in Computer Science*. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-03317-9.
3 citations on pages 245, 246, and 282.

[Dur64] Richard Durstenfeld. Algorithm 235: Random permutation. *Commun. ACM*, 7(7):420, 1964.
Available from: http://doi.acm.org/10.1145/364520.364540.
One citation on page 67.

[Dwo01] Morris J. Dworkin. SP 800-38A 2001 Edition. Recommendation for Block Cipher Modes of Operation: Methods and Techniques. Technical report, National Institute of Standards & Technology, Gaithersburg, MD, United States, December 2001.
One citation on page 66.

[Dwo05] Morris J. Dworkin. SP 800-38B. Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. Technical report, National Institute of Standards & Technology, Gaithersburg, MD, United States, May 2005.
One citation on page 68.

[ECR] ECRYPT. Lightweight Cryptography Lounge.
Available from: http://www.ecrypt.eu.org/lightweight/index.php.
3 citations on pages 235, 236, and 237.

[EGG+12] Thomas Eisenbarth, Zheng Gong, Tim Güneysu, Stefan Heyse, Sebastiaan Indesteege, Stéphanie Kerckhof, François Koeune, Tomislav Nad, Thomas Plos, Francesco Regazzoni, François-Xavier Standaert, and Loïc van Oldeneel tot Oldenzeel. Compact Implementation and Performance Evaluation of Block Ciphers in ATtiny Devices. In Mitrokotsa and Vaudenay [MV12b], pages 172–187.
Available from: http://dx.doi.org/10.1007/978-3-642-31410-0_11.
3 citations on pages 200, 238, and 239.

[EKM+08] Thomas Eisenbarth, Timo Kasper, Amir Moradi, Christof Paar, Mahmoud Salmasizadeh, and Mohammad T. Manzuri Shalmani. On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoqCode Hopping Scheme. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 203–220. Springer, 2008.
Available from: http://dx.doi.org/10.1007/978-3-540-85174-5_12.
One citation on page 137.

[EKP+07] Thomas Eisenbarth, Sandeep S. Kumar, Christof Paar, Axel Poschmann, and Leif Uhsadel. A Survey of Lightweight-Cryptography Implementations. *IEEE Design & Test of Computers*, 24(6):522–533, 2007.
Available from: http://doi.ieeecomputersociety.org/10.1109/MDT.2007.178.
One citation on page 238.

[EM91] Shimon Even and Yishay Mansour. A Construction of a Cioher From a Single Pseudorandom Permutation. In Imai et al. [IRM93], pages 210–224.
Available from: http://dx.doi.org/10.1007/3-540-57332-1_17.
One citation on page 38.

[EM97] Shimon Even and Yishay Mansour. A Construction of a Cipher from a Single Pseudorandom Permutation. *Journal of Cryptology*, 10(3):151–162, 1997.
Available from: http://dx.doi.org/10.1007/s001459900025.
One citation on page 38.

[ES] Niklas Eén and Niklas Sörensson. MiniSat 2.0. An open-source SAT solver package.
Available from: http://minisat.se.
One citation on page 121.

[ETCC07] Juan M. Estévez-Tapiador, John A. Clark, and Julio César Hernández Castro. Non-linear Cryptanalysis Revisited: Heuristic Search for Approximations to S-Boxes. In *11th IMA International Conference on Cryptography and Coding*, pages 99–117, 2007.
Available from: http://dx.doi.org/10.1007/978-3-540-77272-9_7.
One citation on page 92.

[Eve87] Jan-Hendrik Evertse. Linear Structures in Blockciphers. In Chaum and Price [CP88], pages 249–266.
Available from: http://dx.doi.org/10.1007/3-540-39118-5_23.
One citation on page 62.

[EYCP00] Adam J. Elbirt, W. Yip, B. Chetwynd, and Christof Paar. An FPGA Implementation and Performance Evaluation of the AES Block Cipher Candidate Algorithm Finalists. In *Third AES Candidate Conference*, pages 13–27, 2000.
3 citations on pages 234, 236, and 237.

[FD84] James H. Finch and E. Graham Dougall, editors. *A High Performance Encryption Algorithm*. North-Holland - Elsevier Science Publishers, 1984.
One citation on page 133.

[Fei73] Horst Feistel. Cryptography and Computer Privacy. *Scientific American*, 228:15–23, May 1973.
DOI: doi:10.1038/scientificamerican0573-15.
2 citations on pages 126 and 146.

[Fei92] Joan Feigenbaum, editor. *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*. Springer, 1992.
2 citations on pages 256 and 293.

[FGLM93] Jean-Charles Faugère, Patrizia M. Gianni, Daniel Lazard, and Teo Mora. Efficient Computation of Zero-Dimensional Gröbner Bases by Change of Ordering. *J. Symb. Comput.*, 16(4):329–344, 1993.
Available from: http://dx.doi.org/10.1006/jsco.1993.1051.
One citation on page 122.

[FK13a] Pierre-Alain Fouque and Pierre Karpman. Security Amplification against Meet-in-the-Middle Attacks Using Whitening. *IACR Cryptology ePrint Archive*, 2013:618, 2013.
Available from: http://eprint.iacr.org/2013/618.
2 citations on pages 39 and 66.

[FK13b] Pierre-Alain Fouque and Pierre Karpman. Security Amplification against Meet-in-the-Middle Attacks Using Whitening. In Martijn Stam, editor, *IMA Int. Conf.*, volume 8308 of *Lecture Notes in Computer Science*, pages 252–269. Springer, 2013.
Available from: http://dx.doi.org/10.1007/978-3-642-45239-0_15.
2 citations on pages 39 and 66.

[FKL+00] Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Michael Stay, David A. Wagner, and Doug Whiting. Improved Cryptanalysis of Rijndael. In Schneier [Sch01], pages 213–230.
Available from: https://doi.org/10.1007/3-540-44706-7_15,
DOI: doi:10.1007/3-540-44706-7_15.
2 citations on pages 184 and 185.

[FLS+10] Niels Ferguson, Stefan Lucks, Bruce Schneier, Doug Whiting, Mihir Bellare, Tadayoshi Kohno, Jon Callas, and Jesse Walker. The Skein Hash Function Family, October 2010.
Available from: http://www.skein-hash.info.
One citation on page 205.

[FN99] Amos Fiat and Moni Naor. Rigorous Time/Space Trade-offs for Inverting Functions. *SIAM J. Comput.*, 29(3):790–803, 1999.
Available from: http://dx.doi.org/10.1137/S0097539795280512.
2 citations on pages 97 and 98.

[FNS75] Horst Feistel, William A. Notz, and J. Lynn Smith. Some cryptographic techniques for machine-to-machine data communications. *Proceedings of the IEEE*, 63(11):1545–1554, Nov 1975.
Available from: http://dx.doi.org/10.1109/PROC.1975.10005,
DOI: doi:10.1109/PROC.1975.10005.
One citation on page 146.

[Fra04] Matthew K. Franklin, editor. *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*. Springer, 2004.
One citation on page 249.

[FS09] Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.
Available from: http://www.cambridge.org/uk/catalogue/catalogue.asp?isbn=9780521898065.
One citation on page 118.

[FY38] Ronald Aylmer Fisher and Frank Yates. *Statistical tables for biological, agricultural and medical research (3rd ed.)*. Oliver & Boyd, 1938.
One citation on page 67.

[GC01] Kris Gaj and Pawel Chodowiec. Fast Implementation and Fair Comparison of the Final Candidates for Advanced Encryption Standard Using Field Programmable Gate Arrays. In David Naccache, editor, *CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 84–99. Springer, 2001.
Available from: http://dx.doi.org/10.1007/3-540-45353-9_8.
One citation on page 235.

[GDC90] Helen Gustafson, Ed Dawson, and William J. Caelli. Comparison of Block Ciphers. In Jennifer Seberry and Josef Pieprzyk, editors, *AUSCRYPT '90*, volume 453 of *Lecture Notes in Computer Science*, pages 208–220. Springer, 1990.
Available from: http://dx.doi.org/10.1007/BFb0030362.
One citation on page 134.

[GG99] Guang Gong and Solomon W. Golomb. Transform domain analysis of DES. *IEEE Transactions on Information Theory*, 45(6):2065–2073, 1999.
Available from: http://dx.doi.org/10.1109/18.782138.
One citation on page 62.

[GG10] Guang Gong and Kishan Chand Gupta, editors. *Progress in Cryptology - INDOCRYPT 2010 - 11th International Conference on Cryptology in India, Hyderabad, India, December 12-15, 2010. Proceedings*, volume 6498 of *Lecture Notes in Computer Science*. Springer, 2010.
Available from: https://doi.org/10.1007/978-3-642-17401-8,
DOI: doi:10.1007/978-3-642-17401-8.
3 citations on pages 249, 281, and 293.

[GGH+98] Henri Gilbert, Marc Girault, Philippe Hoogvorst, Fabrice Noilhan, Thomas Pornin, Guillaume Poupard, Jacques Stern, and Serge Vaudenay. Decorrelated Fast Cipher: an AES Candidate – Extended Abstract, May 1998.
Available from: http://infoscience.epfl.ch/record/99475/files/GG+98b.ps.
One citation on page 177.

[GGNPS13] Benoît Gérard, Vincent Grosso, María Naya-Plasencia, and François-Xavier Standaert. Block Ciphers That Are Easier to Mask: How Far Can We Go? In Guido Bertoni and Jean-Sébastien Coron, editors, *CHES 2013*, volume 8086 of *Lecture Notes in Computer Science*, pages 383–399. Springer, 2013.
Available from: http://dx.doi.org/10.1007/978-3-642-40349-1_22.
One citation on page 229.

[GHJV00] Henri Gilbert, Helena Handschuh, Antoine Joux, and Serge Vaudenay. A Statistical Attack on RC6. In Schneier [Sch01], pages 64–74.
Available from: http://dx.doi.org/10.1007/3-540-44706-7_5.
One citation on page 166.

[GHL+07a] David Goldenberg, Susan Hohenberger, Moses Liskov, Elizabeth Crump Schwartz, and Hakan Seyalioglu. On Tweaking Luby-Rackoff Blockciphers. *IACR Cryptology ePrint Archive*, 2007:350, 2007.
Available from: http://eprint.iacr.org/2007/350.
One citation on page 29.

[GHL+07b] David Goldenberg, Susan Hohenberger, Moses Liskov, Elizabeth Crump Schwartz, and Hakan Seyalioglu. On Tweaking Luby-Rackoff Blockciphers. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 342–356. Springer, 2007.
Available from: http://dx.doi.org/10.1007/978-3-540-76900-2_21.
One citation on page 29.

[Gla00] Brian Gladman. AES Second Round Implementation Experience, 2000.
Available from: http://gladman.plushost.co.uk/oldsite/cryptography_technology/aesr2.
2 citations on pages 164 and 167.

[GLFV14] Vincent Grosso, Gaëtan Leurent, cois-Xavier Standaert Fran and Kerem Varici. LS-Designs: Bitslice Encryption for Efficient Masked Software Implementations. Preproceedings of FSE 2014, March 2014.
Available from: http://www.uclouvain.be/crypto/services/download/publications.pdf.86950b8b852267a1.6269736c6963655f6369706865722e706466.pdf.
One citation on page 230.

[GLRW10] Jian Guo, San Ling, Christian Rechberger, and Huaxiong Wang. Advanced Meet-in-the-Middle Preimage Attacks: First Results on Full Tiger, and Improved Results on MD4 and SHA-2. In Abe [Abe10], pages 56–75.
Available from: http://dx.doi.org/10.1007/978-3-642-17373-8_4,
DOI: doi:10.1007/978-3-642-17373-8.
One citation on page 95.

[GM01] Henri Gilbert and Marine Minier. New Results on the Pseudorandomness of Some Blockcipher Constructions. In Matsui [Mat02], pages 248–266.
Available from: http://dx.doi.org/10.1007/3-540-45473-X_21.
2 citations on pages 30 and 32.

[GMO09] Juan A. Garay, Atsuko Miyaji, and Akira Otsuka, editors. *Cryptology and Network Security, 8th International Conference, CANS 2009, Kanazawa, Japan, December 12-14, 2009. Proceedings*, volume 5888 of *Lecture Notes in Computer Science*. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-10433-6.
2 citations on pages 270 and 284.

[GNL11] Zheng Gong, Svetla Nikova, and Yurl Wei Law. KLEIN: A New Family of Lightweight Block Ciphers. In Ari Juels and Christof Paar, editors, *RFIDSec 2011*, volume 7055 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-25286-0_1.
2 citations on pages 224 and 235.

[GNNV00] Louis Granboulan, Phong Q. Nguyen, Fabrice Noilhan, and Serge Vaudenay. DFCv2. In Stinson and Tavares [ST01], pages 57–71.
Available from: http://dx.doi.org/10.1007/3-540-44983-3_5.
One citation on page 177.

[Gol96] Dieter Gollmann, editor. *Fast Software Encryption, Third International Workshop, Cambridge, UK, February 21-23, 1996, Proceedings*, volume 1039 of *Lecture Notes in Computer Science*. Springer, 1996.
4 citations on pages 245, 272, 287, and 294.

[Gol98] Oded Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*, volume 17 of *Algorithms and Combinatorics*. Springer, 1998.
Available from: http://link.springer.com/book/10.1007/978-3-662-12521-2/page/1.
One citation on page 67.

[GP07] Louis Granboulan and Thomas Pornin. Perfect Block Ciphers with Small Blocks. In Biryukov [Bir07], pages 452–465.
Available from: http://dx.doi.org/10.1007/978-3-540-74619-5_28,
DOI: doi:10.1007/978-3-540-74619-5.
One citation on page 67.

[GPP11] Jian Guo, Thomas Peyrin, and Axel Poschmann. The PHOTON Family of Lightweight Hash Functions. In Rogaway [Rog11], pages 222–239.
Available from: http://dx.doi.org/10.1007/978-3-642-22792-9_13,
DOI: doi:10.1007/978-3-642-22792-9.
One citation on page 49.

[GPPR11] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED Block Cipher. In Preneel and Takagi [PT11], pages 326–341.
Available from: http://dx.doi.org/10.1007/978-3-642-23951-9_22.
2 citations on pages 49 and 227.

[GPPR12] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED Block Cipher. *IACR Cryptology ePrint Archive*, 2012:600, 2012.
Available from: http://eprint.iacr.org/2012/600.
5 citations on pages 49, 123, 227, 228, and 236.

[GR13a] Kishan Chand Gupta and Indranil Ghosh Ray. On Constructions of Involutory MDS Matrices. In Amr Youssef, Abderrahmane Nitaj, and Aboul Ella Hassanien, editors, *AFRICACRYPT 2013*, volume 7918 of *Lecture Notes in Computer Science*, pages 43–60. Springer, 2013.
Available from: http://dx.doi.org/10.1007/978-3-642-38553-7_3.
One citation on page 48.

[GR13b] Kishan Chand Gupta and Indranil Ghosh Ray. On Constructions of MDS Matrices from Companion Matrices for Lightweight Cryptography. In Alfredo Cuzzocrea, Christian Kittl, Dimitris E. Simos, Edgar Weippl, and Lida Xu, editors, *CD-ARES Workshops*, volume 8128 of *Lecture Notes in Computer Science*, pages 29–43. Springer, 2013.
Available from: http://dx.doi.org/10.1007/978-3-642-40588-4_3.
One citation on page 50.

[GR13c] Kishan Chand Gupta and Indranil Ghosh Ray. On Constructions of MDS Matrices from Companion Matrices for Lightweight Cryptography. *IACR Cryptology ePrint Archive*, 2013:56, 2013.
Available from: http://eprint.iacr.org/2013/056.
One citation on page 50.

[Gro04] Aaron Grothe. Linux Kernel `tea.c`, 2004.
Available from: https://git.kernel.org/cgit/linux/kernel/git/stable/linux-stable.git/tree/crypto/tea.c?id=refs/tags/v2.6.14.
2 citations on pages 160 and 162.

