Analysis of Symmetric Cryptographic Algorithms

Kidmose, Andreas Brasen

Publication date:
2022

Document Version
Publisher's PDF, also known as Version of record

Analysis of Symmetric Cryptographic Algorithms

Andreas Brasen Kidmose

Ph.D. Thesis

April 2022

Document compiled on March 31, 2022.

Supervisor: Lars R. Knudsen
Supervisor: Andrey Bogdanov
Supervisor: Weizhi Meng
Co-supervisor: Birger Andersen
Co-supervisor: Tyge Tiessen

Technical University of Denmark
Department of Applied Mathematics and Computer Science

ISSN:
Serial no.:

Abstract

Nowadays computers have become an integral part of our lives. The Internet has changed the way we do everything, from banking to allowing us to work from home during a pandemic. More and more data is being transmitted through the Internet. The data might be sensitive and therefore we need to ensure the integrity and confidentiality to have a secure Internet.

Cryptography provides the tools necessary to secure communication on the Internet. This thesis deals with the analysis of block ciphers. These block ciphers are used for many applications due to their efficiency, from bulk encryption to IoT devices. How secure these block ciphers are is therefore fundamental for the security of the internet. The way we measure the security of a block cipher is by trying to break parts of it and seeing how far we can get. Thus the more security analysis we do on these block ciphers, the more confidence we have in their security. Understanding how secure they are is therefore critical to our communication systems.

This thesis comes in two parts. The part introduces block ciphers and various techniques for their analysis. The second part contains four publications written during the PhD studies. The first publication presents a tool based on the division property which allows for rapid evaluation of a cipher. We used this tool to find several new and improved distinguishers. The second publication presents a new quantum distinguisher for type-3 generalized Feistel networks based on Simon's and Grover's algorithms. The third publication is a formal analysis of the probabilities involved in boomerang attacks. In particular we prove that the common independence assumption is inherently flawed. The fourth publication analyzes the re-identification attack, which is a problem that can arise when using format preserving encryption to encrypt a database.

Resumé

I dag er computere blevet en integreret del af vores liv. Internettet har ændret den måde, vi gør alting på, lige fra bankoverførsler til at vi kan arbejde hjemmefra under en pandemi. Flere og flere data overføres via internettet. Dataene kan være følsomme, og derfor er vi nødt til at sikre integriteten og fortroligheden for at få et sikkert internet.

Kryptografi giver de nødvendige værktøjer til at sikre kommunikationen på internettet. Denne afhandling omhandler analysen af blokchiffer. Disse blokchiffrer anvendes til mange applikationer på grund af deres effektivitet, lige fra masse-kryptering til IoT-enheder. Hvor sikre disse blokchiffrer er, er derfor af afgørende betydning for internettets sikkerhed. Måden, hvorpå vi måler sikkerheden af en blokchiffer, er ved at forsøge at bryde dele af den og se, hvor langt vi kan komme. Jo flere sikkerhedsanalyser vi foretager af disse blokchiffrer, jo mere tillid har vi til deres sikkerhed. Det er derfor af afgørende betydning for vores kommunikationssystemer, at vi forstår, hvor sikre de er.

Denne afhandling består af to dele. I første del introduceres blokchiffer og forskellige teknikker til analyse af dem. Den anden del indeholder fire publikationer, der er skrevet under ph.d.-studierne. I den første publikation præsenteres et værktøj baseret på divisionsegenskaben, som gør det muligt at foretage en hurtig evaluering af et chiffer. Vi har brugt dette værktøj til at finde flere nye og forbedrede skelningsegenskaber. I den anden publikation præsenteres en ny kvantedifferentiator for generaliserede Feistel-netværk af type 3 baseret på Simons og Grovers algoritmer. Den tredje publikation er en formel analyse af de sandsynligheder, der er involveret i boomerang-angreb. Vi beviser navnlig, at den almindelige uafhængighedsantagelse er behæftet med fejl i sagens natur. I den fjerde publikation analyseres reidentifikationsangrebet, som er et problem, der kan opstå, når man anvender formatbevarende kryptering til at kryptere en database.

Acknowledgements

I had a bit of a tumultuous as a PhD student, as evidenced by the fact that five different people have at some point been part of my supervisor team. I would like to thank them all, Lars Ramkilde Knudsen and Andrey Bogdanov, who were my supervisors previously, and my current supervisors Weizhi Meng, Birger Andersen, and Tyge Tiessen. Especially Tyge Tiessen was instrumental in getting me across the finish line.

I would also like to thank Christian Jensen, who was the section head for most of my time as a PhD student, and our section secretary Ann-Cathrin Dunker. Your support was invaluable.

My co-authors Zahra Eskandari, Samir Hodžić, Lars Ramkilde Knudsen, Stefan Kölbl, and Tyge Tiessen, also deserved a massive thanks for the great collaboration we have had.

The Cyber Security section, now Cybersecurity Engineering, at DTU Compute has expanded and contracted quite a bit during my time. I would like to thank all my current and former co-workers, but listing each and everyone of them would take too long and be an error prone task. Thanks for all the beer, coffee, and puzzles.

I would also like to thank Christina Boura and Thomas Johansson for agreeing to be on the assessment committee, and Carsten Witt for chairing it.

Last but not least, I would like to thank my family and friends, who have been incredible supportive throughout this ordeal.

Contents

Abstract i

Resumé iii

Acknowledgements v

Contents vii

0 Introduction 1
0.1 Outline of the thesis 2

I Introduction to block ciphers and their security 5

1 Block Ciphers 7
1.1 The Block Cipher 7
1.1.1 The Adversary 8
1.2 Feistel Networks 9
1.3 Substitution Permutation Networks 10
1.4 Format Preserving Encryption 11
1.4.1 Methods 12
1.4.2 Challenges 12

2 Cryptanalysis 15
2.1 Elements of an Attack 15
2.2 Differential Cryptanalysis 17
2.3 Higher-Order Attacks 19
2.3.1 Higher-order Differentials 19
2.3.2 Polytopic Cryptanalysis 20
2.3.3 Boomerang Attacks 21
2.4 Integral Attacks and Division Property 22
2.4.1 Integral Attacks 22
2.4.2 Division Property 24
2.5 Quantum Cryptanalysis 25
2.5.1 Preliminaries 26
2.5.2 Grover's Algorithm 27
2.5.3 Simon's Algorithm 28

Bibliography **31**

II Publications 37

Finding Integral Distinguishers with Ease **39**
1 Introduction . . . 41
2 Division property and division trails . . . 44
2.1 Background . . . 44
2.2 Formalism of bit-based division properties . . . 45
2.3 Rules of choice vector propagation . . . 45
3 Solvatore - Automated Finding of Integral Properties . . . 47
3.1 Modeling division property propagation with SAT . . . 48
3.2 Finding integral distinguishers . . . 50
4 Distinguishers and Bounds . . . 51
4.1 Methodology . . . 52
4.2 SPN . . . 53
4.3 ARX . . . 53
4.4 Feistel . . . 55
4.5 Reflection . . . 55
4.6 Bit-sliced . . . 58
4.7 LFSR-based . . . 59
4.8 Overview . . . 59
5 Conclusion and Future Work . . . 59
A Implementation of Present . . . 63
B Overview of Distinguishers . . . 64
B.1 ChaCha . . . 65
B.2 Chaskey . . . 65
B.3 DES . . . 65
B.4 GIFT-64 . . . 65
B.5 LBlock . . . 65
B.6 Mantis . . . 65
B.7 Qarma . . . 65
B.8 RoadRunner . . . 66
B.9 Salsa . . . 66
B.10 SM4 . . . 66

On Quantum Distinguishers for Type-3 Generalized Feistel Network Based on Separability **67**
1 Introduction . . . 69
2 Preliminaries . . . 71
2.1 A brief overview of Simon's and Grover's algorithm . . . 72
3 Analysis of GFNs . . . 75
3.1 On quantum distinquishers for Type-1 and Type-2 GFN . . . 75

3.2 On separability of the Type-3 GFN . . . 78
4 Combining Simon's and Grover's algorithm . . . 82
5 Conclusions . . . 86
A Appendix . . . 90
A.1 Experimental results related to system (9) . . . 90

A Formal Analysis of Boomerang Probabilities **93**
1 Introduction . . . 95
2 Boomerang attacks . . . 99
2.1 Basic boomerangs . . . 99
2.2 Amplified boomerangs . . . 101
3 Independence assumptions in boomerang attacks . . . 102
3.1 Independence of rounds and trails . . . 102
3.2 An inherent problem . . . 102
4 Generalized differences and their transitions . . . 104
5 Rigorous statements for boomerang probabilties . . . 107
6 Comparison to the boomerang connectivity table (BCT) . . . 112
7 Boomerang attacks on Serpent . . . 112
7.1 Short overview on Serpent . . . 113
7.2 Amplified boomerang attack [15] . . . 113
7.3 Rectangle attack [3] . . . 113
8 Summary and conclusion . . . 115
A Description of Serpent . . . 119

Pitfalls in data-masking techniques: Re-identification attacks **123**
1 Introduction . . . 125
1.1 Motivation . . . 126
2 Constructing custom data sets . . . 128
2.1 Algorithms for merging attributes . . . 131
3 Statistical analysis of the customized data sets . . . 136
3.1 Customized data sets with 2 attributes . . . 136
3.2 Customized data sets with 3 attributes . . . 140
3.3 Customized data sets with 4 attributes . . . 141
4 Conclusions . . . 144

0 Introduction

Ever since humans started writing things down, there has been a need to keep the meaning of that writing secret. The history of cryptography is a centuries long arms race between the code makers and the code breakers. Many unbreakable codes have been proposed only be to broken by some clever cryptanalyst. Code making and breaking was mostly an art form, and the domain of linguists.

The transition from art to science started around World War II. The seminal paper by Shannon [34] is usually considered as the starting point of the theoretical study of cryptography. Shannon showed that there are in fact unbreakable cryptosystems, namely the one-time pad, and introduced the design criteria of *confusion* and *diffusion*, which are still in use today.

The advent of the computer revolutionized the field of cryptography, or one might even say that cryptography revolutionized the computer, as some of the first computers were build to break the Enigma code during World War II. The computing power of the adversary increased exponentially, and slowly computers were being connected so they could communicate over long distances. These changes led to a two remarkable developments in the 70's. First the standardization of the Data Encryption Standard (DES) [38] by the *US National Bureau of Standards* (now the *National Institute of Standards and Technology* or *NIST*), which became the focus of most cryptanalytical efforts during the following 30 years. The most common techniques used in cryptanalysis today were developed to attack DES specifically. Secondly the development of public-key cryptography, which solves the problem of communicating with someone without first sharing a secret key.

After almost 30 years the reign of DES came to a stop around the turn of the millennium. Computers had gotten much more powerful which meant that a new standard was needed. In 1997 NIST initiated a competition to find a replacement, which would eventually be known as the Advanced Encryption Standard (AES). Essentially a year long process, where cryptographers submitted their own cipher designs and then tried to break their opponents ciphers. This was done so all the information was out in the open to inspire trust in the process and therefore also the outcome. This has now become the standard way to run cryptographic standardization processes.

In 2000 NIST then announced that Rijndael was chosen to become AES [37] After 20 years there are no serious problems have been discovered in AES. It seems that with all the research going into breaking and making ciphers, the code makers finally have the upper hand.

Now another challenge is on the horizon, the quantum computer. For a long time it has been know that a quantum computer would be a problem for the public-key

systems we use today. We have therefore been working towards developing new standards to replace the old insecure ones. Recently evidence has emerged that a quantum computer might pose a bigger threat to symmetric cryptography than originally thought.

So that was a very brief history of cryptography, but what actually is cryptography? The short version is that cryptography is the study of communication in the presence of an adversary. We usually consider two parties that want to communicate, Alice and Bob. They want to communicate over an insecure channel, where Eve the eavesdropper can listen in. The problem cryptography tries to solve is how Alice and Bob can communicate, without Eve gaining any information on what the communication is about.

In general we have three goals we want to achieve with a cryptographic solution, however, depending on the application we might be not interested in all three simultaneously.

- Confidentiality: Alice and Bob should be confident that Eve cannot read their message.
- Integrity: Eve should not be able to alter the messages in a meaningful way.
- Authenticity: Eve should not be able to impersonate Alice or Bob.

Broadly speaking there are three types of cryptographic schemes based on how the key is used. In a *symmetric* or *secret key* cryptosystem both parties share the same key. In *public-key* or *asymmetric* system there are two different keys, a public and a private key, where the public key is used for encryption and the private key for decryption. Lastly the *keyless* primitives are things like hash functions which is usually lumped in with the symmetric schemes as the building blocks are often the same. This thesis is almost exclusively about symmetric cryptography.

0.1 Outline of the thesis

This thesis is a collection of paper and as such comes in two parts. Part I introduces the concepts needed to understand the publications in Part II. Chapter 1 deals with block ciphers. Chapter 2 gives a general introduction to cryptanalysis, followed by some specific attack vectors. Various types of differential attacks will be introduced including some higher order attacks like integrals, boomerangs, and polytopes. A gentle introduction to quantum computation will also be provided. Part II contains 4 publications written during the PhD. studies.

The first publication introduces a new tool for finding distinguishers using the division property. The tool formulates the division property transitions as a SAT problem, which can then be solved with a SAT solver. We then applied the tool to several ciphers and found some new and improved distinguishers for ChaCha, Chaskey, DES, GIFT, LBlock, Mantis, Qarma, RoadRunner, Salsa, and SM4.

The second publication presents a quantum distinguisher for type-3 generalized Feistel networks. We introduce the concept of separability and use that to find a distinguisher based on the combination of Simon's and Grover's algorithms. The distinguisher for a d-branch type-3 generalized Feistel network covers $d + 1$ rounds with a query complexity of $2^n \cdot \mathcal{O}(n)$ where n is the branch size.

The third publication studies boomerang attacks. First we prove that the usual independence assumption is inherently flawed, that is, if the differentials are independent, then there exists a normal differential that has a higher probability and thus would yield a better attack. We prove some statements about the probabilities of boomerangs, using the polytopic framework. Finally we re-analyze some previously published boomerang attacks in our framework and show one has probability 0, while another has about the same probability despite most trails having probability 0.

And finally the fourth publication takes a look at re-identification attacks. We use a real-world data set, given as the frequencies of the formats, to explore the range of databases that could have resulted in that distribution. What we show is that for the vast majority of user we can uniquely identify them in the databases we construct. This poses a serious problem for format-preserving encryption schemes and other masking schemes, where the format is preserved. Our work highlights that great care should be taken when using these schemes.

Part I

Introduction to block ciphers and their security

1 Block Ciphers

In this chapter we will introduce the main topic of this thesis, namely *block ciphers*. We will look at the two main ways to construct them, Feistel networks in Section 1.2, and substitution permutation networks in Section 1.3. And finally we will give a brief introduction to format preserving encryption Section 1.4

1.1 The Block Cipher

A block cipher is a function that encrypts a message with a key in such a way that only someone with the same key can decrypt it, i.e., recover the original message. The formal definition is given as follows.

Definition 1.1 (Block Cipher)**.** Let k and n be positive integers. A *block cipher* is an encryption function

$$\begin{aligned} \mathrm{Enc} : \mathbb{F}_2^k \times \mathbb{F}_2^k &\to \mathbb{F}_2^n \\ (K, M) &\mapsto C \end{aligned}$$

which, for a fixed $K \in \mathbb{F}_2^k$, is a permutation on $\mathbb{F}_2^n$. The inverse is the decryption function, denoted by Dec, such that $M = \mathrm{Dec}(K, \mathrm{Enc}(K, M))$. We call M, C, and K for the plaintext, ciphertext, and key respectively, and we use $\mathcal{M}$, $\mathcal{C}$, and $\mathcal{K}$ to denote the respective spaces. We call n and k for the block size and key size, respectively.

Often the key will be clear from the context and we will often leave it out, that is $\mathrm{Enc}(\cdot) = \mathrm{Enc}(K, \cdot)$ and $\mathrm{Dec}(\cdot) = \mathrm{Dec}(K, \cdot)$.

Not every function that fits this definition is useful; for instance it should be efficiently computable. We also want our block cipher to be secure. By secure we mean that the best attack on the block cipher should require the same effort as the generic attacks. The two main generic attacks are *exhaustive key search* and *dictionary attacks*. In Exhaustive key search, also called a *brute force* attack, we simply try all the possible keys, which means the computational effort is about 2^k. In a dictionary attack we ask for the encryption of all the possible plaintexts, and store the plaintext/ciphertext pairs. This gives us an equivalent function to the encryption function with that fixed key, at the expense of 2^n data pairs. These two attacks define the security level we expect from an n-bit block cipher with a k-bit key.

This leads us to the concept of an *ideal block cipher*, which is essentially how we would like our block cipher to behave.

Definition 1.2 (Ideal Block Cipher)**.** An *ideal block cipher* is a block cipher, that for each key picks a permutation on $\mathbb{F}_2^n$ uniformly at random.

Constructing an ideal cipher is infeasible, as specifying a random permutation requires us to write down the output for every input. What we do instead, is to build block ciphers from simpler functions, that is, our block cipher is a composition of some key-dependent round function.

$$\text{Enc} = f_r \circ f_{r-1} \circ \cdots \circ f_2 \circ f_1.$$

We call this construction a *product cipher*. The round functions are usually the same except for a round constant and the round key. We will look more at two specific constructions in Section 1.2 and Section 1.3.

1.1.1 The Adversary

It is now time to meet the adversary. First of all, we assume the attacker knows every detail about the system, except for the value of the secret key. This is know as *Kerchhoffs' assumption*.

The attacker can have different goals.

- Key recovery attack: The attacker tries to recover the secret key.
- Message recovery attack: Given a ciphertext the attacker has to find the corresponding plaintext. The attacker can make queries for other plaintext/ciphertext pairs.
- Distinguishing attack: Given black box access to an oracle that is either an ideal cipher or a concrete block cipher, the attacker has to determine which it is.

Obviously, if we can recover the key we can also recover the plaintext. Often a distinguishing attack can be turned into a key recovery attack, but more on that later. We classify the attacker's capabilities based on what type of queries they are allowed to make.

- Ciphertext only: Only ciphertexts are available to the attacker
- Known plaintext: The attacker gets plaintext/ciphertext pairs
- Chosen plaintext/ciphertext: The attacker can chose which plaintexts or ciphertexts to get the corresponding ciphertext or plaintext for
- Adaptively chosen plaintext/ciphertext: The attacker can chose plaintext/ciphertexts based on previous plaintexts or ciphertexts

These are the most common ones, but there are also other models. For example the related key model, where the attacker can get encryptions under two or more different keys with known differences.

For symmetric primitives we cannot reduce the security to a known, or assumed, hard problem like you would do for an asymmetric scheme, Instead we gain confidence in a design by showing that it resists all known, or at least the most common, attacks.

1.2 Feistel Networks

Feistel networks or Feistel ciphers are an important class of block ciphers with a prominent example being the DES [38]. Between its publication and the publication of the AES [37] it was the most used block cipher. Essentially a Feistel network can be used to make a pseudo random permutation from a pseudo random function, that is, it turns an n-bit random function into a $2n$-bit random permutation. They can be used for many things apart from as a block cipher. A well known application is OEAP [3] which can be used to introduce randomness to RSA encryption. It is also used a lot for Format Preserving Encryption, which we will talk more about in Section 1.4.

In the classical Feistel cipher the plaintext is split into two halves. In each round one half is passed trough the round function and then added to the other half, after which the halves are swapped (see Fig. 1.1),

$$(L_{t+1}, R_{t+1}) = (R_t, L_t \oplus f_t(R_t)).$$

In this section we will assume that the round function is an n-bit random function, which means that the Feistel cipher works on $2n$ bits. That is $f_t : \{0,1\}^n \to \{0,1\}^n$ and $\mathrm{Enc}_k : \{0,1\}^{2n} \to \{0,1\}^{2n}$ We have thus essentially reduced the problem of constructing a round function on $2n$ bits to constructing one one n bits.

It is interesting to note here that this transformation is its own inverse when f_t is kept fixed. Therefore to decrypt we just need to apply the round functions in reverse order. Since f_t is only ever applied in one direction there is also no requirement that it is a invertible. This makes Feistel cipher attractive for lightweight designs as the same circuit can be used for encryption and decryption.

Another property that makes Feistel ciphers attractive is the security proofs by Luby and Rackoff [29]. The paper shows that a 3-round Feistel scheme with (pseudo) random round function is secure against chosen plaintext attacks, and 4 rounds is secure against chosen plaintext and ciphertext attacks when the attacker can make at most $q \ll \sqrt{2^n}$ queries.

You might wonder why we don't see many 4-round Feistel ciphers then, which is because the results come with two caveats. First of all, the construction is proved secure when $q \ll \sqrt{2^n}$, and in a practical instantiation this bound might be broken by an attacker. Secondly most real world constructions don't use (pseudo) random functions but rather a keyed permutation. A keyed permutation is easier to implement, but the security proof also does not hold any longer. One classical example is the

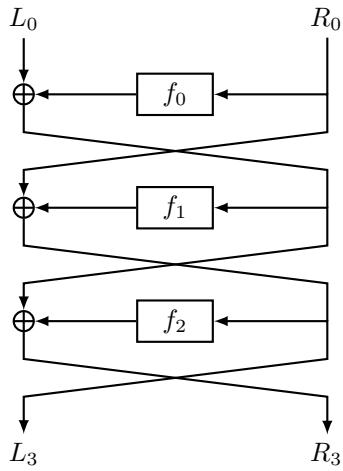


Figure 1.1: A standard 3-round Feistel Network.

5-round impossible differential attack by Knudsen [21]. For more details on Feistel network we refer to [32].

1.3 Substitution Permutation Networks

Substitution permutation networks, SPN for short, are the other major class of block ciphers. An SPN generally iterates a round function with 3 different operations several times. The operations are a non-linear substitution layer, a linear permutation layer, and finally a key addition (see Fig. 1.2). The substitution layer usually consists of applying a small non-linear permutation, called an S-box, in parallel. The permutation layer in its simplest form can be just a bit-permutation which is ideal in lightweight ciphers like PRESENT.

Since the competition to choose the cipher that would become the Advanced Encryption Standard(AES), they have gained widespread popularity. The winning cipher was an SPN called RIJNDAEL, which we will also just refer to as AES. The difference is that RIJNDAEL supports more options for block size and key length, whereas AES is limited to 128-bit block and a key of 128 192 or 256 bits. keys. The state of AES is represented as a 4×4 matrix of bytes. The round function of AES consists of four operations. The substitution layer is `SubBytes`. The permutation layer consist of two operations, `ShiftRows` and `MixColumn`, and finally the key addition called `AddKey`. The success of AES has now given rise to a whole class of AES-like ciphers. These ciphers use the same basic structure with a matrix representation of the state, but may change the the S-box or the `MixColumn` operation to get a more lightweight cipher. An example of this, is the MIDORI cipher by Banik et al. [1], which is designed for lightweight application with a focus on low energy consumption.

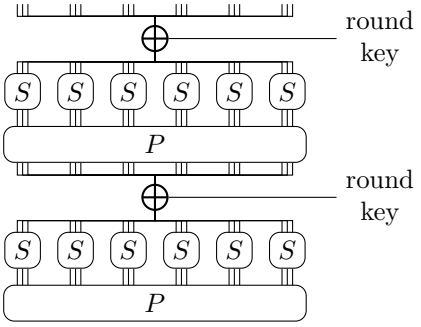


Figure 1.2: Three-round SPN.

Unlike Feistel networks, SPNs do not come with a security proof. A design strategy that gained popularity with the the introduction of AES is the *wide trail* strategy. The main idea is that by choosing the substitution and permutation layer carefully we can upper bound the probability for any differential trail by lower bounding the number of active S-boxes, i.e., S-boxes with a non-zero difference. For more on AES, the book by Daemen and Rijmen [12] is a good reference.

1.4 Format Preserving Encryption

Most of the time we can easily represent the data we want to encrypt as bit strings, which can then be split into blocks and encrypted using an appropriate mode. However, sometimes there are good reason why this might be inconvenient.

As an example, we might be encrypting the entries of a database individually, where the entries are numbers between zero and a billion (10^9). These numbers can be represented with just 30 bits, but if we use AES the ciphertext will be 128 bits long and therefore require 4 times as much storage.

Another common use case example is the American social security number or the Danish equivalent, the CPR number. Here the format is not just a fixed length string of digits, but there are certain restriction on which value some positions can take. For the CPR number the first 6 digits should be a valid date of birth. What we need to solve this is a cipher where $\mathcal{M} = \mathcal{C}$ can be an arbitrary finite domain.

We should note here that there are also other cases where FPE techniques can be used. One example is to generate test data. Since the format is preserved the encrypted data will look like the un-encrypted data, and since it is encrypted it should not leak any information. The encrypted data can therefore safely be used to test a database system, without exposing sensitive information to the developers. Synthetic data can solve the same problem, but it might not be representative.

1.4.1 Methods

One of the first rigorous treatments of this problem was given by Black and Rogaway [8], where they proposed several generic methods for using a standard block cipher to solve the problem. Here we will just explain one of them as an example, namely the *cycle walking cipher*.

The Cycle Walking Cipher

The simplest way to construct an FPE scheme from a block cipher is simply to repeatedly encrypt the message until it again fits the format. This is well-defined since a finite permutation has a unique cycle-decomposition, and we know at least one element of the cycle must adhere to the format. This works as as long as it is efficient to check for membership of the target domain, however, the number of encryption operations needed might make it infeasible. When using an n-bit block cipher, the expected number of block cipher calls is about $2^n/|\mathcal{M}|$, so if the domains have very different sizes this is a problem. One way to solve this would be to have a block cipher for any bit length, within reason, which is exactly what Lee et al. [28] proposed. In this case the block cipher domain is never more than twice as large as the target domain and the expected number of iterations is less than two.

NIST

The need for standardization in the area was recognized by NIST and therefore they published SP 800-38G [39]. It contains recommendation for two modes of operation specifically for FPE. The modes are called FF1 and FF3 [1] and both are based on Feistel networks with AES-128 as the building block of the round function.

1.4.2 Challenges

There are several challenges when designing a format preserving encryption scheme.

Small domains

A significant challenge for the security of FPE schemes is that the domain can be very small. This is not exclusive to FPE, e.g., the practical attack on 64-bit block cipher by Bhargavan and Leurent [4]. For FPE schemes this is particularly hard as the some of the proposed use cases have domains considerably smaller than that. This has been leveraged in several attacks, see e.g. [2], [15], [17]

Arbitrary domains

It is hard to produce a general purpose FPE scheme. In general efficient constructions have been limited to arbitrary lengths strings over a finite alphabet.

[1] FF2 was withdrawn due to safety concerns

Efficiency

A major concern for FPE is how make an efficient encryption scheme. The more general methods like cycle walking are impractical for a lot of cases. For example, using AES and cycle walking to encrypt a social security number would on average require 2^{98} AES evaluations. Using FF1 or FF3 still requires at least one AES call per round, so 10 or 8 total calls for 1 encryption.

Re-identification attacks

A real threat to format preserving encryption schemes, if they are not used carefully, is the re-identification attack. Imagine we have a database where each row corresponds to a person, and for each person we have several attributes which are encrypted individually. A format preserving encryption scheme has been used for the encryption, such that strings of characters are encrypted to strings of characters of the same length. To ensure that equal entries are not encrypted to the same ciphertext, a different tweak is used for each row. On the surface this seems like a sensible way to do things, however, there is a subtle problem.

We might know that a certain person is in the database and we want to figure out which row corresponds to this person. We know some information about this person that is also recorded in the database like name and city. We can now simple filter out all entries where the name and city do not have the correct lengths Unless it is a very common name or a very large database we can probably identify the individual. Once we have done that we might be able to guess data we didn't already know. An extreme example would be a yes/no question where the data is encoded as strings instead of 0/1, since then 3 letters means yes and 2 means no. The paper *"Pitfalls in data-masking techniques: Re-identification attacks"* deals with this problem in detail.

2 Cryptanalysis

In this chapter we will explore various techniques for cryptanalysis of block ciphers. First we will give a general introduction to cryptananlysis. In Section 2.2 we will take at look at the basics of differential cryptanalysis. Section 2.3 will explore higher-order attacks such as higher-order differentials and boomerang attacks. Section 2.4 will give an introduction to integral attacks and the division property. Finally Section 2.5 will look at how a quantum computer can be used for cryptanalysis of symmetric primitives. Let's get cracking!

2.1 Elements of an Attack

Most attacks are build from combining two basic building blocks, a *distinguisher* and *partial key guessing*.

Distinguishers

A *distinguisher* is any property that we can use to distinguish a cipher from a random permutation. They come in two flavors, deterministic and statistical.

A *deterministic distinguisher* is a distinguisher, which holds with probability one or zero regardless of the value of the key. For example an integral distinguisher is a deterministic distinguisher, where sum a set of texts and know the sum of some bits will be 0. For a random permutation the sum of each bit will be 0 with probability $1/2$ and thus ℓ bits will sum to 0 with probability $2^{-\ell}$

For a *statistical distinguisher* the property will have some probability that depends on the key. An example here would be a differential, that is, two texts whose encryption is more likely to have a specific difference. For a random permutation the difference of two outputs should be uniformly distributed and therefore have a probability of 2^{-n}.

Partial key guessing

Suppose that some part of the ciphertext does not depends on all bits of the key. If we then guess these key bits we can verify our guess by decrypting the corresponding part of the ciphertexts. This becomes especially useful when combined with a distinguisher.

Combining a distinguisher and partial key guessing

The most common type of attack consists of combining a distinguisher and partial key guessing. In an idealized model we have an r-round cipher with independent and uniformly random round keys. Now assume we have an $r-1$-round distingsuisher, which we can use for a *last-round* attack. We ask for the encryption of the set of texts that we need to detect the distinguisher. Then we guess the last round key, and decrypt the last round to see if we can detect the distinguisher. When we guess the right key the round is decrypted properly and we can detect the distinguisher, but decrypting with a wrong key will essentially correspond to encrypting another round. This is known as the *wrong key randomization hypothesis*:

Wrong key randomization hypothesis. *Decrypting one round with a wrong key is equivalent to encrypting another round, and thus the output is closer to that of a random permutation.*

When we have an attack we need to measure how good it is. We need to be able to tell if an attack is better than previously published results, which is not always a straight forward thing to do. The metrics we use to judge and compare the effectiveness attacks are:

- Time: A measure of the computational effort needed for the attack, usually measured in encryption operations
- Data: The number and type of plaintext/ciphertext pairs needed for the attack
- Memory: The amount of storage needed
- Probability: The probability of the attack succeeding

These days there are two main branches of cryptanalysis, which every cipher should be secure against, differential cryptanalysis and linear cryptanalysis. The coming sections will go into detail with differential cryptanalysis and the various related methods that have since been developed. As none of the publications in Part II deal with linear cryptanalysis we will only give a brief introduction.