[GRR16a] Lorenzo Grassi, Christian Rechberger, and Sondre Rønjom. Subspace Trail Cryptanalysis and its Applications to AES. *IACR Trans. Symmetric Cryptol.*, 2016(2):192–225, 2016.
Available from: http://tosc.iacr.org/index.php/ToSC/article/view/571.
One citation on page 186.

[GRR16b] Lorenzo Grassi, Christian Rechberger, and Sondre Rønjom. Subspace Trail Cryptanalysis and its Applications to AES. *IACR Cryptology ePrint Archive*, 2016:592, 2016.
Available from: http://eprint.iacr.org/2016/592.
One citation on page 186.

[GT77] Edna K. Grossman and Bryant Tuckerman. Analysis of a Feistel-like cipher weakened by having no rotating key. Technical Report RC 6375, IBM Thomas J. Watson Research Center, 1977.
2 citations on pages 107 and 112.

[GT78] Edna K. Grossman and Bryant Tuckerman. Analysis of a Weakened Feistel-like Cipher. In *1978 International Conference on Communications*, pages 46.3.1–46.3.5. Alger Press Limited, 1978.
2 citations on pages 107 and 112.

[GTR+07] Johann Großschädl, Stefan Tillich, Christian Rechberger, Michael Hofmann, and Marcel Medwed. Energy evaluation of software implementations of block ciphers under memory constraints. In Rudy Lauwereins and Jan Madsen, editors, *DATE 2007*, pages 1110–1115. ACM, 2007.
Available from: http://doi.acm.org/10.1145/1266366.1266607.
2 citations on pages 164 and 170.

[Had93] Jacques Hadamard. Résolution d'une question relative aux déterminants. *Bulletin des Sciences Mathématiques*, 17:240–246, 1893.
One citation on page 48.

[Hal09] Shai Halevi, editor. *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-03356-8.
3 citations on pages 247, 253, and 282.

[Haw98] Philip Hawkes. Differential-Linear Weak Key Classes of IDEA. In Nyberg [Nyb98], pages 112–126.
Available from: http://dx.doi.org/10.1007/BFb0054121.
One citation on page 94.

[HCN08] Miia Hermelin, Joo Yeon Cho, and Kaisa Nyberg. Multidimensional Linear Cryptanalysis of Reduced Round Serpent. In Yi Mu, Willy Susilo, and Jennifer Seberry, editors, *ACISP*, volume 5107 of *Lecture Notes in Computer Science*, pages 203–215. Springer, 2008.
Available from: http://dx.doi.org/10.1007/978-3-540-70500-0_15.
One citation on page 91.

[Hel80] Martin E. Hellman. A cryptanalytic time-memory trade-off. *IEEE Transactions on Information Theory*, 26(4):401–406, 1980.
Available from: http://doi.ieeecomputersociety.org/10.1109/TIT.1980.1056220.
One citation on page 96.

[Hel94] Tor Helleseth, editor. *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, volume 765 of *Lecture Notes in Computer Science*. Springer 1993, 1994.
3 citations on pages 252, 280, and 285.

[Hey02] Howard M. Heys. A Tutorial on Linear and Differential Cryptanalysis, 2002.
Available from: http://www.engr.mun.ca/~howard/PAPERS/ldc_tutorial.pdf,
DOI: doi:10.1080/0161-110291890885.
One citation on page 44.

[HH04] Helena Handschuh and M. Anwar Hasan, editors. *Selected Areas in Cryptography, 11th International Workshop, SAC 2004, Waterloo, Canada, August 9-10, 2004, Revised Selected Papers*, volume 3357 of *Lecture Notes in Computer Science*. Springer, 2004.
One citation on page 272.

[HI10] Seokhie Hong and Tetsu Iwata, editors. *Fast Software Encryption, 17th International Workshop, FSE 2010, Seoul, Korea, February 7-10, 2010, Revised Selected Papers*, volume 6147 of *Lecture Notes in Computer Science*. Springer, 2010.
Available from: http://dx.doi.org/10.1007/978-3-642-13858-4.
4 citations on pages 251, 274, 285, and 291.

[Hir06] Shoichi Hirose. Some Plausible Constructions of Double-Block-Length Hash Functions. In Robshaw [Rob06], pages 210–225.
Available from: http://dx.doi.org/10.1007/11799313_14.
One citation on page 67.

[HL12] Jialin Huang and Xuejia Lai. Revisiting Key Schedule's Diffusion In Relation With Round Function's Diffusion. *IACR Cryptology ePrint Archive*, 2012:415, 2012.
Available from: http://eprint.iacr.org/2012/415.
One citation on page 40.

[HL14] Jialin Huang and Xuejia Lai. Revisiting key schedule's diffusion in relation with round function's diffusion. *Des. Codes Cryptography*, 73(1):85–103, 2014.
Available from: http://dx.doi.org/10.1007/s10623-013-9804-9.
One citation on page 40.

[HMPM05] Creighton T. R. Hager, Scott F. Midkiff, Jung-Min Park, and Thomas L. Martin. Performance and Energy Efficiency of Block Ciphers in Personal Digital Assistants. In *PerCom*, pages 127–136. IEEE Computer Society, 2005.
Available from: http://doi.ieeecomputersociety.org/10.1109/PERCOM.2005.29.
One citation on page 142.

[HN11] Miia Hermelin and Kaisa Nyberg. Linear Cryptanalysis Using Multiple Linear Approximations. *IACR Cryptology ePrint Archive*, 2011:93, 2011.
Available from: http://eprint.iacr.org/2011/093.
One citation on page 91.

[Hou00] Xiang-Dong Hou. New Constructions of Bent Functions. *Journal of Combinatorics, Information and System Sciences*, 25(1–4):173–189, 2000.
One citation on page 59.

[HR10a] Viet Tung Hoang and Phillip Rogaway. On generalized Feistel networks. *IACR Cryptology ePrint Archive*, 2010:301, 2010.
Available from: http://eprint.iacr.org/2010/301.
3 citations on pages 32, 68, and 167.

[HR10b] Viet Tung Hoang and Phillip Rogaway. On Generalized Feistel Networks. In Rabin [Rab10], pages 613–630.
Available from: http://dx.doi.org/10.1007/978-3-642-14623-7_33.
3 citations on pages 32, 68, and 167.

[HSH+06] Deukjo Hong, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lurl, Bonseok Koo, Changhoon Lurl, Donghoon Chang, Jaesang Lurl, Kitae Jeong, Hyun Kim, Jongsung Kim, and Seongtaek Chee. HIGHT: A New Block Cipher Suitable for Low-Resource Device. In Louis Goubin and Mitsuru Matsui, editors,

CHES 2006, volume 4249 of *Lecture Notes in Computer Science*, pages 46–59. Springer, 2006.
Available from: http://dx.doi.org/10.1007/11894063_4.
2 citations on pages 200 and 235.

[HSK02] Yasuo Hatano, Hiroki Sekine, and Toshinobu Kaneko. Higher Order Differential Attack of Camellia (II). In Kaisa Nyberg and Howard M. Heys, editors, *Selected Areas in Cryptography 2002*, volume 2595 of *Lecture Notes in Computer Science*, pages 129–146. Springer, 2002.
Available from: http://dx.doi.org/10.1007/3-540-36492-7_10.
One citation on page 172.

[HT93] Howard M. Heys and Stafford E. Tavares. Cryptanalysis Of Tree-Structured Substitution-Permutation Networks. *Electronic Letters*, 29(1):40–41, January 1993.
Available from: http://dx.doi.org/10.1049/el:19930026.
One citation on page 43.

[HT95] Howard M. Heys and Stafford E. Tavares. Known Plaintext Cryptanalysis Of Tree-Structured Block Ciphers. *Electronic Letters*, 31(10):784–785, May 1995.
Available from: http://dx.doi.org/10.1049/el:19950571.
One citation on page 43.

[HWG10] Swee-Huay Heng, Rebecca N. Wright, and Bok-Min Goi, editors. *Cryptology and Network Security - 9th International Conference, CANS 2010, Kuala Lumpur, Malaysia, December 12-14, 2010. Proceedings*, volume 6467 of *Lecture Notes in Computer Science*. Springer, 2010.
Available from: http://dx.doi.org/10.1007/978-3-642-17619-7.
One citation on page 227.

[HZX99] Yupu Hu, Yuqing Zhang, and Guozhen Xiao. Integral cryptanalysis of SAFER. *Electronics Letters*, 35:1458–1459, August 1999.
DOI: doi:10.1049/el:19990979.
One citation on page 153.

[IKE+13] Yasutaka Igarashi, Toshinobu Kaneko, Yutaka Eguchi, Takahiro Murai, Ryutaro Sueyoshi, Yosuke Hashiguchi, Seiji Fukushima, and Tomohiro Hachino. The Improved 32nd-Order Differential Attack on 8 Rounds of MISTY2 without FL Functions. *International Journal of Cyber-Security and Digital Forensics (IJCSDF)*, 2(3):27–34, October 2013.
Available from: http://sdiwc.net/digital-library/the-improved-32ndorder-differential-attack-on-8-rounds-of-misty2-without-fl-functions.
3 citations on pages 81, 174, and 175.

[IRM93] Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors. *Advances in Cryptology - ASIACRYPT '91, International Conference on the Theory and Applications of Cryptology, Fujiyoshida, Japan, November 11-14, 1991, Proceedings*, volume 739 of *Lecture Notes in Computer Science*. Springer, 1993.
2 citations on pages 260 and 264.

[IS12] Takanori Isobe and Kyoji Shibutani. Security Analysis of the Lightweight Block Ciphers XTEA, LED and Piccolo. In Willy Susilo, Yi Mu, and Jennifer Seberry, editors, *ACISP*, volume 7372 of *Lecture Notes in Computer Science*, pages 71–86. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-31448-3_6.
One citation on page 98.

[ISO91] ISO. ISO-IEC/JTC1/SC27/WG2 N98, Hash functions using a pseudo random algorithm, working document, 1991.
One citation on page 67.

[ISO99] ISO (International Organization for Standardization). ISO/IEC 9797-1:1999 Information technology – Security techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms using a block cipher, 1999.
Available from: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=30656.
One citation on page 68.

[ISSK09] Maryam Izadi, Babak Sadeghiyan, Seyed Saeed Sadeghian, and Hossein Arabnezhad Khanooki. MIBS: A New Lightweight Block Cipher. In Garay et al. [GMO09], pages 334–348.
Available from: http://dx.doi.org/10.1007/978-3-642-10433-6_22.
2 citations on pages 227 and 238.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private Circuits: Securing Hardware against Probing Attacks. In Boneh [Bon03], pages 463–481.
Available from: http://dx.doi.org/10.1007/978-3-540-45146-4_27.
One citation on page 228.

[JH98] Michael J. Jacobson, Jr. and Klaus Huber. The Magenta Block Cipher Algorithm. In *Proceedings of the First AES Candidate Conference*. National Institute of Standard and Technology, 1998.
Available from: http://csrc.nist.gov/archive/aes/index.html.
One citation on page 179.

[JK97] Thomas Jakobsen and Lars R. Knudsen. The Interpolation Attack on Block Ciphers. In Biham [Bih97c], pages 28–40.
Available from: http://dx.doi.org/10.1007/BFb0052332.
One citation on page 120.

[JKL+12] Kitae Jeong, HyungChul Kang, Changhoon Lurl, Jaechul Sung, and Seokhie Hong. Biclique Cryptanalysis of Lightweight Block Ciphers PRESENT, Piccolo and LED. *IACR Cryptology ePrint Archive*, 2012:621, 2012.
Available from: http://eprint.iacr.org/2012/621.
2 citations on pages 205 and 227.

[JKL+13] Kitae Jeong, HyungChul Kang, Changhoon Lurl, Jaechul Sung, Seokhie Hong, and JongIn Lim. Weakness of lightweight block ciphers mCrypton and LED against biclique cryptanalysis. *Peer-to-Peer Networking and Applications*, pages 1–17, 2013.
Available from: http://dx.doi.org/10.1007/s12083-013-0208-4,
DOI: doi:10.1007/s12083-013-0208-4.
One citation on page 197.

[JN13] Thomas Johansson and Phong Q. Nguyen, editors. *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*. Springer, 2013.
Available from: https://doi.org/10.1007/978-3-642-38348-9,
DOI: doi:10.1007/978-3-642-38348-9.
3 citations on pages 254, 261, and 287.

[JNP+13] Jérmy Jean, Ivica Nikolić, Thomas Peyrin, Lei Wang, and Shuang Wu. Security Analysis of PRINCE, 2013. To appear in: Proceedings of FSE 2013, the 20th International Workshop on Fast Software Encryption.
One citation on page 219.

[Joh03] Thomas Johansson, editor. *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24-26, 2003, Revised Papers*, volume 2887 of *Lecture Notes in Computer Science*. Springer, 2003.
3 citations on pages 250, 253, and 289.

[Jou09] Antoine Joux, editor. *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 of *Lecture Notes in Computer Science*. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-01001-9.
One citation on page 82.

[Jou11] Antoine Joux, editor. *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, volume 6733 of *Lecture Notes in Computer Science*. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-21702-9.
2 citations on pages 247 and 252.

[JP03] Jorge Nakahara Jr. and Bart Preneel. Impossible Differential Attacks on Reduced-Round SAFER Ciphers, 2003. NESSIE Public Report, NES/DOC/KUL/WP5/30/1.
One citation on page 154.

[JPV00] Jorge Nakahara Jr., Bart Preneel, and Joos Vandewalle. Linear Cryptanalysis of Reduced-Round Versions of the SAFER Block Cipher Family. In Schneier [Sch01], pages 244–261.
Available from: http://dx.doi.org/10.1007/3-540-44706-7_17.
One citation on page 154.

[JPV01] Jorge Nakahara Jr., Bart Preneel, and Joos Vandewalle. Linear Cryptanalysis of Reduced-Round SAFER++. In *Proceedings of the Second NESSIE Workshop*. NESSIE, September 2001.
One citation on page 154.

[JPV04] Jorge Nakahara Jr., Bart Preneel, and Joos Vandewalle. The Biryukov-Demirci Attack on Reduced-Round Versions of IDEA and MESH Ciphers. In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *ACISP*, volume 3108 of *Lecture Notes in Computer Science*, pages 98–109. Springer, 2004.
Available from: http://dx.doi.org/10.1007/978-3-540-27800-9_9.
One citation on page 145.

[JRPV03] Jorge Nakahara Jr., Vincent Rijmen, Bart Preneel, and Joos Vandewalle. The MESH Block Ciphers. In Kijoon Chae and Moti Yung, editors, *WISA 2003*, volume 2908 of *Lecture Notes in Computer Science*, pages 458–473. Springer, 2003.
Available from: http://dx.doi.org/10.1007/978-3-540-24591-9_34.
2 citations on pages 37 and 145.

[JRS88] Burton S. Kaliski Jr., Ronald L. Rivest, and Alan T. Sherman. Is the Data Encryption Standard a Group? (Results of Cycling Experiments on DES). *Journal of Cryptology*, 1(1):3–36, 1988.
Available from: http://dx.doi.org/10.1007/BF00206323.
One citation on page 146.

[Jun01a] Pascal Junod. On the Complexity of Matsui's Attack. *IACR Cryptology ePrint Archive*, 2001:56, 2001.
Available from: http://eprint.iacr.org/2001/056.
2 citations on pages 118 and 130.

[Jun01b] Pascal Junod. On the Complexity of Matsui's Attack. In Serge Vaudenay and Amr M. Youssef, editors, *Selected Areas in Cryptography 2001*, volume 2259 of *Lecture Notes in Computer Science*, pages 199–211. Springer, 2001.
Available from: http://dx.doi.org/10.1007/3-540-45537-X_16.
2 citations on pages 118 and 130.

[Jun03a] Pascal Junod. On the Optimality of Linear, Differential and Sequential Distinguishers. *IACR Cryptology ePrint Archive*, 2003:64, 2003.
Available from: http://eprint.iacr.org/2003/064.
One citation on page 118.