Just as differential cryptanalysis, linear cryptanalysis was developed to attack DES. It was developed by by Matsui [30], and the basic idea is to approximate the non-linear part of the cipher. The attacker tries to find and input and output mask α, β such that $\alpha \cdot x = \beta \cdot f(x)$ with a probability different from $1/2$. For a random permutation, the probability will be $1/2$ and we can therefore use this as a distinguisher, but we can also directly recover one bit of information about the key. One advantage of linear cryptanalysis over differential cryptanalysis is that it only relies on known plaintexts, whereas differential cryptanalysis normally uses chosen plaintexts.

2.2 Differential Cryptanalysis

Differential cryptanalysis was one of the most important inventions in cryptanalysis. It was developed by Biham and Shamir [6] to attack DES. The main idea is to consider how differences propagate through the cipher. If we have two n-bit plaintexts, m_0, m_1, with a certain difference, $m_0 \oplus m_1 = \alpha$, then for a pseudo random permutation we would expect the ciphertext difference, $c_0 \oplus c_1 = \beta$, to be uniformly distributed on $\mathbb{F}_2^n$. If this is not the case we can use it as a distinguisher.

Now it is time to get a bit more formal. We are almost exclusively working in $\mathbb{F}_2^n$ for some $n \in \mathbb{N}$ with the addition operation being bitwise exclusive or (XOR). For most ciphers this is also the operations used to introduce the key material. In this case the natural difference operation is also XOR, as the difference is then invariant under key addition, however, it is straight forward to generalize to other cases. We will also use Δ to denote the a difference, so the plaintext difference would be Δm and the ciphertext difference Δc.

Definition 2.1 (Differential)**.** A differential is a pair of differences (α, β), called the input and output difference, respectively.

We use the notation $\alpha \xrightarrow{f} \beta$ to denote the differential over the function f. If we consider a particular pair of texts $(x, x \oplus \alpha)$ follows that differential, that is, $f(x) \oplus f(x \oplus \alpha) = \beta$, we write that as $\alpha \xrightarrow[x]{f} \beta$. A pair that follows the differential is called a *right pair*, and otherwise it is called a *wrong pair*. A differential has an associated probability, which is defined as follows:

Definition 2.2 (Differential probability)**.** Let $\mathbf{X}$ be a uniformly distributed random variable on $\mathbb{F}_2^n$. Let $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$. The probability of the differential $\alpha \xrightarrow{f} \beta$ is defined as

$$\Pr\left(\alpha \xrightarrow{f} \beta\right) \coloneqq \Pr_{\mathbf{X}}\left(\alpha \xrightarrow[\mathbf{X}]{f} \beta\right)$$

It is the conditional probability over the message space of getting the output difference given the input difference,

$$\Pr\left(\alpha \xrightarrow{f} \beta\right) = \Pr\left(\Delta c = \beta \mid \ \Delta m = \alpha\right).$$

The natural question now is, how do we calculate the probability of a differential distinguisher? The input space is too large to make it computationally feasible to compute the actual probability. To solve this we can take advantage of the fact that most ciphers are product ciphers. This means they consist of several repetitions of a cryptographically weak round function, that is,

$$f = f_r \circ f_{r-1} \circ \cdots \circ f_1.$$

It is therefore natural to consider the propagation of the differential for each round. This leads to the concept of a *differential trail*, which is also called a *differential characteristic*.

Definition 2.3 (Differential trail)**.** An r-round differential trail is an $(r+1)$-tuple of differences in $\mathbb{F}_2^n$.

That is, we fix the differences in the intermediate rounds $\alpha_0 \xrightarrow{f_1} \alpha_1 \xrightarrow{f_2} \dots \xrightarrow{f_r} \alpha_r$ The probability of a trail is then defined as

Definition 2.4 (Differential trail probability)**.** Let $\mathbf{X}$ be a uniformly distributed random variable on $\mathbb{F}_2^n$. Let $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $f = f_r \circ f_{r-1} \circ \dots \circ f_1$, and $(\alpha_0, \alpha_1, \dots, \alpha_r)$ be an r-round differential trail. The probability of the trail is

$$\Pr_{\mathbf{X}}\left(\alpha_0 \xrightarrow[\mathbf{X}]{f_1} \alpha_1 \text{ and } \alpha_1 \xrightarrow[f_1(\mathbf{X})]{f_2} \alpha_2 \dots \alpha_{r-1} \xrightarrow[f_{r-1}\circ\dots\circ f_1(\mathbf{X})]{f_r} \alpha_r\right)$$

Calculating the probability of a trail is much simpler than calculating the probability of a differential. A pair of texts can only follow one trail. We can therefore calculate the probability of the differential $\alpha \xrightarrow{f} \beta$ as the sum of the trails with input difference α and output difference β, that is,

$$\Pr\left(\alpha \xrightarrow{f} \beta\right) = \sum_{\alpha_1,\dots,\alpha_{r-1} \in \mathbb{F}_2^n} \Pr\left(\alpha \xrightarrow{f_1} \alpha_1 \xrightarrow{f_2} \alpha_2 \dots \alpha_{r-1} \xrightarrow{f_r} \beta\right)$$

There are too many trails to calculate the sum exactly, but by summing over a subset of the trails with the highest probability we can get a lower bound on the probability of the differential.

Now we just need to get the probability of the trails, which is still not an easy task. However, a simplifying assumption makes it tenable to calculate the probability. If we assume the rounds are independent, we can simply multiply the one round transition probabilities to get the trail probability. The assumption is justified by assuming the rounds keys are independent and uniformly random, and that the cipher is a *Markov cipher* [27]. In a Markov cipher the probability of the one round transition is independent of the actual value of the input. In practice these assumptions seem to work well enough to let us multiply the round transition probability to get the trail probability.

So far we have mostly ignored the effect of the key. In an actual differential attack the key will be a fixed value, but when calculating the probability of a differential, it is common to assume the round keys are uniform random values. This assumption is known as the *hypothesis of stochastic equivalence* due to Lai et al. [27], which is stated as follows

Hypothesis of stochastic equivalence. *The probability of the a differential over the message space is approximately equal to the probability over the message and key space for almost all values of the key.*

Luckily it seems to hold true for most good ciphers. This assumption allows us to assume that the distinguisher we have found actually works on a instantiation of a cipher.

There are several extension to the basic differential cryptanalysis. We will now briefly describe *truncated* and *impossible* differentials, but as these are not relevant to the publications in Part II the exposition will be kept short.

Truncated differentials, introduced by Knudsen [22], considers vector spaces of differences instead of fixed differences for the input and output. One advantage is that it allows the use of structures, i.e., imagine we have two differences, α_1 and α_2, by querying the four texts m, $m \oplus \alpha_1$, $m \oplus \alpha_2$, and $m \oplus \alpha_1 \oplus \alpha_2$ we get two pairs that have the first difference and two pairs that have the second difference.

Impossible differentials considers differentials with probability zero. The name seems to originate in the paper by Biham et al. [5] however the technique was used previously.

2.3 Higher-Order Attacks

In this section we will look at higher-order differentials and the related attacks of polytopic cryptanalysis and boomerang attacks.

2.3.1 Higher-order Differentials

Xuejia Lai [26] is usually credited as the first to suggest using higher-order differentials. By framing differential cryptanalysis in terms of derivatives, higher-order derivatives is an obvious extension. The first non-trivial example of higher-order differentials outperforming standard differentials is credited to Lars Knudsen [22].

We are again going to restrict ourselves to the case of $\mathbb{F}_2^n$ where the addition operation is XOR. The derivative of a function, $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$, at a point α, is then defined as

$$\Delta_\alpha f(x) = f(x) \oplus f(x \oplus \alpha)$$

The ith order-derivative at $(\alpha_1, \ldots, \alpha_i)$ is then

$$\Delta^i_{\alpha_1,\ldots,\alpha_i} f(x) = \Delta_{\alpha_i}(\Delta^{i-1}_{\alpha_1,\ldots,\alpha_{i-1}} f(x))$$

Example 2.5. Consider the second-order derivative of $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$ at (α_1, α_2).

$$\begin{aligned}\Delta^2_{\alpha_1,\alpha_2} f(x) &= \Delta_{\alpha_2}(\Delta_{\alpha_1} f(x)) \\ &= \Delta_{\alpha_2}(f(x) \oplus f(x \oplus \alpha_1)) \\ &= f(x) \oplus f(x \oplus \alpha_1) \oplus f(x \oplus \alpha_2) \oplus f(x \oplus \alpha_1 \oplus \alpha_2)\end{aligned}$$

One of the major problems of higher-order differentials is that there is no obvious way to iterate them, since there are multiple input differences but only one output difference. One easy way to use them is to consider the degree, as since it is a derivative, the degree drops by the order of the differential.

$$\deg(\Delta_\alpha(f(x)) \leq \deg(f(x)) - 1$$

A notable example of this approach is the KN-cipher [33], which was broken by Jakobsen and Knudsen [18].

Some other attacks can also be considered as clever ways of using higher-order differentials. Boomerang attacks, which we will meet in Section 2.3.3, can be seen as using second-order differentials to connect unrelated differentials for different parts of the cipher. Integral attacks use the structure of the cipher to try to find sets of input bits such that when they take on all combinations, some output bits will sum to 0. This is essentially a higher-order derivative but more on that in Section 2.4

2.3.2 Polytopic Cryptanalysis

An interesting approach to solve the problem of iterating higher-order differentials is polytopic cryptanalysis introduced by Tiessen [40]. The idea is to track the propagation of *s-polytopes* through the cipher. An *s-polytope* over $\mathbb{F}_2^n$ is an s-tuple of values from $\mathbb{F}_2^n$. As in normal differential we usually consider the relative positions of the texts which leads to the following definitions.

Definition 2.6 (d-difference)**.** A *d-difference* over $\mathbb{F}_2^n$ is a d-tuple of values from $\mathbb{F}_2^n$, corresponding to the position of a $(d+1)$-polytope relative to a single point.

Standard differentials thus corresponds to 1-differences, and higher-order differentials corresponds to truncated d-difference where the output d-difference sum to the output difference of the higher-order differential. The following example shows the relation between polytopes and differences.

Example 2.7. The 4-polytope (m_0, m_1, m_2, m_3), corresponds to the 3-difference $(m_0 \oplus m_1, m_0 \oplus m_2, m_0 \oplus m_3)$. The value m_0 here is called the *anchor*.

Most concepts from simple differential cryptanalysis generalize straight forwardly to d-differences.

Definition 2.8 (Polytopic transition with fixed anchor)**.** Let $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Let $\boldsymbol{\alpha}$ be a d-difference $(\alpha_1, \alpha_2, \ldots, \alpha_d) \in \mathbb{F}_2^{dn}$ and $\boldsymbol{\beta}$ be a d-difference $(\beta_1, \beta_2, \ldots, \beta_d) \in \mathbb{F}_2^{dn}$. The $(d+1)$-polytopic transition with anchor x, is denoted by $\boldsymbol{\alpha} \xrightarrow[x]{f} \boldsymbol{\beta}$, and we say that it holds of and only if

$$\begin{aligned} f(x) \oplus f(x \oplus \alpha_1) &= \beta_1 \\ \text{and } f(x) \oplus f(x \oplus \alpha_2) &= \beta_2 \\ &\cdots \\ \text{and } f(x) \oplus f(x \oplus \alpha_d) &= \beta_d \end{aligned}$$

The probability of a polytopic transition is then defined exactly like for normal differentials by letting the anchor be a random variable.

Definition 2.9 (Polytopic transition)**.** Let f, $\boldsymbol{\alpha}$, and $\boldsymbol{\beta}$ be defined as in Definition 2.8 Let $\mathbf{X}$ be a uniformly distributed random variable on $\mathbb{F}_2^n$. The probability of the polytopic transition $\boldsymbol{\alpha} \xrightarrow{f} \boldsymbol{\beta}$ is defined as

$$\Pr\left(\boldsymbol{\alpha} \xrightarrow{f} \boldsymbol{\beta}\right) \coloneqq \Pr_{\mathbf{X}}\left(\boldsymbol{\alpha} \xrightarrow[\mathbf{X}]{f} \boldsymbol{\beta}\right)$$

An important consequence of this definition is that an input d-difference $\boldsymbol{\alpha}$ can only be mapped to at most 2^n different output difference with non-zero probability, which is given in [40, Lemma 1]. This is due to the fact that there are only 2^n different anchors. After r rounds one d-difference can therefore be mapped to at most 2^{rn} different d-differences. The total number of d-differences is 2^{dn}, so by choosing a suitably large d the ratio of possible to total d-differences can be kept small. This simple fact is what makes impossible polytopes competitive.

The framework can also be used to analyze other attacks, for example boomerangs can be seen as 3-differences. This is the framework we use in the publication "A Formal Analysis of Boomerang Probabilities," which appears as publication three in the second part of this thesis.

2.3.3 Boomerang Attacks

The boomerang attack was introduced by Wagner [45] to attack the COCONUT98 cipher [44], and is shown in Fig. 2.2a. The idea is to split the cipher into two parts $\mathrm{Enc} = E_1 \circ E_0$. We assume there is differential over E_0, $\alpha \xrightarrow{E_0} \beta$, with probability p and a differential over E_1, $\gamma \xrightarrow{E_1} \delta$, with probability q. A boomerang distinguisher is then constructed by first asking for the encryption p_0 and p_1, where $p_0 \oplus p_1 = \alpha$. The two ciphertexts $z_0 = \mathrm{Enc}(p_0)$ and $z_1 = \mathrm{Enc}(p_1)$, are then transformed into two new ciphertexts by adding the difference δ, that is, $z_2 = z_0 \oplus \delta$ and $z_2 = z_0 \oplus \delta$, for which we then ask for the decryption. We then say that the boomerang returns if $p_2 \oplus p_3 = \alpha$, where $p_2 = \mathrm{Enc}^{-1}(z_2)$ and $p_3 = \mathrm{Enc}^{-1}(z_3)$.

If we look at the middle, we see that $y_0 \oplus y_2 = y_1 \oplus y_3 = \gamma$, and therefore $y_2 \oplus y_3 = y_0 \oplus y_1 = \beta$. We can also look at it from the 3-difference perspective, the differences are $(\beta, \gamma, \beta \oplus \gamma)$. This corresponds to a second-order derivative, and this derivative is what forces the boomerang to return.

The classical estimate of the probability for the boomerang to return is based on assuming the differentials are independent. The probability that $y_0 \oplus y_1 = \beta$, $y_0 \oplus y_2 = \gamma$, and $y_1 \oplus y_3 = \gamma$, is pq^2 when we assume independence, and in this case we also have $y_2 \oplus y_3 = \beta$. Therefore the estimated probability of the boomerang returning is p^2q^2.

The intermediate differences are not actually important, and therefore the probability we are really interested in is

$$\Pr\left(\text{Boomerang returns}\right) = \Pr_x\left(\mathrm{Enc}^{-1}(\mathrm{Enc}(x) \oplus \delta) \oplus \mathrm{Enc}^{-1}(\mathrm{Enc}(x \oplus \alpha) \oplus \delta) = \alpha\right).$$

This can be calculated as

$$\sum_{\beta_0 \oplus \beta_1 \oplus \gamma_0 \oplus \gamma_1 = 0} \Pr\left(\alpha \xrightarrow{E_0} \beta_0\right) \cdot \Pr\left(\alpha \xrightarrow{E_0} \beta_1\right) \cdot \Pr\left(\gamma_0 \xrightarrow{E_1} \delta\right) \cdot \Pr\left(\gamma_1 \xrightarrow{E_1} \delta\right),$$

where we again assume that the differentials are independent.

The independence assumption, however, is a rather strong one. Wagner noted, in the original paper, that a careful choice of differences could be used to boost the probability in a Feistel cipher, what would later be called the *Feistel switch*. Several other switches were proposed by Biryukov and Khovratovich [7], which essentially exploit the dependence of the differentials to boost the probability of the boomerang returning. Murphy [31] then showed that the dependency could also go the other way, such that a boomerang never returns.

The sandwich attack was proposed Dunkelman et al. [13], [14], and it is depicted in Fig. 2.2b. The idea is to separate the two sub-cipher by a thin middle part, $\text{Enc} = E_1 \circ E_m \circ E_0$, like two slices of bread with a thin piece of meat in the middle, which is where the name comes from. The probability through the middle part is then

$$r = \Pr\left(x_2 \oplus x_3 = \beta \mid \ x_0 \oplus x_1 = \beta \wedge y_0 \oplus y_2 = \gamma \wedge y_1 \oplus y_3 = \gamma\right),$$

and the total probability is $p^2 q^2 r$. The idea is to then treat the dependency between the differentials in the middle part.

Cid et al. [10] proposed the boomerang connectivity table(BCT) as a tool for calculating the transition probability through E_m, when it consists of one S-box layer.

For an invertible function $f : \{0,1\}^n \to \{0,1\}^n$ the BCT is defined as

$$\text{BCT}(\beta, \gamma) = \left|\left\{x \in \{0,1\}^n \mid \ f^{-1}(f(x) \oplus \gamma) \oplus f^{-1}(f(x \oplus \beta) \oplus \gamma) = \beta\right\}\right|.$$

Notice that this is essentially how we defined the return of the boomerang.

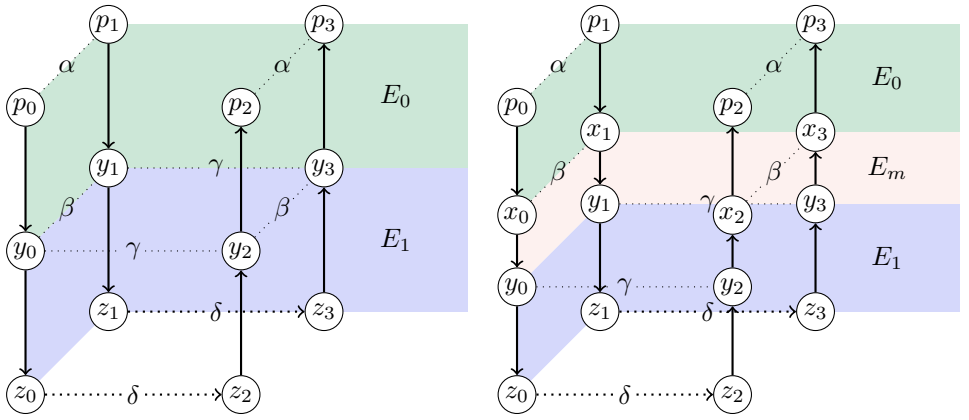
2.4 Integral Attacks and Division Property

In this section we will explore integral attacks and the division property. Integral attacks were introduced as a dedicated structural attack on the block cipher Square [11], and it was later formalized by Knudsen and Wagner [23]. The division property was introduced by Todo [42] as a generalization of the integral attack. It quickly showed its usefulness when Todo [41] used it to present the first attack on the full Misty1. First in Section 2.4.1 we will look at the classical intregral attack and then we will continue with the division property in Section 2.4.2.

2.4.1 Integral Attacks

The attack consists of tracking 4 properties for a set of texts, where the properties usually apply to part of the state. For example for AES it makes sense consider the properties at the byte level. The properties are:

Figure 2.1: The boomerang attack and the sandwich attack



(a) The standard boomerang attack.

(b) The sandwich attack.

- CONSTANT($\mathcal{C}$) where all the texts have the same value
- ALL($\mathcal{A}$) where the texts take all possible values
- BALANCED($\mathcal{B}$) where the texts sum to 0
- UNKNOWN($\mathcal{U}$) where we don't know anything about the texts

These properties form a hierarchy from most information to no information in the order they were presented here.

We start of the attack with at least 1 bit constant and the rest taking all possible values. We call the constant bits inactive and the other bits for active. If after some rounds a part of the state is still balanced we can use that as a distinguisher.

The transition rules are fairly straight forward for SPN-ciphers and especially for AES-like ciphers. All the properties are invariant under key and constant addition. S-boxes preserve CONSTANT and ALL, but BALANCED becomes UNKNOWN. This follows the fact that it is a bijection. For linear layers the picture is a bit more complicated as there is a wide variety with different characteristics. If we consider AES-like ciphers, then the linear layer consists of two operations, `ShiftRows` and `MixColumns`. `ShiftRows` just moves the bytes around and therefore the properties just move with them. For `MixColumns` it is more complicated as it acts on several bytes which could have different properties. If all the bytes are $\mathcal{C}$ then they will also be $\mathcal{C}$ after `MixColumns`, but if just one is $\mathcal{A}$ then they will all be $\mathcal{A}$ afterwards, however, if more than one is $\mathcal{A}$ we can only say that they will be BALANCED. If there is one $\mathcal{B}$ they will all be $\mathcal{B}$ and if there is one $\mathcal{U}$ they will all be $\mathcal{U}$.

To illustrate this Fig. 2.3 shows the classical 3-round attack, which works equally well for Square and AES. In the first round the S-box layer and `ShiftRows` does

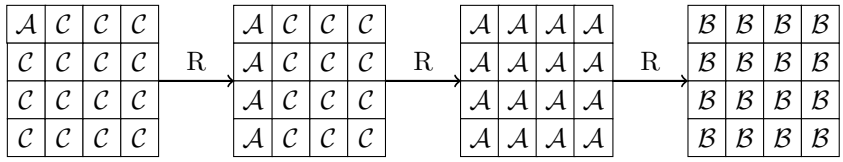


Figure 2.3: The classical 3-round integral attack on SQUARE or AES

nothing, but `MixColumn` spreads out the ALL property to the entire first column. In the second round the S-box layer does nothing, but this time `ShiftRows` spread the $\mathcal{A}$'s of the first column out so there is one in each column, and then `MixColumn` spreads them to the entire column. In the last round we also only need to consider `MixColumn`, where due to the linearity they will all be BALANCED. If we try to continue this propagation the S-boxes in the new round will the parity of all cells UNKNOWN.

We we are essentially doing here is to approximate the algebraic normal form of the output bits. When we take the sum over all the texts we are taking a derivative in the direction of the active bits. If the ANF of a bits does not have the monomial with all the active bits then we know the derivative is 0.

This is a structural attack and as such we do not really exploit the finer details of the cipher. For the S-box we only use the bijectivity and not the degree, and for the linear layer we often just use the fact that it is linear. This is what lead to the development of the division property.

2.4.2 Division Property

In essence the division property is a way to keep track of which bits contain the monomial with all the active bits in their ANF. The way this is done is by keeping track of a number of intermediate properties between $\mathcal{A}$ and $\mathcal{B}$. First we need to introduce a bit of notation which will be a bit of a mix between [42] and [43]. We will keep the discussion to $\mathbb{F}_2^n$ but note that it is trivial to extend to vectors.

We use $\mathbf{hw}(x)$ to denote the hamming weight of x. We also need the bit product function $\pi_u : \mathbb{F}_2^n \to \mathbb{F}_2$.

$$\pi_u(x) = \prod_{i=1}^{n} x[i]^{u[i]}$$

Basically u chooses which bits of x we are going to multiply together. We can now introduce the division property as

Definition 2.10 (Division Property [42])**.** Let $\mathbb{X}$ be a multiset with elements from $\mathbb{F}_2^n$. The multiset X has the division property $\mathcal{D}_k^n$ when the following condition is

fulfilled:

$$\bigoplus_{x \in \mathbb{X}} \pi_u(x) = \begin{cases} 0 & \forall u : \mathbf{hw}(u) < k \\ unknown & otherwise \end{cases}$$

We can now redefine the classical integral properties in terms of the division property. UNKNOWN corresponds to $\mathcal{D}_1^n$ as we don't know the parity for any bits. For BALANCED we know that if we take one bit the parity will be 0, and therefore it has the division property $\mathcal{D}_2^n$. If a multiset has the ALL property, then it has the division property $\mathcal{D}_n^n$. If we take all n bits only the all 1 word will have $\pi_u(x) = 1$, and therefore the parity depends on whether all the elements appear an even or odd number of times. For $n-1$ bits we will have two versions of each pattern, one where the left-out bits is 0 and one where it is 1, the parity will therefore be even. Note that a multiset can have division property $\mathcal{D}_n^n$ without having the ALL property. For example, if all values occurs an even number of times then the multiset will also have division property $\mathcal{D}_n^n$. The power of the division property, as previously mentioned, comes from keeping track of the intermediate properties between $\mathcal{D}_2^n$ and $\mathcal{D}_n^n$. To keep this exposition simple we will skip a rigorous treatment of the propagation rules but refer to [42] and [43].

One of the main draws of the division property is the ease with which it allows translating the problem of finding distinguishers into an MILP or SAT problem. Of course the MILP and SAT problems are NP-complete so it might seem like we have not gained much from the translation. However, there are solvers that work quite well on practical instances of these problems using various heuristics. This translation allows us to build a tool that translates a high-level description into transition rules as a SAT/MILP problem and then have a black-box SAT/MILP solver spit out the answers. This is exactly what the publication "Finding Integral Distinguishers with Ease" in Part II is about.

2.5 Quantum Cryptanalysis

In this section we will look how quantum computers can be used to break classical cryptosystems. Ever since the seminal paper by Shor [35], which shows a polynomial time algorithm that would break RSA, it has been know that quantum computers pose a serious threat to the security of public-key cryptography. It was assumed that the algorithm by Grover [16] was the only threat to symmetric cryptosystems. Grover's algorithm can be used to reduce the time complexity for exhaustive search to the square root of the key size and thus doubling the key would get the original security back. In 2010, however, Kuwakado and Morii [24] showed how to use Simon's algorithm [36] to perform a distinguishing attack on 3-round Feistel cipher in polynomial time. The last decade has seen a lot of research in quantum cryptanalysis of symmetric primitives, from attacking modes [19] to quantizing classical cryptanalysis techniques [20], and analyzing the securit of AES againt quantum attacks [9].

2.5.1 Preliminaries

When we do classical computing, our information unit is the bit, that is $b \in \{0, 1\}$. The equivalent for quantum computing is the qubit, which is a two dimensional complex vector, $|\psi\rangle \in \mathbb{C}^2$. The computational basis is $|0\rangle = \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$ and $|1\rangle = \left[\begin{smallmatrix}0\\1\end{smallmatrix}\right]$. A single qubit can therefore be written as $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ where α and β are called the amplitudes of $|0\rangle$ and $|1\rangle$, respectively. We usually assume that they are normalized such that $|\alpha|^2 + |\beta|^2 = 1$. We say that the qubit is in a *superposition*, meaning that the qubit is not 0 or 1 as a classical bit would be, but some combination of $|0\rangle$ and $|1\rangle$. The way we read out the data in a qubit is by doing a *measurement*, which turns our qubit into a classical bit. If we measure $|\psi\rangle$ we get a 0 with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$. For an n-qubit state use the tensor product, that is $|\psi\rangle \in \left(\mathbb{C}^2\right)^{\otimes n}$

Using $|\cdot\rangle$ for vectors is called the Dirac notation or bra-ket notation and is the usual notation for quantum mechanics. A column vector is a ket with the symbol $|\cdot\rangle$ and a bra is then the transposed complex conjugate denoted by $\langle\cdot|$ then the inner product is $\langle\cdot \mid \cdot\rangle$.

The next step is to consider what kinds of operations we can do to our qubits. We can use all unitary operators[1] $U : \left(\mathbb{C}^2\right)^{\otimes n} \to \left(\mathbb{C}^2\right)^{\otimes n}$. Instead of considering all unitary operators we just concern ourselves with a finite set of gates. Luckily we know universal gate sets exist, that is, gate sets which can approximate any unitary operator to arbitrary precision. The *Solovay-Kitaev theorem* ensures that this approximation is efficient.

An example if such a universal gate set is the Clifford + T gate set, which consists of 4 gates. The three single cubit gates

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \qquad S = \begin{bmatrix}1 & 0\\ 0 & e^{i\frac{\pi}{2}}\end{bmatrix} \qquad T = \begin{bmatrix}1 & 0\\ 0 & e^{i\frac{\pi}{4}}\end{bmatrix}$$

and the two cubit controlled-NOT or CNOT gate

$$CNOT = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}$$

which implements the operator $U : |x\rangle\,|y\rangle \mapsto |x\rangle\,|y \oplus x\rangle$. Especially the Hadamard gate, H, is important. Applied to the all-zero state, $|0\rangle$, we get a superposition were all the basis state have equal amplitudes, here illustrated on a single qubit

$$H\,|0\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\1\end{bmatrix} = \frac{1}{\sqrt{2}}\left(\begin{bmatrix}1\\0\end{bmatrix} + \begin{bmatrix}0\\1\end{bmatrix}\right) = \frac{1}{\sqrt{2}}\,|0\rangle + \frac{1}{\sqrt{2}}\,|1\rangle\,.$$

[1] An operator is unitary if $U^*U = UU^* = I$, where I is the identity and U^* is the adjoint of U

If we instead apply the Hadamard gate to $|1\rangle$ we get $H|1\rangle = 2^{-1/2}(|0\rangle - |1\rangle)$, hence if we use it on n qubits $|x\rangle$ we get

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}}\sum_y (-1)^{x\cdot y}|y\rangle$$

A last thing to notice is that the computations we do need to be reversible. This can be achieved using auxiliary qubits. Say we want to implement a function $f : \{0,1\}^n \to \{0,1\}^n$. We prepare $2n$ qubits $|x\rangle|0\rangle$ and then apply the transformation $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$.

Quantum Adversaries

When we do classical cryptanalysis we classify attackers based on what type of data they can get, e.g., known plaintext or chosen plaintext. We also classify quantum adversaries based on what type of oracle access they have. In the **Q1** model the adversary has access to a quantum computer but can only make classical queries. In the **Q2** model the adversary can make superposition queries, that is, the adversary can prepare a superposition of classical inputs and get the corresponding superposition of the outputs. The **Q2** is obviously a very strong model, but security in the **Q2** model also ensures security in weaker models. Using Grover's algorithm for exhaustive key-search would fall into the **Q1** model as we just need to make a few classical queries to construct the oracle. Simon's algorithm, on the other hand, usually requires superposition queries and thus fall into the **Q2** model.

2.5.2 Grover's Algorithm

Grover's algorithm solves the following search problem:

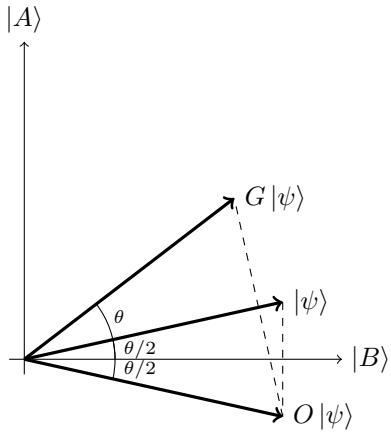
Problem 2.11 (Grover's Problem)**.** Given a set X of size N and a Boolean function $f : X \to \{0,1\}$, find an $x \in X$ for which $f(x) = 1$.

In the following we will assume that $M = |\{x \in X \mid f(x) = 1\}|$, that is, the number of elements for which $f(x) = 1$, is known.