[Jun03b] Pascal Junod. On the Optimality of Linear, Differential, and Sequential Distinguishers. In Biham [Bih03], pages 17–32.
Available from: http://dx.doi.org/10.1007/3-540-39200-9_2.
One citation on page 118.

[JV04a] Pascal Junod and Serge Vaudenay. FOX : A New Family of Block Ciphers. In Handschuh and Hasan [HH04], pages 114–129.
Available from: http://dx.doi.org/10.1007/978-3-540-30564-4_8.
2 citations on pages 118 and 193.

[JV04b] Pascal Junod and Serge Vaudenay. Perfect Diffusion Primitives for Block Ciphers – Building Efficient MDS Matrices. In Handschuh and Hasan [HH04], pages 84–99.
Available from: http://dx.doi.org/10.1007/978-3-540-30564-4_6.
2 citations on pages 48 and 49.

[Kan07] Gen Kanai. Camellia cipher added to Firefox, July 2007.
Available from: https://blog.mozilla.org/gen/2007/07/30/camellia-cipher-added-to-firefox/.
One citation on page 173.

[Kap08] Jens-Peter Kaps. Chai-Tea, Cryptographic Hardware Implementations of xTEA. In Chowdhury et al. [CRD08], pages 363–375.
Available from: http://dx.doi.org/10.1007/978-3-540-89754-5_28,
DOI: doi:10.1007/978-3-540-89754-5.
One citation on page 161.

[KB96] Lars R. Knudsen and Thomas A. Berson. Truncated Differentials of SAFER. In Gollmann [Gol96], pages 15–26.
Available from: http://dx.doi.org/10.1007/3-540-60865-6_38.
3 citations on pages 80, 153, and 154.

[KBS90] Gideon J. Kuhn, Frederick J. Brouwer, and Willem Smit. A Fast Multipurpose Encryption Chip. In *Infosec '90 Symposium, Pretoria*. MIKOMTEK CSIR, March 1990.
One citation on page 136.

[KD79] John B. Kam and George I. Davida. Structured Design of Substitution-Permutation Encryption Networks. *IEEE Transactions on Computers*, 28(10):747–753, 1979.
Available from: http://doi.ieeecomputersociety.org/10.1109/TC.1979.1675242.
One citation on page 42.

[KDH13] Ferhat Karakoç, Hüseyin Demirci, and A. Emre Harmanci. Biclique cryptanalysis of LBlock and TWINE. *Information Processing Letters*, 113(12):423–429, 2013.
Available from: http://dx.doi.org/10.1016/j.ipl.2013.03.011.
One citation on page 215.

[Ker83a] Auguste Kerckhoffs (von Nieuwenhof). La Cryptographie Militaire I-III. (French) [Military cryptography, Parts I-III]. *Journal des Sciences Militaires*, IX:5–38, January 1883.
Available from: http://www.cl.cam.ac.uk/~fapp2/kerckhoffs/la_cryptographie_militaire_i.htm.
One citation on page 21.

[Ker83b] Auguste Kerckhoffs (von Nieuwenhof). La Cryptographie Militaire IV. (French) [Military cryptography – Part IV]. *Journal des Sciences Militaires*, IX:161–191, February 1883.
Available from: http://www.cl.cam.ac.uk/~fapp2/kerckhoffs/la_cryptographie_militaire_i.htm.
One citation on page 21.

[KHL+04] Youngdai Ko, Seokhie Hong, Wonil Lurl, Sangjin Lurl, and Ju-Sung Kang. Related Key Differential Attacks on 27 Rounds of XTEA and Full-Round GOST. In Roy and Meier [RM04], pages 299–316.
Available from: http://dx.doi.org/10.1007/978-3-540-25937-4_19.
One citation on page 161.

[KHP07] Jongsung Kim, Seokhie Hong, and Bart Preneel. Related-Key Rectangle Attacks on Reduced AES-192 and AES-256. In Biryukov [Bir07], pages 225–241.
Available from: https://doi.org/10.1007/978-3-540-74619-5_15,
DOI: doi:10.1007/978-3-540-74619-5_15.
One citation on page 185.

[KKS00] John Kelsey, Tadayoshi Kohno, and Bruce Schneier. Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent. In Schneier [Sch01], pages 75–93.
Available from: http://dx.doi.org/10.1007/3-540-44706-7_6.
3 citations on pages 83, 168, and 169.

[KLPR10] Lars R. Knudsen, Gregor Leander, Axel Poschmann, and Matthew J. B. Robshaw. PRINTcipher: A Block Cipher for IC-Printing. In Mangard and Standaert [MS10], pages 16–32.
Available from: http://dx.doi.org/10.1007/978-3-642-15031-9_2.
One citation on page 209.

[KLR12] Dmitry Khovratovich, Gaëtan Leurent, and Christian Rechberger. Narrow-Bicliques: Cryptanalysis of Full IDEA. In Pointcheval and Johansson [PJ12], pages 392–410.
Available from: http://dx.doi.org/10.1007/978-3-642-29011-4_24.
4 citations on pages 96, 98, 104, and 144.

[KM00a] Lars R. Knudsen and John Erik Mathiassen. A Chosen-Plaintext Linear Attack on DES. In Schneier [Sch01], pages 262–272.
Available from: http://dx.doi.org/10.1007/3-540-44706-7_18.
One citation on page 91.

[KM00b] Lars R. Knudsen and Willi Meier. Correlations in RC6 with a Reduced Number of Rounds. In Schneier [Sch01], pages 94–108.
Available from: http://dx.doi.org/10.1007/3-540-44706-7_7.
One citation on page 166.

[KM04] Lars R. Knudsen and John Erik Mathiassen. On the Role of Key Schedules in Attacks on Iterated Ciphers. In Pierangela Samarati, Peter Y. A. Ryan, Dieter Gollmann, and Refik Molva, editors, *ESORICS 2004*, volume 3193 of *Lecture Notes in Computer Science*, pages 322–334. Springer, 2004.
Available from: http://dx.doi.org/10.1007/978-3-540-30108-0_20.
One citation on page 42.

[KM05] Sébastien Kunz-Jacques and Frédéric Muller. New Improvements of Davies-Murphy Cryptanalysis. In Roy [Roy05], pages 425–442.
Available from: http://dx.doi.org/10.1007/11593447_23,
DOI: doi:10.1007/11593447.
One citation on page 120.

[KMA+98] Masayuki Kanda, Shiho Moriai, Kazumaro Aoki, Hiroki Ueda, Miyako Ohkubo, Youichi Takashima, Kazuo Ohta, and Tsutomu Matsumoto. E2 – a candidate cipher for AES. In *First AES Candidate Conference*. National Institute of Standard and Technology, 1998.
Available from: http://csrc.nist.gov/archive/aes/round1/conf1/saferpls-slides.pdf.
2 citations on pages 170 and 173.

[KMNP10] Simon Knellwolf, Willi Meier, and María Naya-Plasencia. Conditional Differential Cryptanalysis of NLFSR-Based Cryptosystems. In Abe [Abe10], pages 130–145.
Available from: http://dx.doi.org/10.1007/978-3-642-17373-8_8,
DOI: doi:10.1007/978-3-642-17373-8.
One citation on page 208.

[KMNP11] Simon Knellwolf, Willi Meier, and María Naya-Plasencia. Conditional Differential Cryptanalysis of Trivium and KATAN. In Miri and Vaudenay [MV12a], pages 200–212.
Available from: http://dx.doi.org/10.1007/978-3-642-28496-0_12.
One citation on page 208.

[KN10] Dmitry Khovratovich and Ivica Nikolić. Rotational Cryptanalysis of ARX. In Hong and Iwata [HI10], pages 333–346.
Available from: http://dx.doi.org/10.1007/978-3-642-13858-4_19.
One citation on page 207.

[KNR10] Dmitry Khovratovich, Ivica Nikolić, and Christian Rechberger. Rotational Rebound Attacks on Reduced Skein. In Abe [Abe10], pages 1–19.
Available from: http://dx.doi.org/10.1007/978-3-642-17373-8_1,
DOI: doi:10.1007/978-3-642-17373-8.
One citation on page 207.

[Knu94] Lars R. Knudsen. Truncated and Higher Order Differentials. In Preneel [Pre95], pages 196–211.
Available from: http://dx.doi.org/10.1007/3-540-60590-8_16.
One citation on page 80.

[Knu95a] Lars Knudsen. Announcement by James L. Massey of a STRENGTHENED KEY SCHEDULE for the cipher SAFER (Secure And Fast Encryption Routine), for both 64 and 128 bit key lengths, September 1995. Post to Usenet Newsgroup `sci.crypt.research`.
Available from: https://groups.google.com/d/msg/sci.crypt.research/gHOEDoAOQak/2HmtxA7A-30J.
One citation on page 150.

[Knu95b] Lars R. Knudsen. A Key-schedule Weakness in SAFER K-64. In Don Coppersmith, editor, *CRYPTO '95*, volume 963 of *Lecture Notes in Computer Science*, pages 274–286. Springer, 1995.
Available from: http://dx.doi.org/10.1007/3-540-44750-4_22.
One citation on page 150.

[Knu97] Donald E. Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms, 3nd Edition*. Addison-Wesley, 1997.
One citation on page 67.

[Knu98] Lars R. Knudsen. DEAL – A 128-bit Block Cipher. Technical report, NIST AES Proposal, February 1998.
Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.7982.
2 citations on pages 84 and 177.

[Knu99] Lars R. Knudsen, editor. *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*, volume 1636 of *Lecture Notes in Computer Science*. Springer, 1999.
7 citations on pages 249, 256, 275, 278, 280, 283, and 295.

[Knu00] Lars R. Knudsen. A Detailed Analysis of SAFER K. *Journal of Cryptology*, 13(4):417–436, 2000.
Available from: http://dx.doi.org/10.1007/s001450010004.
3 citations on pages 150, 153, and 154.

[Knu02] Lars R. Knudsen, editor. *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, volume 2332 of *Lecture Notes in Computer Science*. Springer, 2002.
2 citations on pages 260 and 263.

[Kob96] Neal Koblitz, editor. *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*. Springer, 1996.
3 citations on pages 275, 276, and 295.

[KR94a] Burton S. Kaliski, Jr. and Matthew J. B. Robshaw. Linear Cryptanalysis Using Multiple Approximations. In Desmedt [Des94], pages 26–39.
Available from: http://dx.doi.org/10.1007/3-540-48658-5_4.
One citation on page 91.

[KR94b] Burton S. Kaliski, Jr. and Matthew J. B. Robshaw. Linear Cryptanalysis Using Multiple Approximations and FEAL. In Preneel [Pre95], pages 249–264.
Available from: http://dx.doi.org/10.1007/3-540-60590-8_19.
One citation on page 91.

[KR96a] Burton S. Kaliski, Jr. and Matthew J. B. Robshaw. Algorithm Alley: Multiple Encryption: Weighing Security and Performance. *Dr. Dobb's Journal of Software Tools*, 21(1):123–127, January 1996.
Available from: http://www.drdobbs.com/algorithm-alley/184409815.
One citation on page 132.

[KR96b] Joe Kilian and Phillip Rogaway. How to Protect DES Against Exhaustive Key Search. In Koblitz [Kob96], pages 252–267.
Available from: http://dx.doi.org/10.1007/3-540-68697-5_20.
2 citations on pages 39 and 132.

[KR96c] Lars R. Knudsen and Matthew J. B. Robshaw. Non-Linear Approximations in Linear Cryptoanalysis. In Ueli M. Maurer, editor, *EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 224–236. Springer, 1996.
Available from: http://dx.doi.org/10.1007/3-540-68339-9_20.
One citation on page 92.

[KR99a] Lars R. Knudsen and Vincent Rijmen. On the Decorrelated Fast Cipher (DFC) and Its Theory. In Knudsen [Knu99], pages 81–94.
Available from: http://dx.doi.org/10.1007/3-540-48519-8_7.
One citation on page 178.

[KR99b] Lars R. Knudsen and Vincent Rijmen. Weaknesses in LOKI97. In *Proceedings of the Second AES Candidate Conference*, pages 168–174, March 1999.
Available from: http://seit.unsw.adfa.edu.au/staff/sites/lpb/research/loki97/knudsen99.pdf.
One citation on page 179.

[KR01a] Joe Kilian and Phillip Rogaway. How to Protect DES Against Exhaustive Key Search (an Analysis of DESX). *Journal of Cryptology*, 14(1):17–35, 2001.
Available from: http://dx.doi.org/10.1007/s001450010015.
2 citations on pages 39 and 132.

[KR01b] Lars R. Knudsen and Håvard Raddum. On Noekeon, 2001.
Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.20.6349.
One citation on page 191.

[KR11] Lars R. Knudsen and Matthew Robshaw. *The Block Cipher Companion*. Information security and cryptography. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-17342-4.
One citation on page 4.

[Kra94] Hugo Krawczyk. LFSR-based Hashing and Authentication. In Desmedt [Des94], pages 129–139.
Available from: http://dx.doi.org/10.1007/3-540-48658-5_15.
One citation on page 67.

[KRRR98] Lars R. Knudsen, Vincent Rijmen, Ronald L. Rivest, and Matthew J. B. Robshaw. On the Design and Security of RC2. In Vaudenay [Vau98a], pages 206–221.
Available from: http://dx.doi.org/10.1007/3-540-69710-1_14.
One citation on page 141.

[KRS+02] François Koeune, Gaël Rouvroy, François-Xavier Standaert, Jean-Jacques Quisquater, Jean-Pierre David, and Jean-Didier Legat. An FPGA Implementation of the Linear Cryptanalysis. In Manfred Glesner, Peter Zipf, and Michel Renovell, editors, *FPL*, volume 2438 of *Lecture Notes in Computer Science*, pages 845–852. Springer, 2002.
Available from: http://dx.doi.org/10.1007/3-540-46117-5_87.
2 citations on pages 91 and 130.

[KRS12] Dmitry Khovratovich, Christian Rechberger, and Alexandra Savelieva. Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family. In Canteaut [Can12], pages 244–263.
Available from: http://dx.doi.org/10.1007/978-3-642-34047-5_15.
2 citations on pages 95 and 101.

[KRW99] Lars R. Knudsen, Matthew J. B. Robshaw, and David Wagner. Truncated Differentials and Skipjack. In Wiener [Wie99], pages 165–180.
Available from: http://dx.doi.org/10.1007/3-540-48405-1_11.
One citation on page 80.

[KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization. In Wiener [Wie99], pages 19–30.
Available from: http://dx.doi.org/10.1007/3-540-48405-1_2.
One citation on page 121.

[KS00] John Kelsey and Bruce Schneier. MARS Attacks! Preliminary Cryptanalysis of Reduced-Round MARS Variants. In *Proceedings of the Third AES Candidate Conference*, pages 169–185, 2000.
One citation on page 168.

[KSGK04] Paris Kitsos, Nicolas Sklavos, Michalis D. Galanis, and Odysseas G. Koufopavlou. 64-bit Block ciphers: hardware implementations and comparison analysis. *Computers & Electrical Engineering*, 30(8):593–604, 2004.
Available from: http://dx.doi.org/10.1016/j.compeleceng.2004.11.001.
3 citations on pages 234, 235, and 236.

[KSS97] Yasuyoshi Kaneko, Fumihiko Sano, and Kouichi Sakurai. On Provable Security against Differential and Linear Cryptanalysis in Generalized Feistel Ciphers with Multiple Random Functions. In *Selected Areas in Cryptography 1997*, pages 185–198, 1997.
Available from: http://www.sacconference.org/SAC-history.html.
2 citations on pages 30 and 33.

[KSW96] John Kelsey, Bruce Schneier, and David Wagner. Key-Schedule Cryptoanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In Koblitz [Kob96], pages 237–251.
Available from: http://dx.doi.org/10.1007/3-540-68697-5_19.
2 citations on pages 154 and 160.

[KSW97] John Kelsey, Bruce Schneier, and David Wagner. Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA. In Yongfei Han, Tatsuaki Okamoto, and Sihan Qing, editors, *ICICS 1997*, volume 1334 of *Lecture Notes in Computer Science*, pages 233–246. Springer, 1997.
Available from: http://dx.doi.org/10.1007/BFb0028479.
3 citations on pages 136, 141, and 160.

[Kuh88] Gideon J. Kuhn. Algorithms for Self-Synchronizing Ciphers. In *Comsig 88, Southern Africa Conference on Communications and Signal Processing, Pretoria*, pages 159–164. CSIR/IEEE, June 1988.
Available from: https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=537.
One citation on page 136.

[Küh01] Ulrich Kühn. Cryptanalysis of Reduced-Round MISTY. In Pfitzmann [Pfi01], pages 325–339.
Available from: http://dx.doi.org/10.1007/3-540-44987-6_20.
One citation on page 175.

[KW02] Lars R. Knudsen and David Wagner. Integral Cryptanalysis. In Daemen and Rijmen [DR02c], pages 112–127.
Available from: http://link.springer.de/link/service/series/0558/bibs/2365/23650112.htm.
2 citations on pages 81 and 174.

[KW13] Lars R. Knudsen and Huapeng Wu, editors. *Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers*, volume 7707 of *Lecture Notes in Computer Science*. Springer, 2013.
Available from: http://dx.doi.org/10.1007/978-3-642-35999-6.
3 citations on pages 246, 291, and 297.

[LAAZ11] Gregor Leander, Mohamed Ahmed Abdelraheem, Hoda AlKhzaimi, and Erik Zenner. A Cryptanalysis of PRINTcipher: The Invariant Subspace Attack. In Rogaway [Rog11], pages 206–221.
Available from: http://dx.doi.org/10.1007/978-3-642-22792-9_12,
DOI: doi:10.1007/978-3-642-22792-9.
One citation on page 211.

[Lai92] Xuejia Lai. *On the design and security of block ciphers*. PhD thesis, Technische Hochschule (Zurich), 1992. Published by Hartung-Gorre Verlag, Konstanz.
Available from: http://e-collection.library.ethz.ch/eserv/eth:38650/eth-38650-02.pdf?pid=eth:38650&dsID=eth-38650-02.pdf.
One citation on page 142.

[Lai94] Xuejia Lai. Higher order derivatives and differential cryptanalysis. In *Symposium on Communication, Coding and Cryptography in honor of James L. Massey on the occasion of his 60th birthday, Feb. 10-13, 1994, Monte-Verita, Ascona, Switzerland*, volume 276 of *Communications and Cryptography – The Springer International Series in Engineering and Computer Science*, pages 227–233, 1994.
One citation on page 80.

[LDKK08] Jiqiang Lu, Orr Dunkelman, Nathan Keller, and Jongsung Kim. New Impossible Differential Attacks on AES. In Chowdhury et al. [CRD08], pages 279–293.
Available from: https://doi.org/10.1007/978-3-540-89754-5_22,
DOI: doi:10.1007/978-3-540-89754-5_22.
3 citations on pages 182, 184, and 185.

[Lea10] Gregor Leander. Small Scale Variants Of The Block Cipher PRESENT. *IACR Cryptology ePrint Archive*, 2010:143, 2010.
Available from: http://eprint.iacr.org/2010/143.
One citation on page 44.

[Lee04] Pil Joong Lee, editor. *Advances in Cryptology - ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5-9, 2004, Proceedings*, volume 3329 of *Lecture Notes in Computer Science*. Springer, 2004.
2 citations on pages 246 and 292.

[LF04] Jérôme Lacan and Jérôme Fimes. Systematic MDS erasure codes based on Vandermonde matrices. *IEEE Communications Letters*, 8(9):570–572, 2004.
Available from: http://dx.doi.org/10.1109/LCOMM.2004.833807.
One citation on page 48.

[LGLL11] Ya Liu, Dawu Gu, Zhiqiang Liu, and Wei Li. Improved results on impossible differential cryptanalysis of reduced-round Camellia-192/256. *IACR Cryptology ePrint Archive*, 2011(671):15, 2011.
Available from: http://eprint.iacr.org/2011/671.pdf.
One citation on page 172.

[LGLL12] Ya Liu, Dawu Gu, Zhiqiang Liu, and Wei Li. Improved results on impossible differential cryptanalysis of reduced-round Camellia-192/256. *Journal of Systems and Software*, 85(11):2451–2458, 2012.
Available from: http://dx.doi.org/10.1016/j.jss.2012.05.051.
One citation on page 172.

[LGZL09] Zhiqiang Liu, Dawu Gu, Jing Zhang, and Wei Li. Differential-Multiple Linear Cryptanalysis. In Feng Bao, Moti Yung, Dongdai Lin, and Jiwu Jing, editors, *Inscrypt 2009*, volume 6151 of *Lecture Notes in Computer Science*, pages 35–49. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-16342-5_3.
One citation on page 94.

[LH94] Susan K. Langford and Martin E. Hellman. Differential-Linear Cryptanalysis. In Desmedt [Des94], pages 17–25.
Available from: http://dx.doi.org/10.1007/3-540-48658-5_3.
One citation on page 93.

[Lim99] Chae Hoon Lim. A Revised Version of Crypton - Crypton V1.0. In Knudsen [Knu99], pages 31–45.
Available from: http://dx.doi.org/10.1007/3-540-48519-8_3.
2 citations on pages 176 and 196.

[LJW13] Leibo Li, Keting Jia, and Xiaoyun Wang. Improved Meet-in-the-Middle Attacks on AES-192 and PRINCE. Cryptology ePrint Archive, Report 2013/573, 2013.
Available from: http://eprint.iacr.org/2013/573.
One citation on page 219.

[LK05] Chae Hoon Lim and Tymur Korkishko. mCrypton - A Lightweight Block Cipher for Security of Low-Cost RFID Tags and Sensors. In JooSeok Song, Taekyoung Kwon, and Moti Yung, editors, *WISA*, volume 3786 of *Lecture Notes in Computer Science*, pages 243–258. Springer, 2005.
Available from: http://dx.doi.org/10.1007/11604938_19.
2 citations on pages 196 and 236.

[LK07] Chu-Wee Lim and Khoongming Khoo. An Analysis of XSL Applied to BES. In Biryukov [Bir07], pages 242–253.
Available from: https://doi.org/10.1007/978-3-540-74619-5_16,
DOI: doi:10.1007/978-3-540-74619-5_16.
One citation on page 183.

[LK08] Jiqiang Lu and Jongsung Kim. Attacking 44 Rounds of the SHACAL-2 Block Cipher Using Related-Key Rectangle Cryptanalysis. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 91-A(9):2588–2596, 2008.
Available from: http://dx.doi.org/10.1093/ietfec/e91-a.9.2588.
One citation on page 188.

[LKKD06] Jiqiang Lu, Jongsung Kim, Nathan Keller, and Orr Dunkelman. Differential and Rectangle Attacks on Reduced-Round SHACAL-1. In Rana Barua and Tanja Lange, editors, *INDOCRYPT 2006*, volume 4329 of *Lecture Notes in Computer Science*, pages 17–31. Springer, 2006.
Available from: http://dx.doi.org/10.1007/11941378_3.
One citation on page 187.

[LLL13] Tanja Lange, Kristin Lauter, and Petr Lisonek, editors. *Selected Areas in Cryptography - 20th International Workshop, SAC 2013, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, Lecture Notes in Computer Science. Springer, 2013.
One citation on page 252.

[LLYC09] Young-Il Lim, Je-Hoon Lurl, Younggap You, and Kyoung-Ro Cho. Implementation of HIGHT cryptic circuit for RFID tag. *IEICE Electronics Express*, 6(4):180–186, 2009.
Available from: https://www.jstage.jst.go.jp/article/elex/6/4/6_4_180/_pdf.
One citation on page 235.

[LM90] Xuejia Lai and James L. Massey. A Proposal for a New Block Encryption Standard. In Damgård [Dam91], pages 389–404.
Available from: http://link.springer.de/link/service/series/0558/bibs/0473/04730389.htm.
One citation on page 142.

[LM92] Xuejia Lai and James L. Massey. Hash Function Based on Block Ciphers. In Rueppel [Rue93], pages 55–70.
Available from: http://dx.doi.org/10.1007/3-540-47555-9_5.
One citation on page 95.

[LMM91] Xuejia Lai, James L. Massey, and Sean Murphy. Markov Ciphers and Differential Cryptanalysis. In Davies [Dav91], pages 17–38.
Available from: http://dx.doi.org/10.1007/3-540-46416-6_2.
3 citations on pages 76, 77, and 142.

[LMR+09] Mario Lamberger, Florian Mendel, Christian Rechberger, Vincent Rijmen, and Martin Schläffer. Rebound Distinguishers: Results on the Full Whirlpool Compression Function. In Matsui [Mat09], pages 126–143.
Available from: http://dx.doi.org/10.1007/978-3-642-10366-7_8.
One citation on page 123.

[LMR+10] Mario Lamberger, Florian Mendel, Christian Rechberger, Vincent Rijmen, and Martin Schläffer. The Rebound Attack and Subspace Distinguishers: Application to Whirlpool. *IACR Cryptology ePrint Archive*, 2010:198, 2010.
Available from: http://eprint.iacr.org/2010/198.
One citation on page 123.

[LN83] Rudolf Lidl and Harald Niederreiter. *Finite Fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Reading, MA: Addison-Wesley, 1983.
One citation on page 57.

[LNP14] Virginie Lallemand and María Naya-Plasencia. Cryptanalysis of KLEIN. Preproceedings of FSE 2014, March 2014.
Available from: https://eprint.iacr.org/2014/090.pdf.
One citation on page 226.

[LP07] Gregor Leander and Axel Poschmann. On the Classification of 4 Bit S-Boxes. In Claude Carlet and Berk Sunar, editors, *WAIFI 2007*, volume 4547 of *Lecture Notes in Computer Science*, pages 159–176. Springer, 2007.
Available from: http://dx.doi.org/10.1007/978-3-540-73074-3_13.
One citation on page 217.

[LPPS07] Gregor Leander, Christof Paar, Axel Poschmann, and Kai Schramm. New Lightweight DES Variants. In Biryukov [Bir07], pages 196–210.
Available from: http://dx.doi.org/10.1007/978-3-540-74619-5_13,
DOI: doi:10.1007/978-3-540-74619-5.
2 citations on pages 133 and 235.

[LR86] Michael Luby and Charles Rackoff. Pseudo-random permutation generators and cryptographic composition. In ACM, editor, *Proceedings of the Eighteenth annual ACM Symposium on Theory of Computing, Berkeley, California, May 28–30, 1986*, pages 356–363, New York, NY 10036, USA, 1986. ACM Press. ACM order number 508860.
3 citations on pages 29, 68, and 70.

[LRW02] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable Block Ciphers. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2002.
Available from: http://dx.doi.org/10.1007/3-540-45708-9_3,
DOI: doi:10.1007/3-540-45708-9.
One citation on page 178.

[LRW11] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable Block Ciphers. *Journal of Cryptology*, 24(3):588–613, 2011.
Available from: http://dx.doi.org/10.1007/s00145-010-9073-y.
One citation on page 178.

[LSL11] Ruilin Li, Bing Sun, and Chao Li. Impossible differential cryptanalysis of SPN ciphers. *IET Information Security*, 5(2):111–120, 2011.
Available from: http://dx.doi.org/10.1049/iet-ifs.2010.0174.
One citation on page 89.

[Lu10] Jiqiang Lu. Differential Attack on Five Rounds of the SC2000 Block Cipher. Cryptology ePrint Archive, Report 2010/593, 2010.
Available from: http://eprint.iacr.org/2010/593.
One citation on page 193.

[Lu11] Jiqiang Lu. Differential Attack on Five Rounds of the SC2000 Block Cipher*. *Journal of Computer Science and Technology*, 26(4):722–731, 2011.
Available from: http://dx.doi.org/10.1007/s11390-011-1171-2.
One citation on page 193.

[Lu12] Jiqiang Lu. A Methodology for Differential-Linear Cryptanalysis and Its Applications - (Extended Abstract). In Canteaut [Can12], pages 69–89.
Available from: http://dx.doi.org/10.1007/978-3-642-34047-5_5.
One citation on page 94.

[Luc99] Stefan Lucks. On Security of the 128-Bit Block Cipher DEAL. In Knudsen [Knu99], pages 60–70.
Available from: http://dx.doi.org/10.1007/3-540-48519-8_5.
One citation on page 177.

[Luc00] Stefan Lucks. Attacking Seven Rounds of Rijndael under 192-bit and 256-bit Keys. In *Proceedings of the Third AES Candidate Conference*, pages 215–229, 2000.
One citation on page 81.

[LV00] Anatoly N. Lebedev and Alexey A. Volchkov. The NUSH Family, September 2000.
Available from: http://www.cosic.esat.kuleuven.be/nessie/workshop/submissions.html.
One citation on page 187.

[LW11] Dong Hoon Lurl and Xiaoyun Wang, editors. *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-25385-0.
2 citations on pages 254 and 284.

[LWZ11] Yanjun Li, Wenling Wu, and Lei Zhang. Improved Integral Attacks on Reduced-Round CLEFIA Block Cipher. In Souhwan Jung and Moti Yung, editors, *WISA*, volume 7115 of *Lecture Notes in Computer Science*, pages 28–39. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-27890-7_3.
One citation on page 203.

[Mac00] Alexis Warner Machado. The Nimbus Cipher, October 2000.
Available from: http://www.cosic.esat.kuleuven.be/nessie/workshop/submissions.html.
One citation on page 187.

[Mal11] Hamid Mala. Biclique Cryptanalysis of the Block Cipher SQUARE. *IACR Cryptology ePrint Archive*, 2011:500, 2011.
Available from: http://eprint.iacr.org/2011/500.
One citation on page 159.

[Mas93] James L. Massey. SAFER K-64: A Byte-Oriented Block-Ciphering Algorithm. In Anderson [And94], pages 1–17.
Available from: http://dx.doi.org/10.1007/3-540-58108-1_1.
One citation on page 148.

[Mas94] James L. Massey. SAFER K-64: One Year Later. In Preneel [Pre95], pages 212–241.
Available from: http://dx.doi.org/10.1007/3-540-60590-8_17.
2 citations on pages 148 and 154.

[Mat93] Mitsuru Matsui. Linear Cryptoanalysis Method for DES Cipher. In Helleseth [Hel94], pages 386–397.
Available from: http://dx.doi.org/10.1007/3-540-48285-7_33.
2 citations on pages 89 and 130.

[Mat97] Mitsuru Matsui. New Block Encryption Algorithm MISTY. In Biham [Bih97c], pages 54–68.
Available from: http://dx.doi.org/10.1007/BFb0052334.
2 citations on pages 170 and 174.

[Mat02] Mitsuru Matsui, editor. *Fast Software Encryption, 8th International Workshop, FSE 2001 Yokohama, Japan, April 2-4, 2001, Revised Papers*, volume 2355 of *Lecture Notes in Computer Science*. Springer, 2002.
3 citations on pages 250, 267, and 293.

[Mat09] Mitsuru Matsui, editor. *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, volume 5912 of *Lecture Notes in Computer Science*. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-10366-7.
3 citations on pages 246, 253, and 279.

[MC13a] James McLaughlin and John A. Clark. Filtered nonlinear cryptanalysis of reduced-round Serpent, and the Wrong-Key Randomization Hypothesis. Cryptology ePrint Archive, Report 2013/089, 2013.
http://eprint.iacr.org/.
One citation on page 79.

[MC13b] James McLaughlin and John A. Clark. Nonlinear cryptanalysis of reduced-round Serpent and metaheuristic search for S-box approximations. *IACR Cryptology ePrint Archive*, page 22, 2013.
Available from: http://eprint.iacr.org/2013/022.
One citation on page 92.