Algorithm 1 Grover's Algorithm

1: $|\psi\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N}|x\rangle$
2: **loop**
3: Apply oracle $O|x\rangle \to (-1)^{f(x)}|x\rangle$
4: Apply Hadamard $H^{\otimes n}$
5: Conditional phase shift $|0\rangle \to |0\rangle$ and $|x\rangle \to -|x\rangle$
6: Apply Hadamard $H^{\otimes n}$
7: **end loop**
8: Measurement

Figure 2.4: Visualization of Grover's algorithm.



First we consider the two states $|A\rangle$, which is a superposition of x for which $f(x) = 1$, and $|B\rangle$, which is the orthogonal state. We call $|A\rangle$ for the good states and $|B\rangle$ for the bad states. The main part of the algorithm is the loop which us called a Grover iteration, and denote by $G = (2|\psi\rangle\langle\psi| - I)O$. Here the first part, $(2|\psi\rangle\langle\psi| - I)$, is also called the *inversion about the average*. We start from an equal superposition, so the oracle will essentially reflect the good states on the axis defined by the bad states. The inversion is then a reflection around the axis of the equal superposition. This is illustrated in Fig. 2.4.

We can also use this representation to estimate the number of iterations needed. We write the initial state $|\psi\rangle$ in terms of $|A\rangle$ and $|B\rangle$

$$|\psi\rangle = \sqrt{\frac{M}{N}}\,|A\rangle + \sqrt{\frac{N-M}{N}}\,|B\rangle\,.$$

From this it is clear that $\sin(\theta/2) = \sqrt{M/N}$, and that the rotation angle is $\theta \approx 2\sqrt{M/N}$, when we assume that $M \ll N$. After r rotations, the probability of measuring a good state is therefore $\sin((r + 1/2)\theta)^2$. If $(r + 1/2)\theta = \pi/2$ the probability is 1, so we need $r \approx \pi/4\sqrt{N/M}$

In cryptanalysis we can use this algorithm to find the secret key. We simply ask for a few known plaintext/ciphertext pairs (m_i, c_i) and then construct f such that on input of a key it returns 1 if and only if the key encrypts m_i to c_i for all i.

2.5.3 Simon's Algorithm

Simon's algorithm solves the following problem:

Problem 2.12 (Simon's Problem)**.** Given a Boolean function $f : \{0,1\}^n \to \{0,1\}^n$ and the promise that there exist an $s \in \{0,1\}^n$ s. t. $f(x) = f(y) \Leftrightarrow x \oplus y \in \{0, s\}$, find s.

This is an instance of the *hidden subgroup problem*, where the group is $(\{0,1\}^n, \oplus)$. Since $f(x) = f(x \oplus s)$ means that f is periodic, the problem is also known as the *hidden period* or *hidden shift* problem and s can also be referred to as the period.

Algorithm 2 Simon's Algorithm

1: **repeat**
2: Start with 2 n-qubit registers $|0\rangle |0\rangle$
3: Apply Hadamard transform to first register $H |0\rangle |0\rangle = \frac{1}{\sqrt{2^n}} \sum_x |x\rangle |0\rangle$
4: Apply oracle to get $\frac{1}{\sqrt{2^n}} \sum_x |x\rangle |f(x)\rangle$
5: Apply Hadamard transform to first register $\frac{1}{2^n} \sum_y \sum_x (-1)^{x \cdot y} |y\rangle |f(x)\rangle$
6: Do a measurement and collect vector y_i
7: **until** n linearly independent vectors have been collected
8: Solve the system of linear equations to find s

We will now explain how Simon's algorithm, Algorithm 2, works. The key to understanding the algorithm is the measurement in step 6.

Firstly, if f is one-to-one, i.e., $s = 0$, all the values in $|f(x)\rangle$ are distinct and we get randomly selected values for y_i. When we try to solve the system we will get a random values s, but we can easily check if $f(0) = f(s)$, and if it is not the case we know the true s is equal to 0.

If $s \neq 0$, we know that the vectors we collect are orthogonal to the shift, that is, $s \cdot y_i = 0$. If $s \cdot y = 0$, then $x \cdot y = (x \oplus s) \cdot y$, and therefore the register for y will have an amplitude of 2^{1-n}, but if $s \cdot y = 1$, then $x \cdot y = (x \oplus s) \cdot y \oplus 1$, and the amplitudes will cancel out. Therefore we do in fact get the correct value for s when we solve the system.

It may not be completely obvious that Simon's algorithm can be used for cryptanalysis. An illustrative example is the attack on the Even-Mansour cipher by Kuwakado and Morii [25]. Encryption with the Even-Monsour cipher is defined as $\text{Enc}(m) = P(m \oplus k_1) \oplus k_2$, where P is a public permutation and k_1, k_2 are secret keys. If we consider the function $f(x) = \text{Enc}(x) \oplus P(x) = P(m \oplus k_1) \oplus k_2 \oplus P(x)$, it is easy to see that $f(x) = f(x \oplus k_1)$, and therefore k_1 can be found with Simon's algorithm. It is then easy to find k_2, as $k_2 = c \oplus P(m \oplus k_1)$.

Bibliography

[1] Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, Kyoji Shibutani, Harunaga Hiwatari, Toru Akishita, and Francesco Regazzoni. "Midori: A Block Cipher for Low Energy". In: *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*. Ed. by Tetsu Iwata and Jung Hee Cheon. Vol. 9453. Lecture Notes in Computer Science. Springer, 2015, pp. 411–436. DOI: 10.1007/978-3-662-48800-3_17. URL: https://doi.org/10.1007/978-3-662-48800-3_17.

[2] Mihir Bellare, Viet Tung Hoang, and Stefano Tessaro. "Message-Recovery Attacks on Feistel-Based Format Preserving Encryption". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*. Ed. by Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi. ACM, 2016, pp. 444–455. DOI: 10.1145/2976749.2978390. URL: https://doi.org/10.1145/2976749.2978390.

[3] Mihir Bellare and Phillip Rogaway. "Optimal Asymmetric Encryption". In: *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings*. Ed. by Alfredo De Santis. Vol. 950. Lecture Notes in Computer Science. Springer, 1994, pp. 92–111. DOI: 10.1007/BFb0053428. URL: https://doi.org/10.1007/BFb0053428.

[4] Karthikeyan Bhargavan and Gaëtan Leurent. "On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*. Ed. by Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi. ACM, 2016, pp. 456–467. DOI: 10.1145/2976749.2978423. URL: https://doi.org/10.1145/2976749.2978423.

[5] Eli Biham, Alex Biryukov, and Adi Shamir. "Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials". In: *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*. Ed. by Jacques Stern. Vol. 1592. Lecture Notes in Computer Science. Springer, 1999, pp. 12–23. DOI: 10.1007/3-540-48910-X_2. URL: https://doi.org/10.1007/3-540-48910-X_2.

[6] Eli Biham and Adi Shamir. "Differential Cryptanalysis of DES-like Cryptosystems". In: *J. Cryptol.* 4.1 (1991), pp. 3–72. DOI: 10.1007/BF00630563. URL: https://doi.org/10.1007/BF00630563.

[7] Alex Biryukov and Dmitry Khovratovich. "Related-Key Cryptanalysis of the Full AES-192 and AES-256". In: *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*. Ed. by Mitsuru Matsui. Vol. 5912. Lecture Notes in Computer Science. Springer, 2009, pp. 1–18. DOI: 10.1007/978-3-642-10366-7_1. URL: https://doi.org/10.1007/978-3-642-10366-7_1.

[8] John Black and Phillip Rogaway. "A Block-Cipher Mode of Operation for Parallelizable Message Authentication". In: *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*. Ed. by Lars R. Knudsen. Vol. 2332. Lecture Notes in Computer Science. Springer, 2002, pp. 384–397. DOI: 10.1007/3-540-46035-7_25. URL: https://doi.org/10.1007/3-540-46035-7_25.

[9] Xavier Bonnetain, María Naya-Plasencia, and André Schrottenloher. "Quantum Security Analysis of AES". In: *IACR Trans. Symmetric Cryptol.* 2019.2 (2019), pp. 55–93. DOI: 10.13154/tosc.v2019.i2.55-93. URL: https://doi.org/10.13154/tosc.v2019.i2.55-93.

[10] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. "Boomerang Connectivity Table: A New Cryptanalysis Tool". In: *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10821. Lecture Notes in Computer Science. Springer, 2018, pp. 683–714. DOI: 10.1007/978-3-319-78375-8_22. URL: https://doi.org/10.1007/978-3-319-78375-8_22.

[11] Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. "The Block Cipher Square". In: *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*. Ed. by Eli Biham. Vol. 1267. Lecture Notes in Computer Science. Springer, 1997, pp. 149–165. DOI: 10.1007/BFb0052343. URL: https://doi.org/10.1007/BFb0052343.

[12] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002. ISBN: 3-540-42580-2. DOI: 10.1007/978-3-662-04722-4. URL: https://doi.org/10.1007/978-3-662-04722-4.

[13] Orr Dunkelman, Nathan Keller, and Adi Shamir. "A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony". In: *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*. Ed. by Tal Rabin.

Vol. 6223. Lecture Notes in Computer Science. Springer, 2010, pp. 393–410. DOI: 10.1007/978-3-642-14623-7_21. URL: https://doi.org/10.1007/978-3-642-14623-7_21.

[14] Orr Dunkelman, Nathan Keller, and Adi Shamir. "A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony". In: *J. Cryptol.* 27.4 (2014), pp. 824–849. DOI: 10.1007/s00145-013-9154-9. URL: https://doi.org/10.1007/s00145-013-9154-9.

[15] F. Betül Durak and Serge Vaudenay. "Breaking the FF3 Format-Preserving Encryption Standard over Small Domains". In: *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*. Ed. by Jonathan Katz and Hovav Shacham. Vol. 10402. Lecture Notes in Computer Science. Springer, 2017, pp. 679–707. DOI: 10.1007/978-3-319-63715-0_23. URL: https://doi.org/10.1007/978-3-319-63715-0%5C_23.

[16] Lov K. Grover. "A Fast Quantum Mechanical Algorithm for Database Search". In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*. Ed. by Gary L. Miller. ACM, 1996, pp. 212–219. DOI: 10.1145/237814.237866. URL: https://doi.org/10.1145/237814.237866.

[17] Viet Tung Hoang, Stefano Tessaro, and Ni Trieu. "The Curse of Small Domains: New Attacks on Format-Preserving Encryption". In: *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10991. Lecture Notes in Computer Science. Springer, 2018, pp. 221–251. DOI: 10.1007/978-3-319-96884-1_8. URL: https://doi.org/10.1007/978-3-319-96884-1_8.

[18] Thomas Jakobsen and Lars R. Knudsen. "The Interpolation Attack on Block Ciphers". In: *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*. Ed. by Eli Biham. Vol. 1267. Lecture Notes in Computer Science. Springer, 1997, pp. 28–40. DOI: 10.1007/BFb0052332. URL: https://doi.org/10.1007/BFb0052332.

[19] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. "Breaking Symmetric Cryptosystems Using Quantum Period Finding". In: *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9815. Lecture Notes in Computer Science. Springer, 2016, pp. 207–237. DOI: 10.1007/978-3-662-53008-5_8. URL: https://doi.org/10.1007/978-3-662-53008-5_8.

[20] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. "Quantum Differential and Linear Cryptanalysis". In: *IACR Trans. Symmetric Cryptol.* 2016.1 (2016), pp. 71–94. DOI: 10.13154/tosc.v2016.i1.71-94. URL: https://doi.org/10.13154/tosc.v2016.i1.71-94.

[21] Lars R. Knudsen. "The Security of Feistel Ciphers with Six Rounds or Less". In: *J. Cryptol.* 15.3 (2002), pp. 207–222. DOI: 10.1007/s00145-002-9839-y. URL: https://doi.org/10.1007/s00145-002-9839-y.

[22] Lars R. Knudsen. "Truncated and Higher Order Differentials". In: *Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings*. Ed. by Bart Preneel. Vol. 1008. Lecture Notes in Computer Science. Springer, 1994, pp. 196–211. DOI: 10.1007/3-540-60590-8_16. URL: https://doi.org/10.1007/3-540-60590-8_16.

[23] Lars R. Knudsen and David A. Wagner. "Integral Cryptanalysis". In: *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers*. Ed. by Joan Daemen and Vincent Rijmen. Vol. 2365. Lecture Notes in Computer Science. Springer, 2002, pp. 112–127. DOI: 10.1007/3-540-45661-9_9. URL: https://doi.org/10.1007/3-540-45661-9_9.

[24] Hidenori Kuwakado and Masakatu Morii. "Quantum distinguisher between the 3-round Feistel cipher and the random permutation". In: *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings*. IEEE, 2010, pp. 2682–2685. DOI: 10.1109/ISIT.2010.5513654. URL: https://doi.org/10.1109/ISIT.2010.5513654.

[25] Hidenori Kuwakado and Masakatu Morii. "Security on the quantum-type Even-Mansour cipher". In: *Proceedings of the International Symposium on Information Theory and its Applications, ISITA 2012, Honolulu, HI, USA, October 28-31, 2012*. IEEE, 2012, pp. 312–316. URL: https://ieeexplore.ieee.org/document/6400943/.

[26] Xuejia Lai. "Higher Order Derivatives and Differential Cryptanalysis". In: *Communications and Cryptography, Two Sides of One Tapestry*. Ed. by Richard E. Blahut, Jr. Daniel J. Costello, Ueli Maurer, and Thomas Mittelholzer. Kluwer Academic Publishers, 1994, pp. 227–233. ISBN: 978-1-4613-6159-6.

[27] Xuejia Lai, James L. Massey, and Sean Murphy. "Markov Ciphers and Differential Cryptanalysis". In: *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*. Ed. by Donald W. Davies. Vol. 547. Lecture Notes in Computer Science. Springer, 1991, pp. 17–38. DOI: 10.1007/3-540-46416-6_2. URL: https://doi.org/10.1007/3-540-46416-6_2.

[28] Jung-Keun Lee, Bonwook Koo, Dongyoung Roh, Woo-Hwan Kim, and Daesung Kwon. "Format-Preserving Encryption Algorithms Using Families of Tweakable Blockciphers". In: *Information Security and Cryptology - ICISC 2014 - 17th International Conference, Seoul, Korea, December 3-5, 2014, Revised Selected Papers*. Ed. by Jooyoung Lee and Jongsung Kim. Vol. 8949. Lecture Notes in Computer Science. Springer, 2014, pp. 132–159. DOI: 10.1007/978-3-319-15943-0_9. URL: https://doi.org/10.1007/978-3-319-15943-0_9.

[29] Michael Luby and Charles Rackoff. “How to Construct Pseudorandom Permutations from Pseudorandom Functions”. In: *SIAM J. Comput.* 17.2 (1988), pp. 373–386.

[30] Mitsuru Matsui. “Linear Cryptanalysis Method for DES Cipher”. In: *Advances in Cryptology - EUROCRYPT ’93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*. Ed. by Tor Helleseth. Vol. 765. Lecture Notes in Computer Science. Springer, 1993, pp. 386–397. DOI: 10.1007/3-540-48285-7_33. URL: https://doi.org/10.1007/3-540-48285-7_33.

[31] Sean Murphy. “The Return of the Cryptographic Boomerang”. In: *IEEE Transactions on Information Theory* 57.4 (2011), pp. 2517–2521. DOI: 10.1109/TIT.2011.2111091. URL: https://doi.org/10.1109/TIT.2011.2111091.

[32] Valérie Nachef, Jacques Patarin, and Emmanuel Volte. *Feistel Ciphers - Security Proofs and Cryptanalysis*. Springer, 2017. ISBN: 978-3-319-49528-6. DOI: 10.1007/978-3-319-49530-9. URL: https://doi.org/10.1007/978-3-319-49530-9.