[MDRM10] Hamid Mala, Mohammad Dakhilalian, Vincent Rijmen, and Mahmoud Modarres-Hashemi. Improved Impossible Differential Cryptanalysis of 7-Round AES-128. In Gong and Gupta [GG10], pages 282–291.
Available from: https://doi.org/10.1007/978-3-642-17401-8_20,
DOI: doi:10.1007/978-3-642-17401-8_20.
One citation on page 184.

[MH81] Ralph C. Merkle and Martin E. Hellman. On the Security of Multiple Encryption. *Commun. ACM*, 24(7):465–467, 1981.
Available from: http://doi.acm.org/10.1145/358699.358718.
3 citations on pages 95, 100, and 132.

[Mil82] Frank Miller. *Telegraphic code to insure privacy and secrecy in the transmission of telegrams*. Charles. M. Cornwell, New York, 1882.
Available from: http://books.google.com/books?id=SPwdLwEACAAJ.
One citation on page 24.

[MIO89] Shoji Miyaguchi, M. Iwata, and K. Ohta. New 128-bit hash function. In *Proceeding of fourth international joint workshop on computer communications, Tokyo*, 1989.
One citation on page 67.

[MKK98] James L. Massey, Gurgen H. Khachatrian, and Melsik K. Kuregian. SAFER+, Cylink Corporation's Submission for the Advanced Encryption Standard. In *First AES Candidate Conference*. National Institute of Standard and Technology, 1998.
Available from: http://csrc.nist.gov/archive/aes/round1/conf1/saferpls-slides.pdf.
2 citations on pages 150 and 155.

[MKK00] James L. Massey, Gurgen H. Khachatrian, and Melsik K. Kuregian. Nomination of SAFER++ as candidate algorithm for the New European Schemes for Signatures, Integrity, and Encryption, September 2000.
Available from: http://web.eecs.utk.edu/~dunigan/cns06/SAFER.ps.
2 citations on pages 151 and 155.

[MMO85] Stephen M. Matyas, C.H. Meyer, and J. Oseas. Generating strong one-way functions with cryptographic algorithm. *IBM Techn. Disclosure Bull.*, 27(10A):5658–5659, 1985.
One citation on page 67.

[Mor96] Pat Morin. Provably Secure and Efficient Block Ciphers. In *Selected Areas in Cryptography 1996*, pages 30–37, 1996.
Available from: http://www.sacconference.org/SAC-history.html.
One citation on page 70.

[Mor17] Pawel Morawiecki. Practical attacks on the round-reduced PRINCE. *IET Information Security*, 11(3):146–151, 2017.
Available from: https://doi.org/10.1049/iet-ifs.2015.0432,
DOI: doi:10.1049/iet-ifs.2015.0432.
One citation on page 219.

[MPC04] Willi Meier, Enes Pasalic, and Claude Carlet. Algebraic Attacks and Decomposition of Boolean Functions. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 474–491. Springer, 2004.
Available from: http://dx.doi.org/10.1007/978-3-540-24676-3_28.
2 citations on pages 57 and 58.

[MPL+11] Amir Moradi, Axel Poschmann, San Ling, Christof Paar, and Huaxiong Wang. Pushing the Limits: A Very Compact and a Threshold Implementation of AES. In Paterson [Pat11], pages 69–88.
Available from: http://dx.doi.org/10.1007/978-3-642-20465-4_6.
One citation on page 237.

[MPRS11a] Lara Maines, Matteo Piva, Anna Rimoldi, and Massimiliano Sala. On the provable security of BEAR and LION schemes. *Applicable Algebra in Engineering, Communication and Computing*, 22(5-6):413–423, 2011.
Available from: http://dx.doi.org/10.1007/s00200-011-0159-z.
One citation on page 70.

[MPRS11b] Lara Maines, Matteo Piva, Anna Rimoldi, and Massimiliano Sala. On the provable security of BEAR and LION schemes. *CoRR*, abs/1105.0259, 2011.
Available from: http://arxiv.org/abs/1105.0259.
One citation on page 70.

[MRS09] Ben Morris, Phillip Rogaway, and Till Stegers. How to Encipher Messages on a Small Domain. In Halevi [Hal09], pages 286–302.
Available from: http://dx.doi.org/10.1007/978-3-642-03356-8_17.
One citation on page 30.

[MRST09] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In Dunkelman [Dun09], pages 260–276.
Available from: http://dx.doi.org/10.1007/978-3-642-03317-9_16.
One citation on page 123.

[MRST10] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. Rebound Attacks on the Reduced Grøstl Hash Function. In Pieprzyk [Pie10], pages 350–365.
Available from: http://dx.doi.org/10.1007/978-3-642-11925-5_24.
One citation on page 123.

[MS77] Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error correcting codes*, volume 16 of *North-Holland Mathematical Library*. Elsevier/North-Holland, 1977.
2 citations on pages 46 and 51.

[MS86a] Judy H. Moore and Gustavus J. Simmons. Cycle Structure of the Weak and Semi-Weak DES Keys. In *EUROCRYPT '86*, page 16, 1986.
One citation on page 146.

[MS86b] Judy H. Moore and Gustavus J. Simmons. Cycle Structures of the DES with Weak and Semi-Weak Keys. In Andrew M. Odlyzko, editor, *CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 9–32. Springer, 1986.
Available from: http://dx.doi.org/10.1007/3-540-47721-7_2.
One citation on page 146.

[MS89] Willi Meier and Othmar Staffelbach. Nonlinearity Criteria for Cryptographic Functions. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *EUROCRYPT '89*, volume 434 of *Lecture Notes in Computer Science*, pages 549–562. Springer, 1989.
Available from: http://dx.doi.org/10.1007/3-540-46885-4_53.
2 citations on pages 60 and 62.

[MS10] Stefan Mangard and François-Xavier Standaert, editors. *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*. Springer, 2010.
Available from: http://dx.doi.org/10.1007/978-3-642-15031-9.
4 citations on pages 251, 273, 286, and 288.

[MSQ07] François Macé, François-Xavier Standaert, and Jean-Jacques Quisquater. ASIC Implementations of the Block Cipher SEA for Constrained Applications. In *RFIDsec 2007, Workshop Record, Malaga, Spain, 2007*, pages 103–114, 2007.
Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.128.5397.
One citation on page 236.

[MT99] Mitsuru Matsui and Toshio Tokita. Cryptanalysis of a Reduced Version of the Block Cipher E2. In Knudsen [Knu99], pages 71–80.
Available from: http://dx.doi.org/10.1007/3-540-48519-8_6.
One citation on page 173.

[Mul03] Frédéric Muller. A New Attack against Khazad. In Chi-Sung Laih, editor, *ASIACRYPT '91*, volume 2894 of *Lecture Notes in Computer Science*, pages 347–358. Springer, 2003.
Available from: http://dx.doi.org/10.1007/978-3-540-40061-5_22.
One citation on page 190.

[Mur98] Sean Murphy. An Analysis of SAFER. *Journal of Cryptology*, 11(4):235–251, 1998.
Available from: http://dx.doi.org/10.1007/s001459900046.
One citation on page 150.

[MV91] Alfred Menezes and Scott A. Vanstone, editors. *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, volume 537 of *Lecture Notes in Computer Science*. Springer, 1991.
2 citations on pages 255 and 288.

[MV12a] Ali Miri and Serge Vaudenay, editors. *Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*, volume 7118 of *Lecture Notes in Computer Science*. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-28496-0.
2 citations on pages 250 and 274.

[MV12b] Aikaterini Mitrokotsa and Serge Vaudenay, editors. *Progress in Cryptology - AFRICACRYPT 2012 - 5th International Conference on Cryptology in Africa, Ifrance, Morocco, July 10-12, 2012. Proceedings*, volume 7374 of *Lecture Notes in Computer Science*. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-31410-0.
2 citations on pages 260 and 264.

[MY92] Mitsuru Matsui and Atsuhiro Yamagishi. A New Method for Known Plaintext Attack of FEAL Cipher. In Rueppel [Rue93], pages 81–91.
Available from: http://dx.doi.org/10.1007/3-540-47555-9_7.
One citation on page 89.

[MY00] Shiho Moriai and Yiqun Lisa Yin. Cryptanalysis of Twofish (II). Technical report, IEICE, 2000.
Available from: http://www.schneier.com/twofish-analysis-shiho.pdf.
One citation on page 163.

[NEC13] NEC. The block cipher CIPHERUNICORN, 2000-2013.
Available from: http://www.hnes.co.jp/solution/code/index02-e.html.
One citation on page 192.

[NIS98] NIST. SKIPJACK and KEA Algorithm Specifications, Version 2.0. Technical report, National Institute of Standards and Technology, May 1998.
Available from: http://csrc.nist.gov/groups/ST/toolkit/documents/skipjack/skipjack.pdf.
One citation on page 164.

[NIS12] NIST. FIPS 180-4: Secure Hash Standard. Technical report, National Institute of Standards & Technology, Gaithersburg, MD, United States, March 2012.
One citation on page 67.

[NK95] Kaisa Nyberg and Lars R. Knudsen. Provable Security Against a Differential Attack. *Journal of Cryptology*, 8(1):27–37, 1995.
Available from: http://dx.doi.org/10.1007/BF00204800.
One citation on page 89.

[NP11] María Naya-Plasencia. How to Improve Rebound Attacks. In Rogaway [Rog11], pages 188–205.
Available from: http://dx.doi.org/10.1007/978-3-642-22792-9_11,
DOI: doi:10.1007/978-3-642-22792-9.
2 citations on pages 99 and 123.

[NPSS10] Ivica Nikolic, Josef Pieprzyk, Przemyslaw Sokolowski, and Ron Steinfeld. Known and Chosen Key Differential Distinguishers for Block Ciphers. In Kyung Hyune Rhurl and DaeHun Nyang, editors, *ICISC 2010*, volume 6829 of *Lecture Notes in Computer Science*, pages 29–48. Springer, 2010.
Available from: http://dx.doi.org/10.1007/978-3-642-24209-0_3.
One citation on page 123.

[NPTV11] María Naya-Plasencia, Deniz Toz, and Kerem Varici. Rebound Attack on JH42. In Lurl and Wang [LW11], pages 252–269.
Available from: http://dx.doi.org/10.1007/978-3-642-25385-0_14.
One citation on page 123.

[NPV03] Jorge Nakahara, Jr., Bart Preneel, and Joos Vandewalle. A Note on Weak Keys of PES, IDEA, and Some Extended Variants. In Colin Boyd and Wenbo Mao, editors, *ISC 2003*, volume 2851 of *Lecture Notes in Computer Science*, pages 267–279. Springer, 2003.
Available from: http://dx.doi.org/10.1007/10958513_21.
One citation on page 144.

[NR99] Moni Naor and Omer Reingold. On the Construction of Pseudorandom Permutations: Luby-Rackoff Revisited. *Journal of Cryptology*, 12(1):29–66, 1999.
Available from: http://dx.doi.org/10.1007/PL00003817.
One citation on page 30.

[NSZW09] Jorge Nakahara, Pouyan Sepehrdad, Bingsheng Zhang, and Meiqin Wang. Linear (Hull) and Algebraic Cryptanalysis of the Block Cipher PRESENT. In Garay et al. [GMO09], pages 58–75.
Available from: http://dx.doi.org/10.1007/978-3-642-10433-6_5.
One citation on page 205.

[NTT01] NTT (Nippon Telegraph and Telephone Corporation). Announcement of Royalty-frurl Licenses for Essential Patents of NTT Encryption and Digital Signature Algorithms, April 2001.
Available from: http://www.ntt.co.jp/news/news01e/0104/010417.html.
One citation on page 173.

[NTT06] NTT (Nippon Telegraph and Telephone Corporation). The Open Source Community OpenSSL Project Adopts the Next Generation International Standard Cipher "Camellia" Developed in Japan, November 2006.
Available from: http://www.ntt.co.jp/news/news06e/0611/061108a.html.
One citation on page 173.

[NWW11] Phuong Ha Nguyen, Hongjun Wu, and Huaxiong Wang. Improving the Algorithm 2 in Multidimensional Linear Cryptanalysis. In Parampalli and Hawkes [PH11], pages 61–74.
Available from: http://dx.doi.org/10.1007/978-3-642-22497-3_5.
One citation on page 169.

[NWW13a] Ivica Nikolic, Lei Wang, and Shuang Wu. Cryptanalysis of Round-Reduced LED, March 2013. To appear in: Proceedings of FSE 2013, the 20th International Workshop on Fast Software Encryption.
One citation on page 116.

[NWW13b] Ivica Nikolic, Lei Wang, and Shuang Wu. The Parallel-Cut Meet-In-The-Middle Attack. *IACR Cryptology ePrint Archive*, 2013:530, 2013.
Available from: http://eprint.iacr.org/2013/530.
2 citations on pages 104 and 226.

[NWWL10] Phuong Ha Nguyen, Lei Wei, Huaxiong Wang, and San Ling. On Multidimensional Linear Cryptanalysis. In Ron Steinfeld and Philip Hawkes, editors, *ACISP*, volume 6168 of *Lecture Notes in Computer Science*, pages 37–52. Springer, 2010.
Available from: http://dx.doi.org/10.1007/978-3-642-14081-5_3.
One citation on page 91.

[Nyb90] Kaisa Nyberg. Constructions of Bent Functions and Difference Sets. In Damgård [Dam91], pages 151–160.
Available from: http://link.springer.de/link/service/series/0558/bibs/0473/04730151.htm.
One citation on page 60.

[Nyb91] Kaisa Nyberg. Perfect Nonlinear S-Boxes. In Davies [Dav91], pages 378–386.
Available from: http://dx.doi.org/10.1007/3-540-46416-6_32.
2 citations on pages 59 and 146.

[Nyb93] Kaisa Nyberg. Differentially Uniform Mappings for Cryptography. In Helleseth [Hel94], pages 55–64.
Available from: http://dx.doi.org/10.1007/3-540-48285-7_6.
2 citations on pages 61 and 181.

[Nyb94] Kaisa Nyberg. Linear Approximation of Block Ciphers. In Santis [San95], pages 439–444.
Available from: http://dx.doi.org/10.1007/BFb0053460.
One citation on page 90.

[Nyb96] Kaisa Nyberg. Generalized Feistel Networks. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*, pages 91–104. Springer, 1996.
Available from: http://dx.doi.org/10.1007/BFb0034838.
3 citations on pages 32, 52, and 54.

[Nyb98] Kaisa Nyberg, editor. *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*. Springer, 1998.
2 citations on pages 253 and 268.

[Nyb08] Kaisa Nyberg, editor. *Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers*, volume 5086 of *Lecture Notes in Computer Science*. Springer, 2008.
5 citations on pages 257, 260, 263, 294, and 298.

[OBSC10] Dag Arne Osvik, Joppe W. Bos, Deian Stefan, and David Canright. Fast Software AES Encryption. In Hong and Iwata [HI10], pages 75–93.
Available from: http://dx.doi.org/10.1007/978-3-642-13858-4_5.
2 citations on pages 238 and 239.

[O'C11a] Derek O'Connor. A Historical Note on Shuffle Algorithms. Published by the author, 2011.
Available from: https://www.academia.edu/1205620/.
One citation on page 67.

[O'C11b] Derek O'Connor. A Historical Note on the Fisher-Yates and Durstenfeld Shuffle Algorithms. Published by the author, 2011.
Available from: http://www.scribd.com/doc/65464763/.
One citation on page 67.

[Ohk09] Kenji Ohkuma. Weak Keys of Reduced-Round PRESENT for Linear Cryptanalysis. In Michael J. Jacobson Jr., Vincent Rijmen, and Reihaneh Safavi-Naini, editors, *Selected Areas in Cryptography 2009*, volume 5867 of *Lecture Notes in Computer Science*, pages 249–265. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-05445-7_16.
One citation on page 205.

[OMSK00] Kenji Ohkuma, Hirofumi Muratani, Fumihiko Sano, and Shin-Ichi Kawamura. The Block Cipher Hierocrypt. In Stinson and Tavares [ST01], pages 72–88.
Available from: http://dx.doi.org/10.1007/3-540-44983-3_6.
4 citations on pages 46, 174, 187, and 192.