[33] Kaisa Nyberg and Lars R. Knudsen. “Provable Security Against a Differential Attack”. In: *J. Cryptol.* 8.1 (1995), pp. 27–37. DOI: 10.1007/BF00204800. URL: https://doi.org/10.1007/BF00204800.

[34] Claude E. Shannon. “Communication theory of secrecy systems”. In: *Bell Syst. Tech. J.* 28.4 (1949), pp. 656–715. DOI: 10.1002/j.1538-7305.1949.tb00928.x. URL: https://doi.org/10.1002/j.1538-7305.1949.tb00928.x.

[35] Peter W. Shor. “Algorithms for Quantum Computation: Discrete Logarithms and Factoring”. In: *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*. IEEE Computer Society, 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700. URL: https://doi.org/10.1109/SFCS.1994.365700.

[36] Daniel R. Simon. “On the Power of Quantum Computation”. In: *SIAM J. Comput.* 26.5 (1997), pp. 1474–1483. DOI: 10.1137/S0097539796298637. URL: https://doi.org/10.1137/S0097539796298637.

[37] National Institute of Standards and Technology (NIST). *Advanced Encryption Standard (AES)*. Federal Information Processing Standard (FIPS), Publication 197, U.S. Department of Commerce, Washington D.C. Nov. 2001.

[38] National Institute of Standards and Technology (NIST). *Data Encryptions Standard (DES)*. Federal Information Processing Standard (FIPS), Publication 46, U.S. Department of Commerce, Washington D.C. Jan. 1977.

[39] National Institute of Standards and Technology (NIST). *NIST Special Publication 800-38G, Recommendation for Block Cipher Modes of Operation: Methods for Format-Preserving Encryption*. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38G.pdf [accessed 10/04/2018]. Mar. 2016.

[40] Tyge Tiessen. "Polytopic Cryptanalysis". In: *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*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Vol. 9665. Lecture Notes in Computer Science. Springer, 2016, pp. 214–239. DOI: 10.1007/978-3-662-49890-3_9. URL: https://doi.org/10.1007/978-3-662-49890-3_9.

[41] Yosuke Todo. "Integral Cryptanalysis on Full MISTY1". In: *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*. Ed. by Rosario Gennaro and Matthew Robshaw. Vol. 9215. Lecture Notes in Computer Science. Springer, 2015, pp. 413–432. DOI: 10.1007/978-3-662-47989-6_20. URL: https://doi.org/10.1007/978-3-662-47989-6_20.

[42] Yosuke Todo. "Structural Evaluation by Generalized Integral Property". In: *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 287–314. DOI: 10.1007/978-3-662-46800-5_12. URL: https://doi.org/10.1007/978-3-662-46800-5_12.

[43] Yosuke Todo and Masakatu Morii. "Bit-Based Division Property and Application to Simon Family". In: *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*. Ed. by Thomas Peyrin. Vol. 9783. Lecture Notes in Computer Science. Springer, 2016, pp. 357–377. DOI: 10.1007/978-3-662-52993-5_18. URL: https://doi.org/10.1007/978-3-662-52993-5_18.

[44] Serge Vaudenay. "Provable Security for Block Ciphers by Decorrelation". In: *STACS 98, 15th Annual Symposium on Theoretical Aspects of Computer Science, Paris, France, February 25-27, 1998, Proceedings*. Ed. by Michel Morvan, Christoph Meinel, and Daniel Krob. Vol. 1373. Lecture Notes in Computer Science. Springer, 1998, pp. 249–275. DOI: 10.1007/BFb0028566. URL: https://doi.org/10.1007/BFb0028566.

[45] David A. Wagner. "The Boomerang Attack". In: *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*. Ed. by Lars R. Knudsen. Vol. 1636. Lecture Notes in Computer Science. Springer, 1999, pp. 156–170. DOI: 10.1007/3-540-48519-8_12. URL: https://doi.org/10.1007/3-540-48519-8_12.

Part II

Publications

Finding Integral Distinguishers with Ease

Publication Information

Zahra Eskandari, Andreas Brasen Kidmose, Stefan Kölbl, and Tyge Tiessen. "Finding Integral Distinguishers with Ease". In: *Selected Areas in Cryptography - SAC 2018 - 25th International Conference, Calgary, AB, Canada, August 15-17, 2018, Revised Selected Papers*. Ed. by Carlos Cid and Michael J. Jacobson Jr. Vol. 11349. Lecture Notes in Computer Science. https://doi.org/10.1007/978-3-030-10970-7_6. Springer, 2018, pp. 115–138

Contribution

- Main contribution in picking targets for analysis and verifying the results.

Remarks

This publication has been slightly edited to fit the format.

Finding Integral Distinguishers with Ease

Zahra Eskandari[1], Andreas Brasen Kidmose[2], Stefan Kölbl[2,3], and Tyge Tiessen[2]
zahra.eskandari@mail.um.ac.ir, abki@dtu.dk, stek@mailbox.org

[1] Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran
[2] DTU Compute, Technical University of Denmark, Denmark
[3] Cybercrypt, Denmark

Abstract. The division property method is a technique to determine integral distinguishers on block ciphers. While the complexity of finding these distinguishers is higher, it has recently been shown that MILP and SAT solvers can efficiently find such distinguishers. In this paper, we provide a framework to automatically find those distinguishers which solely requires a description of the cryptographic primitive. We demonstrate that by finding integral distinguishers for 30 primitives with different design strategies.

We provide several new or improved bit-based division property distinguishers for ChaCha, Chaskey, DES, GIFT, LBlock, Mantis, Qarma, RoadRunner, Salsa and SM4. Furthermore, we present an algorithm to find distinguishers with lower data complexity more efficiently.

Keywords: Integral Attacks, Division Property, Tool

1 Introduction

Block ciphers, stream ciphers, and hash functions are the fundamental symmetric cryptographic primitives that are at the base of almost all cryptographic protocols. One of the most successful set of techniques to evaluate their security are techniques based on higher-order derivatives.

Higher-order derivatives were first considered in the context of symmetric cryptography by Xuejia Lai [16] and shown by Lars R. Knudsen [14] to attack weaknesses not covered by differential cryptanalysis, and successfully used to break a cipher design [12]. A higher-order derivative in the context of cryptography is the discrete equivalent of higher-order derivatives of multivariate continuous functions. The cryptographic primitive can be seen as a vectorial Boolean function where a higher-order derivative evaluates this function at a given point with respect to some directions/-subspace. Such a derivative can for example be used to find the coefficients of the monomials of the algebraic normal form (ANF) of a cryptographic primitive.

An important category of higher-order attacks is integral cryptanalysis. This type of cryptanalysis appeared first in the Square attack [6], and was later generalised

to be apply to other ciphers as well ([5], [15]). In integral cryptanalysis, the goal is to find a set of input bits and a set of output bits, such that when taking the sum over a set of input messages taking all possible values in the selected input bits and arbitrary but constant values in the other input bits, the sum will be balanced in the selected output bits. This can be described as a higher-order derivative that can be taken at any point and evaluates to zero in the specified output bits.

Originally such property was derived using arguments based on the structure of the primitive but Yosuke Todo demonstrated in his EUROCRYPT 2015 paper [26] a novel method to derive integral distinguishers using the so-called division property formalism whose effectiveness he demonstrated with an attack on full-round Misty [24]. The technique originally being used on words of at least four bits, has since been applied to bit-based designs as well, albeit at a higher computational cost [28].

Another type of higher-order attacks are so-called cube attacks [11], [29]. In these attacks the cryptographic primitive is viewed as a vectorial Boolean function in both public and secret input bits. By finding coefficients of terms in the public bits that are linear in the secret bits, it is possible to derive a set of linear equations that we can solve to extract the secret input bits. This technique has successfully been applied to stream ciphers and hash functions [9], [10].

Contributions

This paper presents a new framework to analyse the security of cryptographic primitives with respect to the bit-based division property by providing a simple way to find distinguishers and testing the number of rounds required for no such distinguisher to exist. We take a look at how finding division property distinguishers can be efficiently automated. To this end, we elaborate how the bit-based division property can be mapped to conditions on the state bits which in turn maps easily to a SAT problem.

Our tool focuses especially on the usability and allows to describe the cryptographic primitives at a high level by providing commonly used operations like S-boxes, linear layers, bit-permutations or modular addition. This completely removes the need of constructing any domain specific models like previous search strategies [21], [30], [31].

In order to demonstrate the usability of our tool we implemented 30 primitives following different design strategies. We then use our tool to find several new integral distinguishers, provide a bound for which number no such distinguishers exist in our model and also evaluate for which design strategies our approach becomes computationally infeasible.

In particular we find the following new results:

- We provide the first bit-based integral distinguishers for the permutations used in ChaCha (6 rounds), Chaskey (4 rounds) and Salsa (6 rounds). We further show that for one more round no distinguisher of this type exists.
- For DES we show that by using the bit-based division property we can improve upon the word-based division property distinguishers by Todo [26] and add one

round. We also show that for 8 rounds no such distinguishers exist.

- We present the first integral distinguisher for both Mantis (3 forward, 2 backward rounds) and several variants of Qarma (2 forward, 2 backward rounds).

- For the SM4 block cipher we can show a distinguisher for 12 rounds and that no bit-based division property distinguisher exists for 13 rounds. This improves the best previously known integral distinguisher by 4 rounds [17].

- We find a distinguisher for 17 rounds of LBlock, which improves the best previously known results by one round [30].

- We present 9-round distinguishers for GIFT-64 which improve upon the data complexity of the distinguishers provided by the designers [1].

- For RoadRunner we are able to extend the distinguishers found by the authors [2] by one additional round.

For several other primitives we provide a bound at which no bit-based division property distinguishers exists in our model. Furthermore, we present an efficient algorithm to find distinguishers with reduced data complexity by only covering the search space which can actually lead to distinguishers.

Software. We place the tool developed for this paper into the public domain and it is available at `https://github.com/kste/solvatore`.

Related Work

The division property has been applied to a large variety of cryptographic primitives and has led to significant improvements [24], [26] over classical integral attacks in some cases. With the extension of the division property to bit-based designs [28] the technique can be applied to a larger class of cryptographic primitives. However finding distinguishers with this approach is a difficult task and requires a lot of effort.

The first automated approach for finding bit-based division property distinguishers was presented in [22] and is based on reducing the problem to mixed integer linear programming (MILP). This simplifies the search for distinguishers and allows to apply the bit-based division property to a larger class of cryptographic primitives. Another automated approach based on constraint programming has been proposed in [23] to find integral distinguishers for Present. In the paper the authors show that this approach can have a better performance than the MILP based technique. The search for ARX and word-based division property has been dealt with in [21] by using SAT resp. SMT solvers.

2 Division property and division trails

The methodology of division properties was devised by Yosuke Todo in his EUROCRYPT 2015 paper [26]. We elaborate this methodology here in the setting where the words are single bits, i.e., when applied as bit-based division property. While using the original formalism, we will look at it from a slightly different angle to simplify the discussion. For the division property over larger word sizes, we refer to the original paper.

2.1 Background

The formalism of division properties belongs to the family of attack vectors collectively named integral cryptanalysis. The goal of integral cryptanalytic techniques is to find a set of input texts such that the sum of the resulting output texts evaluates to zero in some of the bits. If such a property can be found it directly yields a distinguisher which often can be turned into a key recovery attack.

The most common sets of input texts that are used are those that are equal in some bit positions and take all possible combination of values in the remaining bit positions. The first attack that successfully used this attack vector is the Square attack [7] on the block cipher Square that is equally applicable to the Advanced Encryption Standard (AES).

There are two main methods that are used to derive an integral distinguisher: structural properties and algebraic degree bounds. In the Square attack and subsequent generalizations [4] the integral property could be derived by only looking at structural properties of the cipher such as the SPN or Feistel structure without taking much of the cipher details into consideration (such as concrete S-box, concrete linear layer).

Later it was recognised that these kinds of integral distinguishers correspond to discrete derivatives [16] where the derivative is taken with respect to the active input bits, i.e., those that are varied. As such the structural techniques are a way to determine output bits whose polynomial representations do not contain terms that include all active input bits simultaneously. Taking the derivative with respect to these active input bits will thus necessarily evaluate to zero in these output bits.

The second major technique that is used to derive integral distinguishers uses this view of integral distinguishers as derivatives. By determining upper bounds on the algebraic degree of the polynomials of the output bits, we can determine that derivatives of sufficient degree have to evaluate to zero. Similar to the structural method, the methods used to bound the degree usually ignore large parts of the implementation details, for example by just looking at the degree of rounds and multiplying these.

The division property is an improvement with respect to this situation as it manages to take more implementation details of the cipher into consideration. The downside to this is an increased cost of finding the distinguishers.

2.2 Formalism of bit-based division properties

In the bit-based division property methodology, the goal is to find, given a set of chosen active input bits, those output bits whose polynomial representations do not contain terms that feature all of these active bits simultaneously. While this could principally be done by simply calculating the exact polynomial representations of the output bits, this is computationally infeasible in all but toy examples. With division properties we use an approximation instead that guarantees to only find valid distinguishers but might fail to find all distinguishers.

In this approximation, we continually track which bits of the state would need to be multiplied to generate a bit whose polynomial representation can contain terms of all active bits. Let us consider an initial state of four bits (x_0, x_1, x_2, x_3) where we activate bits x_1 and x_2, i.e., we are interested in which state bits we would need to multiply to create a term that contains both bits. For this initial state the minimal way of generating such a term is by multiplying those two bits directly. We write this combination as the choice vector $(0, 1, 1, 0)$.[4]

If we now add x_1 to x_3, we get the new state $(x_0, x_1, x_2, x_3 + x_1)$. Now we can generate a term that contains both x_1 and x_2 in two different minimal ways: first again by multiplying the second and third bit or by multiplying the third and the last bit. These correspond to the choice vectors $(0, 1, 1, 0)$ and $(0, 0, 1, 1)$.[5] The only original choice vector $(0, 1, 1, 0)$ has thus been transformed to two choice vectors by the application of the addition.

If we now applied another operation to this state, each of the choice vectors is transformed to other minimal choice vectors, and by iterating this process a tree of minimal choice vectors is spanned whose final nodes are the minimal choice vectors of output bits whose multiplication can create a term that contains all active input bits.

To determine whether a minimal choice vector can be reached from the initial choice vector of active bits, we need to determine whether a path exists in this tree from the initial choice vector to the output choice vector. We will refer to such path as a *division trail*. In particular, to determine whether a specific output bit is zero when evaluating the derivative with respect to the active bits, we need to determine whether the choice vector that only chooses this output bit is reachable. If it is not reachable, we know that this output bit cannot have terms in its polynomial representation that contain all active bits simultaneously and thus the derivative has to evaluate to zero. Should the choice vector be reachable though, nothing definite can be said about the derivative.

2.3 Rules of choice vector propagation

To trace a division trail of minimal choice vectors, we need to know how these minimal choice vectors of state bits are transformed to new choice vectors under the

[4] In the original paper, this was written slightly more verbosely as $\mathcal{D}^4_{(0,1,1,0)}$.

[5] In the original paper, this would be written as $\mathcal{D}^4_{(0,1,1,0),(0,0,1,1)}$.

application of operations. In the following we will shortly discuss the application of XOR, AND, bit-copying and S-boxes. As the influence of the operations is local, it is sufficient to restrict the discussion to those bits involved in the operation.

Bit-Copying

Let us take a look at the scenario where we have two state bits, and the value of the first bit is copied to the second bit. There are four possible original choice vectors: $(0,0)$, $(1,0)$, $(0,1)$, and $(1,1)$. The first choice vector implies that to generate a term that can contain all active bits, we don't need to multiply any of the two bits. So clearly we still do not need to multiply any of the bits after copying the first bit onto the second, leading to the transition $(0,0) \to (0,0)$.