[öVTK09] Onur özen, Kerem Varici, Cihangir Tezcan, and Çelebi Kocair. Lightweight Block Ciphers Revisited: Cryptanalysis of Reduced Round PRESENT and HIGHT. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP 2009*, volume 5594 of *Lecture Notes in Computer Science*, pages 90–107. Springer, 2009.
Available from: http://dx.doi.org/10.1007/978-3-642-02620-1_7.
2 citations on pages 201 and 205.

[PA93] Andreas Pfitzmann and Ralf Aßmann. More efficient software implementations of (generalized) DES. *Computers & Security*, 12(5):477–500, 1993. Preprint (1990) available as: Institut für Rechnerentwurf und Fehlertoleranz, Universität Karlsruhe, Interner Bericht 18/90, http://www.semper.org/sirene/publ/PfAss_90DES_IB.ps.gz.
Available from: http://dx.doi.org/10.1016/0167-4048(93)90069-H.
One citation on page 195.

[Pan01] Victor Y. Pan. *Structured Matrices and Polynomials: Unified Superfast Algorithms*. Birkhäuser, Boston, and Springer-Verlag New York, 2001.
One citation on page 48.

[Pat11] Kenneth G. Paterson, editor. *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-20465-4.
One citation on page 282.

[Pfi01] Birgit Pfitzmann, editor. *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*. Springer, 2001.
2 citations on pages 250 and 277.

[PGV91] Bart Preneel, René Govaerts, and Joos Vandewalle. Boolean Functions Satisfying Higher Order Propagation Criteria. In Davies [Dav91], pages 141–152.
Available from: http://dx.doi.org/10.1007/3-540-46416-6_12.
One citation on page 62.

[PGV93] Bart Preneel, René Govaerts, and Joos Vandewalle. Hash Functions Based on Block Ciphers: A Synthetic Approach. In Stinson [Sti94], pages 368–378.
Available from: http://dx.doi.org/10.1007/3-540-48329-2_31.
One citation on page 67.

[PH11] Udaya Parampalli and Philip Hawkes, editors. *Information Security and Privacy - 16th Australasian Conference, ACISP 2011, Melbourne, Australia, July 11-13, 2011. Proceedings*, volume 6812 of *Lecture Notes in Computer Science*. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-22497-3.
3 citations on pages 284, 296, and 297.

[Pie10] Josef Pieprzyk, editor. *Topics in Cryptology - CT-RSA 2010, The Cryptographers' Track at the RSA Conference 2010, San Francisco, CA, USA, March 1-5, 2010. Proceedings*, volume 5985 of *Lecture Notes in Computer Science*. Springer, 2010.
Available from: http://dx.doi.org/10.1007/978-3-642-11925-5.
2 citations on pages 258 and 282.

[PJ12] David Pointcheval and Thomas Johansson, editors. *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-29011-4.
2 citations on pages 262 and 273.

[PLL+90] Bart Preneel, Werner Van Leekwijck, Luc Van Linden, René Govaerts, and Joos Vandewalle. Propagation Characteristics of Boolean Functions. In Damgård [Dam91], pages 161–173.
Available from: http://link.springer.de/link/service/series/0558/bibs/0473/04730161.htm.
One citation on page 62.

[PLW10] Axel Poschmann, San Ling, and Huaxiong Wang. 256 Bit Standardized Crypto for 650 GE - GOST Revisited. In Mangard and Standaert [MS10], pages 219–233.
Available from: http://dx.doi.org/10.1007/978-3-642-15031-9_15.
2 citations on pages 140 and 235.

[Por98] Thomas Pornin. Optimal Resistance Against the Davies and Murphy Attack. In Kazuo Ohta and Dingyi Pei, editors, *ASIACRYPT '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 148–159. Springer, 1998.
Available from: http://dx.doi.org/10.1007/3-540-49649-1_13.
One citation on page 120.

[Pos09] Axel Poschmann. *Lightweight Cryptography - Cryptographic Engineering for a Pervasive World*. PhD thesis, Ruhr University Bochum, Bochum, Germany, 2009.
One citation on page 236.

[PQ03] Gilles Piret and Jean-Jacques Quisquater. Integral Cryptanalysis on reduced-round Safer++. *IACR Cryptology ePrint Archive*, 2003:33, 2003.
Available from: http://eprint.iacr.org/2003/033.
One citation on page 154.

[PR13] Emmanuel Prouff and Matthieu Rivain. Masking against Side-Channel Attacks: A Formal Security Proof. In Johansson and Nguyen [JN13], pages 142–159.
Available from: http://dx.doi.org/10.1007/978-3-642-38348-9_9,
DOI: doi:10.1007/978-3-642-38348-9.
One citation on page 228.

[PRC12] Gilles Piret, Thomas Roche, and Claude Carlet. PICARO - A Block Cipher Allowing Efficient Higher-Order Side-Channel Resistance. In Feng Bao, Pierangela Samarati, and Jianying Zhou, editors, *ACNS*, volume 7341 of *Lecture Notes in Computer Science*, pages 311–328. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-31284-7_19.
2 citations on pages 228 and 229.

[Pre93] Bart Preneel. *Analysis and Design of Cryptographic Hash Functions*. PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium, 1993.
One citation on page 62.

[Pre95] Bart Preneel, editor. *Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings*, volume 1008 of *Lecture Notes in Computer Science*. Springer, 1995.
7 citations on pages 256, 274, 275, 280, 288, 294, and 296.

[Pre00] Bart Preneel, editor. *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*. Springer, 2000.
2 citations on pages 256 and 258.

[PSN95] Josef Pieprzyk and Reihaneh Safavi-Naini, editors. *Advances in Cryptology - ASIACRYPT '94, 4th International Conference on the Theory and Applications of Cryptology, Wollongong, Australia, November 28 - December 1, 1994, Proceedings*, volume 917 of *Lecture Notes in Computer Science*. Springer, 1995.
2 citations on pages 248 and 252.

[PT11] Bart Preneel and Tsuyoshi Takagi, editors. *Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings*, volume 6917 of *Lecture Notes in Computer Science*. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-23951-9.
2 citations on pages 267 and 291.

[Rab10] Tal Rabin, editor. *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*. Springer, 2010.
Available from: http://dx.doi.org/10.1007/978-3-642-14623-7.
2 citations on pages 262 and 269.

[RDJ+01] Atri Rudra, Pradeep K. Dubey, Charanjit S. Jutla, Vijay Kumar, Josyula R. Rao, and Pankaj Rohatgi. Efficient Rijndael Encryption Implementation with Composite Field Arithmetic. In Çetin Kaya Koç, David Naccache, and Christof Paar, editors, *CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 171–184. Springer, 2001.
Available from: http://dx.doi.org/10.1007/3-540-44709-1_16.
One citation on page 183.

[RDP+96] Vincent Rijmen, Joan Daemen, Bart Preneel, Antoon Bosselaers, and Erik De Win. The Cipher SHARK. In Gollmann [Gol96], pages 99–111.
Available from: http://dx.doi.org/10.1007/3-540-60865-6_47.
2 citations on pages 46 and 189.

[Rec09] Christian Rechberger. *Cryptanalysis of Hash Functions*. PhD thesis, Graz University of Technology, Graz, Austria, January 2009.
One citation on page 74.

[Rij97] Vincent Rijmen. *Cryptanalysis and Design of Iterated Block Ciphers*. PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium, 1997.
Available from: https://www.cosic.esat.kuleuven.be/publications/thesis-4.ps.
One citation on page 156.

[Riv90] Ronald L. Rivest. The MD4 Message Digest Algorithm. In Menezes and Vanstone [MV91], pages 303–311.
Available from: http://dx.doi.org/10.1007/3-540-38424-3_22.
One citation on page 67.

[Riv94] Ronald L. Rivest. The RC5 Encryption Algorithm. In Preneel [Pre95], pages 86–96.
Available from: http://dx.doi.org/10.1007/3-540-60590-8_7.
One citation on page 156.

[RM04] Bimal K. Roy and Willi Meier, editors. *Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers*, volume 3017 of *Lecture Notes in Computer Science*. Springer, 2004.
2 citations on pages 273 and 292.

[Rob06] Matthew J. B. Robshaw, editor. *Fast Software Encryption, 13th International Workshop, FSE 2006, Graz, Austria, March 15-17, 2006, Revised Selected Papers*, volume 4047 of *Lecture Notes in Computer Science*. Springer, 2006.
2 citations on pages 269 and 292.

[Rog11] Phillip Rogaway, editor. *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*. Springer, 2011.
Available from: https://doi.org/10.1007/978-3-642-22792-9,
DOI: doi:10.1007/978-3-642-22792-9.
4 citations on pages 250, 267, 277, and 284.

[Rot76] Oscar S. Rothaus. On "Bent" Functions. *J. Comb. Theory, Ser. A*, 20(3):300–305, 1976.
Available from: http://dx.doi.org/10.1016/0097-3165(76)90024-8.
One citation on page 59.

[Roy05] Bimal K. Roy, editor. *Advances in Cryptology - ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4-8, 2005, Proceedings*, volume 3788 of *Lecture Notes in Computer Science*. Springer, 2005.
Available from: https://doi.org/10.1007/11593447,
DOI: doi:10.1007/11593447.
2 citations on pages 258 and 274.

[RP10] Matthieu Rivain and Emmanuel Prouff. Provably Secure Higher-Order Masking of AES. In Mangard and Standaert [MS10], pages 413–427.
Available from: http://dx.doi.org/10.1007/978-3-642-15031-9_28.
One citation on page 228.

[RPLP08] Carsten Rolfes, Axel Poschmann, Gregor Leander, and Christof Paar. Ultra-Lightweight Implementations for Smart Devices - Security for 1000 Gate Equivalents. In Gilles Grimaud and François-Xavier Standaert, editors, *CARDIS 2008*, volume 5189 of *Lecture Notes in Computer Science*, pages 89–103. Springer, 2008.
Available from: http://dx.doi.org/10.1007/978-3-540-85893-5_7.
One citation on page 236.

[RPW97] Vincent Rijmen, Bart Preneel, and Erik De Win. On Weaknesses of Non-surjective Round Functions. *Designs, Codes and Cryptography*, 12(3):253–266, 1997.
Available from: http://dx.doi.org/10.1023/A:1008224928678.
One citation on page 222.

[RR16a] Shahram Rasoolzadeh and Håvard Raddum. Cryptanalysis of 6-round PRINCE using 2 Known Plaintexts. *IACR Cryptology ePrint Archive*, 2016:132, 2016. Presented at ArcticCrypt 2016.
Available from: http://eprint.iacr.org/2016/132.
One citation on page 219.

[RR16b] Shahram Rasoolzadeh and Håvard Raddum. Cryptanalysis of PRINCE with Minimal Data. In David Pointcheval, Abderrahmane Nitaj, and Tajjeeddine Rachidi, editors, *Progress in Cryptology - AFRICACRYPT 2016 - 8th International Conference on Cryptology in Africa, Fes, Morocco, April 13-15, 2016, Proceedings*, volume 9646 of *Lecture Notes in Computer Science*, pages 109–126. Springer, 2016.
Available from: https://doi.org/10.1007/978-3-319-31517-1_6,
DOI: doi:10.1007/978-3-319-31517-1_6.
One citation on page 219.

[RR16c] Shahram Rasoolzadeh and Håvard Raddum. Faster Key Recovery Attack on Round-Reduced PRINCE. In Andrey Bogdanov, editor, *Lightweight Cryptography for Security and Privacy - 5th International Workshop, LightSec 2016, Aksaray, Turkey, September 21-22, 2016, Revised Selected Papers*, volume 10098 of *Lecture Notes in Computer Science*, pages 3–17. Springer, 2016.
Available from: https://doi.org/10.1007/978-3-319-55714-4_1,
DOI: doi:10.1007/978-3-319-55714-4_1.
One citation on page 219.

[RRY00] Ronald L. Rivest, Matthew J. B. Robshaw, and Yiqun Lisa Yin. RC6 as the AES. In *Proceedings of the Third AES Candidate Conference*, pages 337–342, 2000.
One citation on page 165.

[RS06] Håvard Raddum and Igor Semaev. New Technique for Solving Sparse Equation Systems. *IACR Cryptology ePrint Archive*, 2006:475, 2006.
Available from: http://eprint.iacr.org/2006/475.
One citation on page 121.

[RSQL04] Gaël Rouvroy, François-Xavier Standaert, Jean-Jacques Quisquater, and Jean-Didier Legat. Compact and Efficient Encryption/Decryption Module for FPGA Implementation of the AES Rijndael Very Well Suited for Small Embedded Applications. In *ITCC (2)*, pages 583–587. IEEE Computer Society, 2004. Preprint available from http://dial.academielouvain.be/downloader/downloader.py?pid=boreal:81780&datastream=PDF_01.
Available from: http://doi.ieeecomputersociety.org/10.1109/ITCC.2004.1286716.
One citation on page 183.

[Rue93] Rainer A. Rueppel, editor. *Advances in Cryptology - EUROCRYPT '92, Workshop on the Theory and Application of of Cryptographic Techniques, Balatonfüred, Hungary, May 24-28, 1992, Proceedings*, volume 658 of *Lecture Notes in Computer Science*. Springer, 1993.
2 citations on pages 278 and 283.

[Saa98a] Markku-Juhani Olavi Saarinen. Chosen ciphertext attack against the Block Tea algorithm, September 1998. Post to Usenet Newsgroup `sci.crypt.research`.
Available from: http://groups.google.com/group/sci.crypt.research/msg/f52a533d1e2fa15e.
One citation on page 162.

[Saa98b] Markku-Juhani Olavi Saarinen. A chosen key attack against the secret S-Boxes of GOST, Aug 1998. Available from the author. Oldest reference on the Cryptography Digest mailing list: http://www.mail-archive.com/cryptography-digest@senator-bedfellow.mit.edu/msg00934.html, 16 June 1999.
One citation on page 112.

[Saa03] Markku-Juhani Olavi Saarinen. Cryptanalysis of Block Ciphers Based on SHA-1 and MD5. In Johansson [Joh03], pages 36–44.
Available from: http://dx.doi.org/10.1007/978-3-540-39887-5_4.
One citation on page 187.

[San95] Alfredo De Santis, editor. *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings*, volume 950 of *Lecture Notes in Computer Science*. Springer 1994, 1995.
3 citations on pages 260, 285, and 293.

[SB81] Ingrid Schaumüller-Bichl. *Zur Analyse des Data Encryption Standard und Synthese Verwandter Chiffriersysteme*. PhD thesis, Linz University, Linz, Austria, May 1981.
One citation on page 133.

[Sch93] Bruce Schneier. Description of a New Variable-Length Key, 64-bit Block Cipher (Blowfish). In Anderson [And94], pages 191–204.
Available from: http://dx.doi.org/10.1007/3-540-58108-1_24.
One citation on page 155.

[Sch96] Bruce Schneier. *Applied cryptography - protocols, algorithms, and source code in C (2nd ed.)*. Wiley, 1996.
4 citations on pages 68, 134, 136, and 151.

[Sch98] Bruce Schneier. A Self-Study Course in Block-Cipher Cryptanalysis, 1998.
Available from: http://www.schneier.com/.
One citation on page 24.

[Sch01] Bruce Schneier, editor. *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings*, volume 1978 of *Lecture Notes in Computer Science*. Springer, 2001.
5 citations on pages 265, 266, 271, 273, and 297.

[Sch02] Bruce Schneier. Crypto-Gram Newsletter. Online Newsletter, May 2002.
Available from: https://www.schneier.com/crypto-gram-0205.html.
One citation on page 21.

[Sco85] Robert Scott. Wide Open Encryption Design Offers Flexible Implementations. *Cryptologia*, 9(1):75–90, 1985.
One citation on page 134.

[SDM13] Mohsen Shakiba, Mohammad Dakhilalian, and Hamid Mala. Non-isomorphic Biclique Cryptanalysis and Its Application to Full-Round mCrypton. *IACR Cryptology ePrint Archive*, 2013:141, 2013.
Available from: http://eprint.iacr.org/2013/141.
One citation on page 197.

[SDMO12] Mahdi Sajadieh, Mohammad Dakhilalian, Hamid Mala, and Behnaz Omoomi. On construction of involutory MDS matrices from Vandermonde Matrices in GF(2^q). *Designs, Codes and Cryptography*, 64(3):287–308, 2012.
Available from: http://dx.doi.org/10.1007/s10623-011-9578-x.
One citation on page 48.

[SDMS12] Mahdi Sajadieh, Mohammad Dakhilalian, Hamid Mala, and Pouyan Sepehrdad. Recursive Diffusion Layers for Block Ciphers and Hash Functions. In Canteaut [Can12], pages 385–401.
Available from: http://dx.doi.org/10.1007/978-3-642-34047-5_22.
2 citations on pages 50 and 51.

[Sel08] Ali Aydin Selçuk. On Probability of Success in Linear and Differential Cryptanalysis. *Journal of Cryptology*, 21(1):131–147, 2008.
Available from: http://dx.doi.org/10.1007/s00145-007-9013-7.
One citation on page 90.

[Sha49] Claude Shannon. Communication Theory of Secrecy Systems. *The Bell System Technical Journal*, 28(4):656–715, October 1949. A footnote on the initial page says: "The material in this paper appeared in a confidential report, 'A Mathematical Theory of Cryptography', dated Sept. 1, 1946, which has now been declassified.".
Available from: http://pages.cs.wisc.edu/~rist/642-fall-2012/shannon-secrecy.pdf.
2 citations on pages 23 and 24.

[Sha85] Adi Shamir. On the Security of DES. In Williams [Wil86], pages 280–281.
Available from: http://dx.doi.org/10.1007/3-540-39799-X_22.
One citation on page 89.

[SHAS07] Takeshi Sugawara, Naofumi Homma, Takafumi Aoki, and Akashi Satoh. ASIC Performance Comparison for the ISO Standard Block Ciphers. In *Proceedings of the 2nd Joint Workshop on Information Security, JWIS 2007*, August 2007.
Available from: http://www.aoki.ecei.tohoku.ac.jp/crypto/pdf/JWIS2007_sugawara.pdf.
One citation on page 237.

[Shi95] Ken Shirriff. Differential Cryptanalysis of Madryga, October 1995.
Available from: http://files.righto.com/papers/madryga.ps.
One citation on page 134.

[Shi02] Taizo Shirai. Differential, linear, boomerang and rectangle cryptanalysis of reduced-round Camellia. In *Proceedings of the Third NESSIE Workshop*. NESSIE, November 2002.
One citation on page 172.

[Shi10] Kyoji Shibutani. On the Diffusion of Generalized Feistel Structures Regarding Differential and Linear Cryptanalysis. In Biryukov et al. [BGS11], pages 211–228.
Available from: http://dx.doi.org/10.1007/978-3-642-19574-7_15.
One citation on page 54.

[Sid71] Vladimir Michilovich Sidelnikov. On the mutual correlation of sequences. *Soviet Math. Dokl*, 12:197–201, 1971.
One citation on page 59.

[SIH+11] Kyoji Shibutani, Takanori Isobe, Harunaga Hiwatari, Atsushi Mitsuda, Toru Akishita, and Taizo Shirai. Piccolo: An Ultra-Lightweight Blockcipher. In Preneel and Takagi [PT11], pages 342–357.
Available from: http://dx.doi.org/10.1007/978-3-642-23951-9_23.
2 citations on pages 226 and 236.

[SKW+98a] Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson. AES Performance Comparison. In *Second AES Candidate Conference*. National Institute of Standard and Technology, 1998.
Available from: http://csrc.nist.gov/archive/aes/round1/conf2/Schneier.pdf.
One citation on page 148.

[SKW+98b] Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson. Twofish, A Block Encryption Algorithm. In *First AES Candidate Conference*. National Institute of Standard and Technology, 1998.
Available from: http://csrc.nist.gov/archive/aes/round1/conf1/twofish-slides.pdf.
One citation on page 162.

[SM87] Akihiro Shimizu and Shoji Miyaguchi. Fast Data Encipherment Algorithm FEAL. In Chaum and Price [CP88], pages 267–278.
Available from: http://dx.doi.org/10.1007/3-540-39118-5_24.
One citation on page 137.

[SM10] Tomoyasu Suzaki and Kazuhiko Minematsu. Improving the Generalized Feistel. In Hong and Iwata [HI10], pages 19–39.
Available from: http://dx.doi.org/10.1007/978-3-642-13858-4_2.
4 citations on pages 52, 53, 54, and 213.

[SMMK12] Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. TWINE: A Lightweight Block Cipher for Multiple Platforms. In Knudsen and Wu [KW13], pages 339–354.
Available from: http://dx.doi.org/10.1007/978-3-642-35999-6_22.
3 citations on pages 213, 235, and 236.

[SMVP11] Gautham Sekar, Nicky Mouha, Vesselin Velichkov, and Bart Preneel. Meet-in-the-Middle Attacks on Reduced-Round XTEA. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *Lecture Notes in Computer Science*, pages 250–267. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-19074-2_17.
2 citations on pages 161 and 162.

[SON07] SONY Corporation. Sony Develops "CLEFIA" – New Block Cipher Algorithm Based on State-of-the-art Design Technologies, March 2007.
Available from: http://www.sony.net/SonyInfo/News/Press/200703/07-028E/index.html.
One citation on page 201.

[SON10] SONY Corporation. The 128-bit blockcipher CLEFIA - self evaluations report. Technical report, SONY, 2010.
Available from: http://www.sony.net/Products/cryptography/clefia/download/index.html.
One citation on page 201.

[Sor84] Arthur Sorkin. Lucifer, A Cryptographic Algorithm. *Cryptologia*, 8(1):22–41, 1984. An addendum can be found in Cryptologia 8(3):260–261, and a preprint at http://fuseki.com/lucifer.pdf.
Available from: http://dx.doi.org/10.1080/0161-118491858746.
2 citations on pages 28 and 126.

[SP04] Taizo Shirai and Bart Preneel. On Feistel Ciphers Using Optimal Diffusion Mappings Across Multiple Rounds. In Lee [Lee04], pages 1–15.
Available from: http://dx.doi.org/10.1007/978-3-540-30539-2_1.
One citation on page 54.

[SPGQ06] François-Xavier Standaert, Gilles Piret, Neil Gershenfeld, and Jean-Jacques Quisquater. SEA: A Scalable Encryption Algorithm for Small Embedded Applications. In Josep Domingo-Ferrer, Joachim Posegga, and Daniel Schreckling, editors, *CARDIS 2006*, volume 3928 of *Lecture Notes in Computer Science*, pages 222–236. Springer, 2006.
Available from: http://dx.doi.org/10.1007/11733447_16.
4 citations on pages 197, 198, 199, and 211.

[SPR+04] François-Xavier Standaert, Gilles Piret, Gaël Rouvroy, Jean-Jacques Quisquater, and Jean-Didier Legat. ICEBERG : An Involutional Cipher Efficient for Block Encryption in Reconfigurable Hardware. In Roy and Meier [RM04], pages 279–299.
Available from: http://dx.doi.org/10.1007/978-3-540-25937-4_18.
2 citations on pages 195 and 199.

[SRQL02] François-Xavier Standaert, Gael Rouvroy, Jean-Jacques Quisquater, and Jean-Didier Legat. Efficient FPGA implementations of block ciphers KHAZAD and MISTY1. In *Proceedings of the Third NESSIE Workshop*. NESSIE, November 2002.
3 citations on pages 195, 234, and 235.

[SRQL03a] François-Xavier Standaert, Gaël Rouvroy, Jean-Jacques Quisquater, and Jean-Didier Legat. Efficient Implementation of Rijndael Encryption in Reconfigurable Hardware: Improvements and Design Tradeoffs. In Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors, *CHES*, volume 2779 of *Lecture Notes in Computer Science*, pages 334–350. Springer, 2003.
Available from: http://dx.doi.org/10.1007/978-3-540-45238-6_27.
One citation on page 195.

[SRQL03b] François-Xavier Standaert, Gaël Rouvroy, Jean-Jacques Quisquater, and Jean-Didier Legat. A methodology to implement block ciphers in reconfigurable hardware and its application to fast and compact AES RIJNDAEL. In *FPGA*, pages 216–224, 2003.
Available from: http://doi.acm.org/10.1145/611817.611849.
One citation on page 195.

[SS04] Taizo Shirai and Kyoji Shibutani. Improving Immunity of Feistel Ciphers against Differential Cryptanalysis by Using Multiple MDS Matrices. In Roy and Meier [RM04], pages 260–278.
Available from: http://dx.doi.org/10.1007/978-3-540-25937-4_17.
2 citations on pages 54 and 202.

[SS06] Taizo Shirai and Kyoji Shibutani. On Feistel Structures Using a Diffusion Switching Mechanism. In Robshaw [Rob06], pages 41–56.
Available from: http://dx.doi.org/10.1007/11799313_4.
One citation on page 54.

[SSA+07] Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-Bit blockcipher CLEFIA. In Biryukov [Bir07], pages 181–195.
Available from: https://doi.org/10.1007/978-3-540-74619-5,
DOI: doi:10.1007/978-3-540-74619-5.
2 citations on pages 201 and 237.

[ST01] Douglas R. Stinson and Stafford E. Tavares, editors. *Selected Areas in Cryptography, 7th Annual International Workshop, SAC 2000, Waterloo, Ontario, Canada, August 14-15, 2000, Proceedings*, volume 2012 of *Lecture Notes in Computer Science*. Springer, 2001.
3 citations on pages 246, 267, and 285.

[Sta14] François-Xavier Standaert. Private communication, May 2014. In an email to the author, François-Xavier Standaert wrote: *None of our designs is patented BTW. I don't plan to change this in the future ;)*.
4 citations on pages 196, 200, 230, and 231.

[Ste99] Jacques Stern, editor. *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*. Springer, 1999.
2 citations on pages 248 and 295.

[Ste05] Michael Steil. 17 Mistakes Microsoft Made in the Xbox Security System. In *22nd Chaos Communication Congress (http://events.ccc.de/congress/2005/)*. Chaos Computer Club, December 2005.
Available from: http://events.ccc.de/congress/2005/fahrplan/attachments/591-paper_xbox.pdf.
One citation on page 160.

[Sti94] Douglas R. Stinson, editor. *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*. Springer, 1994.
2 citations on pages 248 and 286.

[STK02] Takeshi Shimoyama, Masahiko Takenaka, and Takeshi Koshiba. Multiple Linear Cryptanalysis of a Reduced Round RC6. In Daemen and Rijmen [DR02c], pages 76–88.
Available from: http://dx.doi.org/10.1007/3-540-45661-9_6.
One citation on page 167.

[Sug96] Makoto Sugita. Pseudorandomness of a Block Cipher MISTY. Technical Report 167, IEICE, 1996.
Available from: http://ci.nii.ac.jp/naid/110003296958.
One citation on page 33.

[SV94] Claus-Peter Schnorr and Serge Vaudenay. Black Box Cryptoanalysis of Hash Networks Based on Multipermutations. In Santis [San95], pages 47–57.
Available from: http://dx.doi.org/10.1007/BFb0053423.
One citation on page 45.

[SV98] Jacques Stern and Serge Vaudenay. CS-Cipher. In Vaudenay [Vau98a], pages 189–205.
Available from: http://dx.doi.org/10.1007/3-540-69710-1_13.
One citation on page 187.

[Syl67] James Joseph Sylvester. Thoughts on inverse orthogonal matrices, simultaneous sign successions, and tessellated pavements in two or more colours, with applications to Newton's rule, ornamental tile-work, and the theory of numbers. *Philosophical Magazine*, 34:461–475, 1867.
One citation on page 48.

[SYY+01] Takeshi Shimoyama, Hitoshi Yanami, Kazuhiro Yokoyama, Masahiko Takenaka, Kouichi Itoh, Jun Yajima, Naoya Torii, and Hidema Tanaka. The Block Cipher SC2000. In Matsui [Mat02], pages 312–327.
Available from: http://dx.doi.org/10.1007/3-540-45473-X_26.
2 citations on pages 187 and 192.

[SZ97] Kouichi Sakurai and Yuliang Zheng. On Non-Pseudorandomness from Block Ciphers with Provable Immunity Against Linear Cryptanalysis. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E80(1):19–24, January 1997.
Available from: http://coitweb.uncc.edu/~yzheng/publications/files/ieice96.pdf.
One citation on page 33.

[TCG91] Anne Tardy-Corfdir and Henri Gilbert. A Known Plaintext Attack of FEAL-4 and FEAL-6. In Feigenbaum [Fei92], pages 172–181.
Available from: http://dx.doi.org/10.1007/3-540-46766-1_12.
One citation on page 89.

[Tez10a] Cihangir Tezcan. The Improbable Differential Attack: Cryptanalysis of Reduced Round CLEFIA. *IACR Cryptology ePrint Archive*, 2010:435, 2010.
Available from: http://eprint.iacr.org/2010/435.
2 citations on pages 85 and 203.

[Tez10b] Cihangir Tezcan. The Improbable Differential Attack: Cryptanalysis of Reduced Round CLEFIA. In Gong and Gupta [GG10], pages 197–209.
Available from: http://dx.doi.org/10.1007/978-3-642-17401-8_15,
DOI: doi:10.1007/978-3-642-17401-8.
2 citations on pages 85 and 203.

[THK99] Hidema Tanaka, Kazuyuki Hisamatsu, and Toshinobu Kaneko. Strenght of MISTY1 without FL Function for Higher Order Differential Attack. In Marc P. C. Fossorier, Hideki Imai, Shu Lin, and Alain Poli, editors, *AAECC*, volume 1719 of *Lecture Notes in Computer Science*, pages 221–230. Springer, 1999.
Available from: http://dx.doi.org/10.1007/3-540-46796-3_22.
2 citations on pages 81 and 175.

[Tie16] Tyge Tiessen. Polytopic Cryptanalysis. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 214–239. Springer, 2016.
Available from: https://doi.org/10.1007/978-3-662-49890-3_9,
DOI: doi:10.1007/978-3-662-49890-3_9.
One citation on page 186.

[Tok11] Natalia Tokareva. Generalizations of Bent Functions. A Survey. *Journal of Applied and Industrial Mathematics*, 5(1):110–129, January 2011.
Available from: http://rd.springer.com/article/10.1134/S1990478911010133.
One citation on page 60.

[Tos01] Toshiba Corporation. Specification on a Block Cipher : Hierocrypt–L1, September 2001.
Available from: http://www.cosic.esat.kuleuven.be/nessie/workshop/submissions/Hierocrypt-L1-revised-spec.pdf.
2 citations on pages 187 and 192.

[Tos02] Toshiba Corporation. Specification on a Block Cipher : Hierocrypt—3, May 2002.
Available from: http://www.cryptrec.go.jp/english/cryptrec_03_spec_cypherlist_files/PDF/08_02espec.pdf.
2 citations on pages 187 and 192.

[TRA90] TRADOC, The U.S. Army Training and Doctrine Command. Basic Cryptanalysis, Field Manual FM 34-40-2, September 1990.
Available from: http://www.umich.edu/~umich/fm-34-40-2/.
One citation on page 126.

[TTS+08] Yukiyasu Tsunoo, Etsuko Tsujihara, Maki Shigeri, Teruo Saito, Tomoyasu Suzaki, and Hiroyasu Kubo. Impossible Differential Cryptanalysis of CLEFIA. In Nyberg [Nyb08], pages 398–411.
Available from: http://dx.doi.org/10.1007/978-3-540-71039-4_25.
One citation on page 203.

[Vau94] Serge Vaudenay. On the Need for Multipermutations: Cryptanalysis of MD4 and SAFER. In Preneel [Pre95], pages 286–297.
Available from: http://dx.doi.org/10.1007/3-540-60590-8_22.
3 citations on pages 45, 148, and 152.