In the case of $(1,0)$, we need the first bit in the product to generate a term with all active bits but the second one is not required. Thus after copying, we can choose either the first or the second bit (both would also be possible but not minimal). We thus have the two transitions: $(1,0) \to (1,0)$ and $(1,0) \to (0,1)$.

Now in the case of $(1,0)$ and $(1,1)$, the second bit is needed in the product to create a term with all active bits. As it is copied over, it is no longer possible after copying to create this term and thus no valid transitions exist.

XOR

Now for the case where there are two state bits and the first is XORed onto the second. Again we have to look at the four cases $(0,0)$, $(1,0)$, $(0,1)$, and $(1,1)$. As with bit copying, in the case of $(0,0)$, the bits are not necessary in the product, so they are not necessary after the addition as well. This leads to the transition $(0,0) \to (0,0)$.

In the case of $(1,0)$, the first bit value is needed in the product. After the addition, the bit value is also present as part of the sum in the second bit. We can thus either choose the first or the second bit in the product, leading to the transitions $(1,0) \to (1,0)$ and $(1,0) \to (0,1)$.

When we have the case $(0,1)$, the second bit value is needed in the product. As it is still only present in the second bit after the addition, the only valid transition here is $(0,1) \to (0,1)$.

Finally, in the case of $(1,1)$, the product of both bits is needed to create a term with all active bits. Although the second bit contains both original bit values after the addition, it only does so as a sum while we need the product of both. Thus also after the addition, we have to choose both bits, leading to the transition $(1,1) \to (1,1)$.

AND

If we now have again two state bits and we multiply the first onto the second, the situation is analogous to the case of the XOR except if the choice vector before the multiplication is $(1,1)$. In this case the product of both bit values is needed to create

a term of all active bits. As the multiplication creates exactly this product in the second bit, the only minimal transition here is $(1,1) \rightarrow (0,1)$.

S-boxes

The easiest way to see how choice vectors are transformed by an S-box is to look at the polynomial representation of the S-box, i.e., the algebraic normal form (ANF). It is tedious but straightforward to deduce the valid output choice vectors for a given input choice vector using the ANF. It can hence be easily automated and we only need to do this once for an S-box.

3 Solvatore - Automated Finding of Integral Properties

Finding integral distinguishers using division properties is a difficult task. Especially for bit-based designs the analysis often requires extensive manual work which is prone to errors. Automatic tools can be very useful and simplify the analysis of cryptographic primitives, allowing us to explore a larger set of attack vectors. On the other hand they can also be very useful in the design process of cryptographic primitives, to optimise parameters and quickly test different design strategies.

In the following, we present our automated tool Solvatore, which simplifies the search for bit-based division property distinguishers by providing a framework for implementing a large variety of cryptographic primitives. One of the main focuses of the framework is to not only automate finding the bit-based division property distinguishers, as done in previous work [20], [21], [30], but also to completely abstract away the need for dealing with generating models for the primitives or requiring any domain specific knowledge. This makes it much simpler and less error-prone compared to other approaches to add new primitives to the framework and in general it is far easier to implement a primitive in our tool than writing a standard C implementation as many details can be omitted.

Currently our framework supports the following operations to construct cryptographic primitives:

- Bit operations: bit-copying, **and**, and **xor**.
- Arbitrary S-boxes.
- Linear layers using matrix multiplication over arbitrary fields.
- Modular Addition.
- Bit-permutations.
- Generic cell permutations for ShiftRows or Midori-like constructions.

As an example the full description of Present is given in section A which only requires to define the S-box, bit-permutation and on which bits those are applied. In order to analyse the security of Present against the bit-based division property our

tool provides functions for checking whether an output bit is balanced for a given choice vector.

In the following we show how we can reduce the problem of finding a division trail to a satisfiability problem. For this we have to construct a Boolean formula which is satisfiable if and only if it forms a valid division trail.

3.1 Modeling division property propagation with SAT

The *Boolean satisfiability problem* (SAT) is a well known problem from computer science. The problem is to decide whether there exists an assignment of variables in a Boolean formula in conjunctive normal form (CNF) such that the formula evaluates to `true`. While the problem is known to be NP-complete, the SAT instances we will construct here are very structured and can often be solved quickly in practice by modern SAT solvers. In the following we show how to reduce the problem of finding division trails to a SAT problem and how this can be useful in the cryptanalysis of cryptographic primitives.

First, we introduce a variable for each bit of the choice vector $S^i = (s_0, \ldots, s_{n-1})$ after the ith operation applied to the state where n is the size of the state. The next step is to define how the choice vector can propagate through different Boolean functions which occur in the round functions of cryptographic primitives. The rules for this have been explained in subsection 2.3 and have also been studied in [24], [26]. We therefore focus here on how we can construct a Boolean formula in CNF which is SAT if and only if the assignment of the variables forms a valid transition of choice vectors.

Bit-Copying. The **copy** operation copies a bit a to an output bit b, and all valid transitions of choice vectors are given by

$$\begin{aligned} \mathbf{copy}(a_{\text{old}}, b_{\text{old}}) &\to \{(a_{\text{new}}, b_{\text{new}})\} \\ \mathbf{copy}(0,0) &\mapsto \{(0,0)\} \\ \mathbf{copy}(1,0) &\mapsto \{(1,0),(0,1)\}. \end{aligned}$$

The set of clauses C_{copy} which form a Boolean formula which is SAT iff $(a_{\text{old}}, b_{\text{old}}) \xrightarrow{\text{copy}} (a_{\text{new}}, b_{\text{new}})$ is given by

$$\begin{aligned} C_{\text{copy}} = \{&(\neg b_{\text{old}}), (\neg a_{\text{old}} \vee b_{\text{new}} \vee a_{\text{new}}), (a_{\text{old}} \vee \neg b_{\text{new}}), \\ &(a_{\text{old}} \vee \neg a_{\text{new}}), (\neg a_{\text{new}} \vee \neg b_{\text{new}})\}. \end{aligned} \tag{1}$$

And. The **and** operation corresponds to the result of $a \wedge b \rightarrow b$. The valid transitions are given by

$$\begin{aligned}\mathbf{and}(a_{\text{old}}, b_{\text{old}}) &\rightarrow \{(a_{\text{new}}, b_{\text{new}})\} \\ \mathbf{and}(0,0) &\mapsto \{(0,0)\} \\ \mathbf{and}(0,1) &\mapsto \{(0,1)\} \\ \mathbf{and}(1,0) &\mapsto \{(1,0),(0,1)\} \\ \mathbf{and}(1,1) &\mapsto \{(0,1)\}.\end{aligned}$$

Just as for the **copy** operation, translating this to a SAT sentence is straightforward and gives the following set of clauses

$$\begin{aligned}C_{\text{and}} = \{&(a_{\text{old}} \vee \neg a_{\text{new}}), (\neg b_{\text{old}} \vee b_{\text{new}}), (\neg b_{\text{new}} \vee \neg a_{\text{new}}), \\ &(\neg a_{\text{old}} \vee b_{\text{new}} \vee a_{\text{new}}), (a_{\text{old}} \vee b_{\text{old}} \vee \neg b_{\text{new}}).\end{aligned} \tag{2}$$

Xor. The **xor** operation corresponds to the result of $a \oplus b \rightarrow b$. The valid transitions are given by

$$\begin{aligned}\mathbf{xor}(a_{\text{old}}, b_{\text{old}}) &\rightarrow \{(a_{\text{new}}, b_{\text{new}})\} \\ \mathbf{xor}(0,0) &\mapsto \{(0,0)\} \\ \mathbf{xor}(0,1) &\mapsto \{(0,1)\} \\ \mathbf{xor}(1,0) &\mapsto \{(1,0),(0,1)\} \\ \mathbf{xor}(1,1) &\mapsto \{(1,1)\}\end{aligned}$$

which corresponds to the following clauses

$$\begin{aligned}C_{\text{xor}} = \{&(a_{\text{old}} \vee \neg a_{\text{new}}), (\neg b_{\text{old}} \vee b_{\text{new}}), (b_{\text{old}} \vee \neg b_{\text{new}} \vee \neg a_{\text{new}}), \\ &(\neg a_{\text{old}} \vee a_{\text{new}} \vee b_{\text{new}}), (b_{\text{old}} \vee a_{\text{old}} \vee \neg b_{\text{new}}), \\ &(\neg b_{\text{old}} \vee \neg a_{\text{old}} \vee a_{\text{new}}).\end{aligned} \tag{3}$$

S-boxes. As described in subsection 2.3, the transition rules for S-boxes can easily be deduced automatically. The rules create a truth table for involved variables which can be transformed to a CNF using standard methods.

Linear Layers. Many popular designs, like the AES, use a complex linear layer in order to get good diffusion. These linear layers are often represented as $d \times d$ matrices over some field $\mathbb{F}_2^k$. In order to model the trail propagation we can represent these transformations as $kd \times kd$ matrices over $\mathbb{F}_2$, which then can be decomposed into the basic **copy** and **xor** operations.

In order to simplify the description of such linear layers in our tool, we implemented this decomposition and it is only required to provide the irreducible polynomial for the field $\mathbb{F}_2^k$ and the matrix. From the irreducible polynomial it is possible to deduce the $k \times k$ matrices that represent the elements of $\mathbb{F}_k$ as matrices over $\mathbb{F}_2$. Substituting these matrices in the original matrix over $\mathbb{F}_k$ now creates the $nk \times nk$ binary matrix.

Modular Addition. Modular addition is used as a non-linear component in ARX-ciphers like HIGHT, LEA, and SPECK. We can use the same approach as [20] to decompose the modular addition into `xor` and `and`. Let z, x, y be n bit-variables with z_i, y_i, x_i as the ith bits, counting from the least significant bit, and $z = x \boxplus y$. The modular addition modulo 2^n is given by:

$$\begin{aligned} &z_i = x_i \oplus y_i \oplus c_i \\ &\text{where} \\ &c_i = x_{i-1}y_{i-1} \oplus (x_{i-1} \oplus y_{i-1})c_{i-1} \text{ for } i > 0 \\ &c_0 = 0 \end{aligned}$$

So far we have assumed that both x, y are variables, however in some ciphers one of them is a constant, e.g. a round key. Since we can ignore `xor` and `and` with a constant we get the following expressions.

$$\begin{aligned} &z_i = x_i \oplus c_i \\ &\text{where} \\ &c_i = x_{i-1} \oplus x_{i-1}c_{i-1} \text{ for } i > 0 \\ &c_0 = 0 \end{aligned}$$

Similar, if we want to find a distinguisher on a cipher like Bel-T or the inverse of an ARX-cipher we also need modular subtraction. To do modular subtraction we can use the fact that

$$x \boxminus y = x \boxplus (-y) = x \boxplus (2^n - y) = x \boxplus ((2^n - 1) - y) \boxplus 1 = x \boxplus \overline{y} \boxplus 1 \qquad (4)$$

Since the NOT operation has no effect on whether a bit is balanced or not we can omit it to get $x \boxminus y = x \boxplus y \boxplus 1$. This means that we can do modular subtraction with one modular addition and one constant addition.

3.2 Finding integral distinguishers

In order to find useful integral properties of a cipher, we have to propagate an initial choice vector S^0 and check whether it is impossible to reach certain choice vectors S^r after r rounds. If we can show that an output choice vector that is everywhere zero except for a single **1** in one bit is unreachable, we know that this bit has to be balanced.

In particular we are often interested in whether any bit in the output will be balanced. This corresponds to showing that at least one of the vectors in the set

$$S^r \in \{w \in \mathbb{F}_2^n \mid \mathbf{hw}(w) = 1\}. \qquad (5)$$

is unreachable, where $\mathbf{hw}(x)$ is the Hamming weight of the vector.

Contrarily, we can also use this approach to show the absence of a bit-based division property distinguisher in our model. Checking all possible options for the starting choice vector would be (for most primitives) computationally infeasible. Fortunately it is sufficient to show for all starting choice vectors in the set

$$S^0 \in \{w \in \mathbb{F}_2^n \mid \mathbf{hw}(w) = n - 1\}. \tag{6}$$

that all choice vectors in the set in Equation 5 are reachable. This works because the balancedness of the output bits is preserved when we exchange the input choice vector with any vector greater than it (with respect to the above ordering).

We will use the following notation to simplify the description of the distinguishers found later in the paper. The set of *active* bits will be denoted as

$$A = \{i \mid S_i^0 = 1,\ \ i = 0, \ldots, n-1\} \tag{7}$$

and correspondingly the set of *constant* bits as

$$\overline{A} = \overline{\{i \mid S_i^0 = 1,\ \ i = 0, \ldots, n-1\}} = \{i \mid S_i^0 = 0,\ \ i = 0, \ldots, n-1\}. \tag{8}$$

The set of bits which are balanced at the output is denoted as B. We can now describe a distinguisher, for a function f, as

$$A \xrightarrow{f} B. \tag{9}$$

If a valid division trail from A to B exists we will also use the more compact notation $\mathbf{DP}(A) = B$ if the function is clear from context.

Note that while the notation for the set of active bits at the input and the balanced bits at the output looks very similar it conveys a very different meaning in the context of the division property. For a range of bits $s_i, s_{i+1}, \ldots, s_j$ we will use the notation s_{i-j}.

4 Distinguishers and Bounds

We implemented a variety of cryptographic primitives in Solvatore to demonstrate the versatility of our tool and the ease of adding primitives with different design principles.

- **SPN**: GIFT, LED, Midori, Photon, Present, Skinny, Spongent
- **ARX**: BelT, ChaCha, Chaskey, Lea, Hight, Salsa, Sparx, Speck
- **Feistel**: DES, LBlock, Misty, RoadRunner, Skipjack, SM4, Twine
- **Reflection**: Mantis, Prince, Qarma
- **Bit-sliced**: Ascon, Rectangle

- **LFSR-based**: Bivium, Trivium, Kreyvium

We will first go over the general methodology and after that over the results on the different primitive classes obtained using Solvatore. This includes both bit-based division property distinguishers and finding the number of rounds at which no such distinguisher exists anymore. All results have been obtained on an Intel Core i7-4770S running Ubuntu 17.10 using the Python interface to CryptoMiniSat 5.0.1. Several examples for distinguishers we found are given in section B.

4.1 Methodology

Finding a Bound. As a first step we try to find the number of rounds r^* at which no bit-based division property distinguisher in our model exists. This is done by testing all set of active bits of type

$$A_j = \{i \mid i \in \mathbb{Z}_n \setminus j\} \quad \forall j \in \mathbb{Z}_n. \tag{10}$$

This corresponds to all vectors where a single bit is constant. If for all possible choices the set of balanced bits $B_j = \mathbf{DP}(A_j)$ is empty we know that no such distinguisher exists for r^* rounds.

Reducing Data Complexity. In order to reduce the data complexity for the distinguishers covering the most rounds we use different strategies. The naive approach would be to increase the number of constant bits c, try out all possible combinations and check whether the resulting set of balanced bits B is not empty. This might work in some cases however the complexity increases very quickly as we have to test all $\binom{n}{c}$ possible choices.