[Vau96] Serge Vaudenay. On the Weak Keys of Blowfish. In Gollmann [Gol96], pages 27–32.
Available from: http://dx.doi.org/10.1007/3-540-60865-6_39.
One citation on page 156.

[Vau98a] Serge Vaudenay, editor. *Fast Software Encryption, 5th International Workshop, FSE '98, Paris, France, March 23-25, 1998, Proceedings*, volume 1372 of *Lecture Notes in Computer Science*. Springer, 1998.
3 citations on pages 248, 276, and 293.

[Vau98b] Serge Vaudenay. Provable Security for Block Ciphers by Decorrelation. In Michel Morvan, Christoph Meinel, and Daniel Krob, editors, *STACS 98*, volume 1373 of *Lecture Notes in Computer Science*, pages 249–275. Springer, 1998.
Available from: http://dx.doi.org/10.1007/BFb0028566.
3 citations on pages 63, 64, and 65.

[Vau99a] Serge Vaudenay. Adaptive-Attack Norm for Decorrelation and Super-Pseudorandomness. In Howard M. Heys and Carlisle Michael Adams, editors, *Selected Areas in Cryptography*, volume 1758 of *Lecture Notes in Computer Science*, pages 49–61. Springer, 1999.
Available from: http://dx.doi.org/10.1007/3-540-46513-8_4.
One citation on page 64.

[Vau99b] Serge Vaudenay. On the Lai-Massey Scheme. In Kwok-Yan Lam, Eiji Okamoto, and Chaoping Xing, editors, *ASIACRYPT '99*, volume 1716 of *Lecture Notes in Computer Science*, pages 8–19. Springer, 1999.
Available from: http://dx.doi.org/10.1007/b72231.
One citation on page 37.

[Vau99c] Serge Vaudenay. Resistance Against General Iterated Attacks. In Stern [Ste99], pages 255–271.
Available from: http://dx.doi.org/10.1007/3-540-48910-X_18.
One citation on page 64.

[Vau00] Serge Vaudenay. Decorrelated Fast Cipher, September 2000.
Available from: http://lasec.epfl.ch/memo/dfc.shtml.
One citation on page 65.

[Vau02] Serge Vaudenay. Introduction to Decorrelation Theory – online manual, July 2002.
Available from: http://lasec.epfl.ch/memo/dec_manual.shtml.
One citation on page 65.

[Vau03] Serge Vaudenay. Decorrelation: A Theory for Block Cipher Security. *Journal of Cryptology*, 16(4):249–286, 2003.
Available from: http://dx.doi.org/10.1007/s00145-003-0220-6.
3 citations on pages 63, 64, and 65.

[Ver26] Gilbert Sandford Vernam. Cipher Printing Telegraph Systems for Secret Wire and Radio Telegraphic Communications. *Journal American Institute of Electrical Engineers*, XLV:109–115, 1926.
One citation on page 24.

[VHVM88] Ingrid Verbauwhede, F. Hoornaert, Joos Vandewalle, and H. De Man. Security and performance optimization of a new DES data encryption chip. *IEEE Journal of Solid-State Circuits*, 23(3):647–656, 1988.
One citation on page 235.

[Vie07] Michael Vielhaber. Breaking ONE.FIVIUM by AIDA an Algebraic IV Differential Attack. *IACR Cryptology ePrint Archive*, 2007:413, 2007.
Available from: http://eprint.iacr.org/2007/413.
One citation on page 82.

[vOW90] Paul C. van Oorschot and Michael J. Wiener. A Known Plaintext Attack on Two-Key Triple Encryption. In Damgård [Dam91], pages 318–325.
Available from: http://link.springer.de/link/service/series/0558/bibs/0473/04730318.htm.
One citation on page 101.

[vOW96] Paul C. van Oorschot and Michael J. Wiener. Improving Implementable Meet-in-the-Middle Attacks by Orders of Magnitude. In Koblitz [Kob96], pages 229–236.
Available from: http://dx.doi.org/10.1007/3-540-68697-5_18.
One citation on page 97.

[Wag99] David Wagner. The Boomerang Attack. In Knudsen [Knu99], pages 156–170.
Available from: http://dx.doi.org/10.1007/3-540-48519-8_12.
2 citations on pages 83 and 103.

[Wan08] Meiqin Wang. Differential Cryptanalysis of Reduced-Round PRESENT. In Serge Vaudenay, editor, *AFRICACRYPT 2008*, volume 5023 of *Lecture Notes in Computer Science*, pages 40–49. Springer, 2008.
Available from: http://dx.doi.org/10.1007/978-3-540-68164-9_4.
One citation on page 205.

[Web85] A.F. Webster. *Plaintext/Ciphertext Bit Dependencies in Cryptographic Algorithms*. PhD thesis, Queen's University at Kingston, May 1985.
One citation on page 43.

[WFS99] David Wagner, Niels Ferguson, and Bruce Schneier. Cryptanalysis of FROG. In *Proceedings of the Second AES Candidate Conference*, March 1999.
Available from: http://www.schneier.com/paper-frog.html.
One citation on page 178.

[WFY+04] Dai Watanabe, Soichi Furuya, Hirotaka Yoshida, Kazuo Takaragi, and Bart Preneel. A New Keystream Generator MUGI. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 87-A(1):37–45, 2004.
Available from: http://search.ieice.org/bin/summary.php?id=e87-a_1_37{&}category=D{&}year=2004{&}lang=E{&}abst=.
One citation on page 46.

[Wie99] Michael J. Wiener, editor. *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*. Springer, 1999.
One citation on page 276.

[Wil86] Hugh C. Williams, editor. *Advances in Cryptology - CRYPTO '85, Santa Barbara, California, USA, August 18-22, 1985, Proceedings*, volume 218 of *Lecture Notes in Computer Science*. Springer, 1986.
4 citations on pages 257, 259, 290, and 297.

[Win83] Robert S. Winternitz. Producing a One-Way Hash Function from DES. In David Chaum, editor, *CRYPTO '83*, pages 203–207. Plenum Press, New York, 1983.
One citation on page 67.

[Win84] Robert S. Winternitz. A Secure One-Way Hash Function Built from DES. In *IEEE Symposium on Security and Privacy*, pages 88–90. IEEE Computer Society, 1984.
One citation on page 67.

[WLFQ99] Wenling Wu, Bao Li, Dengguo Feng, and Sihan Qing. Cryptanalysis of some AES Candidate Algorithms. In Vijay Varadharajan and Yi Mu, editors, *ICICS '99*, volume 1726 of *Lecture Notes in Computer Science*, pages 13–21. Springer, 1999.
Available from: http://dx.doi.org/10.1007/978-3-540-47942-0_3.
One citation on page 179.

[WLFQ00] Wenling Wu, Bao Li, Dengguo Feng, and Sihan Qing. Linear cryptanalysis of LOKI97. *Journal of Software*, 11(2):202–206, February 2000.
One citation on page 179.

[WLH11] Yongzhuang Wei, Jiqiang Lu, and Yupu Hu. Meet-in-the-Middle Attack on 8 Rounds of the AES Block Cipher under 192 Key Bits. In Feng Bao and Jian Weng, editors, *Information Security Practice and Experience - 7th International Conference, ISPEC 2011, Guangzhou, China, May 30 - June 1, 2011. Proceedings*, volume 6672 of *Lecture Notes in Computer Science*, pages 222–232. Springer, 2011.
Available from: https://doi.org/10.1007/978-3-642-21031-0_17,
DOI: doi:10.1007/978-3-642-21031-0_17.
One citation on page 184.

[WN94] David J. Wheeler and Roger M. Needham. TEA, a Tiny Encryption Algorithm. In Preneel [Pre95], pages 363–366.
Available from: http://dx.doi.org/10.1007/3-540-60590-8_29.
One citation on page 159.

[WN97] David J. Wheeler and Roger M. Needham. TEA Extensions, October 1997.
Available from: http://www.ftp.cl.cam.ac.uk/ftp/users/djw3/xtea.ps.
One citation on page 160.

[WN98] David J. Wheeler and Roger M. Needham. Correction to XTEA. Technical report, Computer Laboratory, Cambridge University, England, October 1998.
Available from: http://www.movable-type.co.uk/scripts/xxtea.pdf.
One citation on page 162.

[WRG+11] Lei Wei, Christian Rechberger, Jian Guo, Hongjun Wu, Huaxiong Wang, and San Ling. Improved Meet-in-the-Middle Cryptanalysis of KTANTAN (Poster). In Parampalli and Hawkes [PH11], pages 433–438.
Available from: http://dx.doi.org/10.1007/978-3-642-22497-3_31.
2 citations on pages 95 and 101.

[WS12] Xiaoyun Wang and Kazue Sako, editors. *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-34961-4.
2 citations on pages 249 and 254.

[WSMP11] Meiqin Wang, Yue Sun, Nicky Mouha, and Bart Preneel. Algebraic Techniques in Differential Cryptanalysis Revisited. In Parampalli and Hawkes [PH11], pages 120–141.
Available from: http://dx.doi.org/10.1007/978-3-642-22497-3_9.
One citation on page 75.

[WT85] A.F. Webster and Stafford E. Tavares. On the Design of S-Boxes. In Williams [Wil86], pages 523–534.
Available from: http://dx.doi.org/10.1007/3-540-39799-X_41.
4 citations on pages 23, 43, 61, and 146.

[WWCH10] Meiqin Wang, Xiaoyun Wang, Kam Pui Chow, and Lucas Chi Kwong Hui. New Differential Cryptanalytic Results for Reduced-Round CAST-128. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E93.A(12):2744–2754, 2010.
One citation on page 147.

[WWW12] Shengbao Wu, Mingsheng Wang, and Wenling Wu. Recursive Diffusion Layers for (Lightweight) Block Ciphers and Hash Functions. In Knudsen and Wu [KW13], pages 355–371.
Available from: http://dx.doi.org/10.1007/978-3-642-35999-6_23.
2 citations on pages 50 and 51.

[WWY12] Yanfeng Wang, Wenling Wu, and Xiaoli Yu. Biclique Cryptanalysis of Reduced-Round Piccolo Block Cipher. In Mark Dermot Ryan, Ben Smyth, and Guilin Wang, editors, *ISPEC 2012*, volume 7232 of *Lecture Notes in Computer Science*, pages 337–352. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-29101-2_23.
One citation on page 227.

[WZ11] Wenling Wu and Lei Zhang. LBlock: A Lightweight Block Cipher. In Javier Lopez and Gene Tsudik, editors, *ACNS 2011*, volume 6715 of *Lecture Notes in Computer Science*, pages 327–344, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-21554-4_19.
One citation on page 215.

[WZF06] Wenling Wu, Wentao Zhang, and Dengguo Feng. Impossible Differential Cryptanalysis of ARIA and Camellia. *IACR Cryptology ePrint Archive*, 2006:350, 2006.
Available from: http://eprint.iacr.org/2006/350.
One citation on page 172.

[Yar10] Elias Yarrkov. Cryptanalysis of XXTEA. *IACR Cryptology ePrint Archive*, 2010:254, 2010.
Available from: http://eprint.iacr.org/2010/254.
One citation on page 162.

[Yca13] Bernard Ycart. A case of mathematical eponymy: the Vandermonde determinant. *Revue d'Histoire des Mathématiques*, 9(1):43–77, 2013.
Available from: http://arxiv.org/abs/1204.4716.
One citation on page 48.

[YG00] Amr M. Youssef and Guang Gong. On the Interpolation Attacks on Block Ciphers. In Schneier [Sch01], pages 109–120.
Available from: http://dx.doi.org/10.1007/3-540-44706-7_8.
One citation on page 120.

[YLL13] Zheng Yuan, Xian Li, and Haixia Liu. Impossible Differential-Linear Cryptanalysis of Reduced-Round CLEFIA-128. Cryptology ePrint Archive, Report 2013/301, 2013.
Available from: http://eprint.iacr.org/2013/301.
One citation on page 94.

[YMT97] Amr M. Youssef, Serge Mister, and Stafford Tavares. Akelarre: a new Block Cipher Algorithm. In Carlisle Adams and Mike Just, editors, *Selected Areas in Cryptography 1997*, page 14. Selected Areas in Cryptography Organizing Board, 1997.
Available from: http://www.sacconference.org/SAC-history.html.
One citation on page 47.

[YPK02] Yongjin Yeom, Sangwoo Park, and Iljun Kim. On the Security of CAMELLIA against the Square Attack. In Daemen and Rijmen [DR02c], pages 89–99.
Available from: http://dx.doi.org/10.1007/3-540-45661-9_7.
One citation on page 172.

[YTH96] Amr M. Youssef, Stafford Tavares, and Howard M. Heys. A New Class of Substitution Permutation Networks. In *Selected Areas in Cryptography 1996*, pages 132–147, 1996.
Available from: http://www.sacconference.org/SAC-history.html.
2 citations on pages 118 and 215.

[Yua10] Zheng Yuan. Cryptology ePrint Archive, Report 2010/093, 2010. Available at: http://eprint.iacr.org/2010/093.
2 citations on pages 184 and 185.

[YWLZ11] Xiaoli Yu, Wenling Wu, Yanjun Li, and Lei Zhang. Cryptanalysis of Reduced-Round KLEIN Block Cipher. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, *Inscrypt 2011*, volume 7537 of *Lecture Notes in Computer Science*, pages 237–250. Springer, 2011.
Available from: http://dx.doi.org/10.1007/978-3-642-34704-7_18.
One citation on page 226.

[Zhe97] Yuliang Zheng. The SPEED Cipher. In Rafael Hirschfeld, editor, *Financial Cryptography 1997*, volume 1318 of *Lecture Notes in Computer Science*, pages 71–90. Springer, 1997.
Available from: http://dx.doi.org/10.1007/3-540-63594-7_68.
One citation on page 32.

[Zhe02] Yuliang Zheng, editor. *Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, 2002, Proceedings*, volume 2501 of *Lecture Notes in Computer Science*. Springer, 2002.
Available from: https://doi.org/10.1007/3-540-36178-2,
DOI: doi:10.1007/3-540-36178-2.
2 citations on pages 250 and 259.

[ZMI89] Yuliang Zheng, Tsutomu Matsumoto, and Hideki Imai. On the Construction of Block Ciphers Provably Secure and Not Relying on Any Unproved Hypotheses. In Brassard [Bra90], pages 461–480.
Available from: http://dx.doi.org/10.1007/0-387-34805-0_42.
3 citations on pages 30, 32, and 167.

[ZRHD08] Muhammad Reza Z'aba, Håvard Raddum, Matthew Henricksen, and Ed Dawson. Bit-Pattern Based Integral Attack. In Nyberg [Nyb08], pages 363–381.
Available from: http://dx.doi.org/10.1007/978-3-540-71039-4_23.
2 citations on pages 192 and 205.

[ZWCZ12a] Jingyuan Zhao, Meiqin Wang, Jiazhe Chen, and Yuliang Zheng. New Impossible Differential Attack on SAFER+ and SAFER++. In Taekyoung Kwon, Mun-Kyu Lurl, and Daesung Kwon, editors, *ICISC 2012*, volume 7839 of *Lecture Notes in Computer Science*, pages 170–183. Springer, 2012.
Available from: http://dx.doi.org/10.1007/978-3-642-37682-5_13.
2 citations on pages 153 and 154.

[ZWCZ12b] Jingyuan Zhao, Meiqin Wang, Jiazhe Chen, and Yuliang Zheng. New Impossible Differential Attack on SAFER_{+} and SAFER_{++}. Cryptology ePrint Archive, Report 2012/715, 2012.
Available from: http://eprint.iacr.org/2012/715.
2 citations on pages 153 and 154.

[ZWY10] Shihui Zheng, Licheng Wang, and Yixian Yang. A new Impossible Differential Attack on SAFER Ciphers. *Computers & Electrical Engineering*, 36(1):180–189, 2010.
Available from: http://dx.doi.org/10.1016/j.compeleceng.2009.08.004.
One citation on page 154.