This can be improved by only testing those combinations of constant bits which can actually lead to non-empty sets B. First, we compute the set of constant bits

$$G_1 = \{j \mid \mathbf{DP}(A_j) = B_j \wedge (|B_j| > 0) \quad \forall j \in \mathbb{Z}_n\} \tag{11}$$

for which at least one of the bits after r rounds is balanced, similar to the case where we try to find the bound. Next, we look at all combinations of two elements of G_1 which share at least one balanced bit

$$G_2 = \{\{i, j\} \mid (i \neq j) \wedge (|\mathbf{DP}(A_i) \cap \mathbf{DP}(A_j)|) > 0, \forall i, j \in G_1\}. \tag{12}$$

We can continue the last step in a similar way until G_i is empty by testing all combinations of the sets of bits in G_i repeatedly. Note that in the next step we would not have single indices but sets of indices and we therefore look whether the union of these sets of constant bits lead to a non-empty set B. Another advantage of this approach is that we only need to test those bits for the balancedness property which were already balanced in the last iteration.

In each step the elements in G_i are a set of constant bits which will have at least one balanced bit in the output after r rounds. This approach improves the complexity

Table 1: Results from the optimised search for SPONGENT-88. Combinations are the number of pairs (i, j) in the sets G_i which share bits in their corresponding sets B_i and B_j.

	G_1	G_2	G_3	G_4	G_5
Size ($\|G\|$)	43	40	25	1	0
Combinations	878	643	234	0	-

of finding distinguishers with lower data complexity significantly, but often it is still computationally infeasible to find an optimal distinguisher. For more structured designs it often helps to look at the word level and only look at maximizing the number of constant words as there are fewer combinations which we have to check.

4.2 SPN

We will use 9 rounds of SPONGENT-88 as an example to show the benefits of the optimised search for a distinguisher with lower data complexity. In order to estimate the complexity we will count for how many choice vectors we would have to compute the set of balanced bits B. Using the optimised search we only have to test 1819 choice vectors (see Table 1) to find distinguishers with up to 4 constant bits and exclude any distinguisher with 5 constant bits. Using the naive approach we would have to test 679120 choice vectors to find all distinguishers up to 4 bits and check $\binom{128}{5}$ combinations to exclude the existence of any further distinguishers.

For SKINNY-64 we can find a distinguisher with the same data complexity as the one given by the authors [3] with one additional balanced bit and show that no distinguishers exist for 11 rounds.

For GIFT-64 we use our optimal approach and no better distinguisher exists. We can find a 9-round distinguisher similar to the one by the authors [1], but also distinguishers with a lower data complexity. For GIFT-128 finding distinguishers takes significantly longer and we were only able to find a distinguisher with high data complexity similar to the original one.

For several variants of PHOTON we can find distinguishers with low data complexity by searching for combinations of constant words. However for more rounds the search time increases quickly and we are not able to improve any results. The complex linear layer generates a large number of clauses which seems to be the main limiting reason.

4.3 ARX

First we look at the permutation used in the CHASKEY MAC [18]. We can find a distinguisher for 3 rounds with only two constant words, one with high complexity for 4 rounds and show that no bit-based division property distinguishers for 5 rounds

Table 2: Overview of our distinguishers and bounds for SPN-based designs.

Cipher	Rounds	Active Bits	Balanced Bits
GIFT-64	9	61	5
	9	62	11
	9	63	30
	10	No Distinguisher	
GIFT-128	11	127	32
	12	No Distinguisher	
LED	5	60	64
	8	No Distinguisher	
Midori- 64	6	48	16
	8	No Distinguisher	
Midori- 128	5	104	128
Photon-100	4	12	100
	5	99	100
Photon-144	4	24	144
Photon-196	4	28	196
Photon-256	4	32	256
Present	9	60	1
	10	No Distinguisher	
Skinny-64	10	48	9
	11	No Distinguisher	
Spongent-88	9	84	3
	9	87	54
	10	No Distinguisher	
Spongent-136	10	132	8
	10	135	93
	11	No Distinguisher	
Spongent-176	12	No Distinguisher	

exist. This confirms the claim by the authors that Chaskey is likely to resist this type of attacks. Considering the construction used for the MAC it seems infeasible to mount an attack based on the 4-round distinguisher.

The large state of Salsa and ChaCha make it difficult to adopt our approach for reducing the data complexity. We therefore keep whole words constants and try to find the maximum number. For 6 rounds of Salsa the only distinguisher which exists keeps the first word constant and the one for ChaCha has only a single constant bit. In both cases no distinguisher exists for 7 rounds. On the actual mode

in which Salsa and ChaCha are used as a stream cipher we can only control the 64-bit nonce in a single block. In this setting there are no bit-based division property distinguisher for 4 rounds of Salsa and 2 rounds of ChaCha.

We can also confirm the results from [21] using our optimal search algorithm for Hight, Lea and Speck. We noticed that Solvatore performs significantly better for finding these distinguishers even though we use the same SAT solver. It only took us 28/195/51 seconds compared to 15/30/6 minutes for finding the optimal distinguishers for Hight/Lea/Speck. This gap could be explained by the slightly different model resp. using a better search strategy.

Bel-T is a block cipher which has been adopted as a national standard in the Republic of Belarus and combines S-boxes with modular addition. There is only a very limited amount of cryptanalysis available [13] (also provides an English description of the algorithm). We provide the first analysis with respect to integral attacks for Bel-T and can find a fairly efficient distinguisher for 2 rounds while showing that none exist for 3 rounds.

In the case of Sparx we can confirm the results by the authors [8]. The full summary of the results for ARX-based primitives can also be found in Table 3.

4.4 Feistel

For DES we improve the best bit-based division property distinguisher [26] by one round. The original distinguisher for DES also uses the division property but only word-based which makes this improvement possible.

One of the most successful applications of the division property is the full break of Misty [25]. It is also based on the analysis on the word level so one might suspect that it can be improved by looking at the bit-based division property. We tried to find the same distinguishers as in the original attack automatically however the complexity seems too high without further optimizations. We could only find a distinguisher for 3 rounds.

The best integral distinguisher on SM4 covers 8 rounds [17]. By using the bit-based division property we can improve those distinguishers to 12 rounds, although at a high complexity. We further can show that no such distinguishers exist for 13 rounds.

In the case of LBlock we are able to extend the distinguisher found with MILP [30] by one additional round and for Roadrunner we can find a 5-round distinguisher which also covers one more round than the best known distinguisher [2].

For all variants of Simon and Simeck we can reproduce the results from [30], show that these have the lowest data complexity and that there are no distinguisher in our model for more rounds.

4.5 Reflection

Block ciphers based on the reflection design strategy, introduced by Prince, are a popular choice for low-latency designs. We will denote the number of rounds as $f + b$,

Table 3: Overview of our distinguishers and bounds for ARX-based designs.

Cipher	Rounds	Active Bits	Balanced Bits
ChaCha	6	511	138
	7	No Distinguisher	
Chaskey	3	64	6
	4	127	5
	5	No Distinguisher	
Lea	8	126	16
	8	118	1
	9	No Distinguisher	
Hight	18	63	2
	19	No Distinguisher	
Salsa	6	480	129
	7	No Distinguisher	
Speck-32	6	31	1
	7	No Distinguisher	
Speck-48	6	45	1
	7	No Distinguisher	
Speck-64	6	61	1
	7	No Distinguisher	
Speck-96	6	93	1
	7	No Distinguisher	
Speck-128	6	125	1
	7	No Distinguisher	
BelT	2	45	5
	3	No Distinguisher	
Sparx-64	3	32	32
	4	No Distinguisher	
Sparx-128	4	96	64
	5	No Distinguisher	

where f are the rounds before the middle layer and b the rounds after the middle layer (see Table 5).

For Prince we can find a bit-based division property distinguisher with the same complexity as the best higher-order differential given in [19] and show that for one additional round none exist. Very similar distinguisher also exist for Mantis with the only difference being that one can extend those by one round in forward and backwards direction. The distinguishers for Qarma can cover a similar number of

Table 4: Overview of our distinguishers and bounds for Feistel networks.

Cipher	Rounds	Active Bits	Balanced Bits
DES	7	60	8
	8	No Distinguisher	
LBlock	17	63	4
	18	No Distinguisher	
Misty	3	32	64
RoadRunner	5	58	8
	6	No Distinguisher	
Skipjack	19($A^8B^8A^3$)	47	16
	20($A^8B^8A^4$)	56	8
	21($A^8B^8A^5$)	No Distinguisher	
Simon32	14	31	16
	15	No Distinguisher	
Simon48	16	47	24
	17	No Distinguisher	
Simon64	18	63	22
	19	No Distinguisher	
Simon96	22	95	5
	23	No Distinguisher	
Simon128	26	127	3
	27	No Distinguisher	
Simeck32	15	31	7
	16	No Distinguisher	
Simeck48	18	47	5
	19	No Distinguisher	
Simeck64	21	63	5
	22	No Distinguisher	
SM4	12	126	32
	13	No Distinguisher	
Twine	16	63	32
	17	No Distinguisher	

rounds although at a much higher data complexity.

Table 5: Results on reflection ciphers.

Cipher	Rounds	Active Bits	Balanced Bits
Mantis	$2+2$	12	16
	$3+2$	32	16
	$3+3$	No Distinguisher	
Prince	$1+1$	12	64
	$2+1$	32	64
	$1+2$	32	64
	$2+2$	No Distinguisher	
Qarma-64/σ_0	$2+2$	48	16
	$3+3$	No Distinguisher	
Qarma-64/σ_1	$2+2$	52	64
	$3+3$	No Distinguisher	
Qarma-64/σ_2	$2+2$	52	64
	$3+3$	No Distinguisher	
Qarma-128/σ_0	$2+2$	96	128
	$3+3$	No Distinguisher	
Qarma-128/σ_1	$2+2$	96	128
	$3+3$	No Distinguisher	
Qarma-128/σ_2	$2+2$	120	128
	$3+3$	No Distinguisher	

Table 6: Results on bit-sliced ciphers.

Cipher	Rounds	Active Bits	Balanced Bits
Ascon	5	16	320
Rectangle	9	60	
	10	No Distinguisher	

4.6 Bit-sliced

In this category we look at two LS-designs (see Table 6). The permutation used in the authenticated encryption scheme Ascon and the block cipher Rectangle. For Ascon we can improve the data complexity of the 5 round distinguisher [26] by a factor of 4, however for more rounds we could not improve any results as the computations takes too long. For Rectangle we are able to show that no distinguisher exists for 10 rounds and find the already known 9-round distinguisher from [30].

Table 7: Results on LFSR-based stream ciphers.

Cipher	Rounds	Active Bits	Balanced Bits
Bivium	681	79	1
Trivium	707	79	1
Kreyvium	713	127	1

4.7 LFSR-based

We looked at three LFSR-based stream ciphers which share a similar structure. The active bits are taken over the choice of IV and our distinguishers here checks whether the output bit of the key stream is balanced after r rounds. It is very likely that there are more bits balanced in the state, but we can only distinguish the key stream if the resulting key stream bit is also balanced.

While we could find some distinguishers the time it takes to find a balanced output bit of the keystream quickly increases and other approaches seem to be more promising for constructing distinguisher based on the division property for this type of ciphers [27].

4.8 Overview

Using Solvatore we were able to demonstrate several new distinguishers, reduce the data complexity and show at which number of rounds a primitive becomes resistant against bit-based division property. In Figure 1 we give an overview of the number of rounds required before no bit-based division property distinguisher exists in relation to the full number of rounds of the primitive. It can be seen that most ciphers provide a fairly large security margin against these type of attacks and also for many of these designs there are indeed better distinguishers based on other techniques like differential and linear cryptanalysis.

The performance of Solvatore varies a lot from the designs and for some it is not feasible to find good distinguishers. For instance we also implemented both AES and Keccak in our tool, but we could only obtain very limited results which could not improve upon the state-of-the-art.

5 Conclusion and Future Work

In this work we presented a new framework to automatically find division property distinguishers for a large class of cryptographic primitives by reducing the problem to SAT. We also provide a cryptanalysis tool implementing this approach, providing a simple way to describe primitives, allowing both designers and cryptanalysts to evaluate cryptographic primitives against this attack vector.

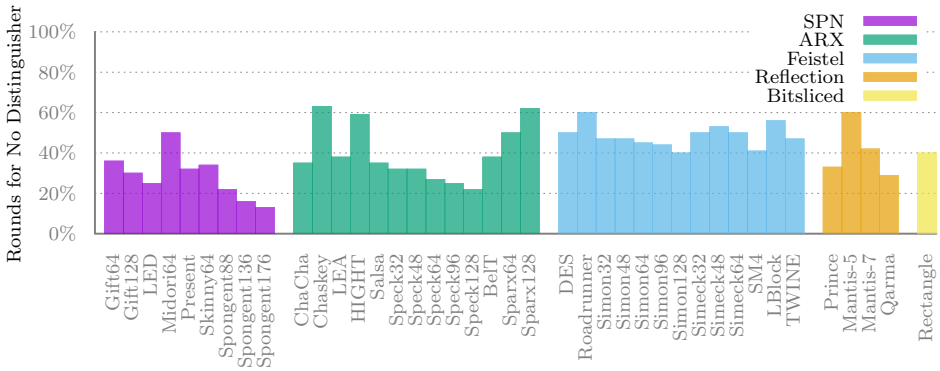


Figure 1: Overview of the fraction of rounds required before we can show that no bit-based division property distinguishers exist in our model.

Using this tool we present several new or improved bit-based division property distinguishers for ChaCha, Chaskey, DES, GIFT, LBlock, Mantis, Qarma, RoadRunner, Salsa and SM4.

Furthermore, we provide an improved algorithm for finding distinguisher with an optimal data complexity and show for several primitives that no bit-based division property distinguisher can exist for more rounds.

References

[1] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. "GIFT: A Small Present - Towards Reaching the Limit of Lightweight Encryption". In: *CHES*. Vol. 10529. Lecture Notes in Computer Science. Springer, 2017, pp. 321–345.

[2] Adnan Baysal and Sühap Sahin. "RoadRunneR: A Small and Fast Bitslice Block Cipher for Low Cost 8-Bit Processors". In: *LightSec*. Vol. 9542. Lecture Notes in Computer Science. Springer, 2015, pp. 58–76.

[3] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. "The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS". In: *CRYPTO (2)*. Vol. 9815. Lecture Notes in Computer Science. Springer, 2016, pp. 123–153.

[4] Alex Biryukov and Adi Shamir. "Structural Cryptanalysis of SASAS". In: *Advances in Cryptology - EUROCRYPT 2001*. Ed. by Birgit Pfitzmann. Vol. 2045. Lecture Notes in Computer Science. Springer, 2001, pp. 394–405. ISBN: 3-540-42070-3.

[5] Alex Biryukov and Adi Shamir. “Structural Cryptanalysis of SASAS”. In: *Journal of Cryptology* 23.4 (2010), pp. 505–518.

[6] Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. “The Block Cipher Square”. In: *Fast Software Encryption, FSE ’97*. Ed. by Eli Biham. Vol. 1267. Lecture Notes in Computer Science. Springer, 1997, pp. 149–165. ISBN: 3-540-63247-6.

[7] Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. “The Block Cipher Square”. In: *Fast Software Encryption, FSE ’97*. Ed. by Eli Biham. Vol. 1267. Lecture Notes in Computer Science. Springer, 1997, pp. 149–165. ISBN: 3-540-63247-6.

[8] Daniel Dinu, Léo Perrin, Aleksei Udovenko, Vesselin Velichkov, Johann Großschädl, and Alex Biryukov. “Design Strategies for ARX with Provable Bounds: Sparx and LAX”. In: *ASIACRYPT (1)*. Vol. 10031. Lecture Notes in Computer Science. 2016, pp. 484–513.

[9] Itai Dinur, Pawel Morawiecki, Josef Pieprzyk, Marian Srebrny, and Michal Straus. “Cube Attacks and Cube-Attack-Like Cryptanalysis on the Round-Reduced Keccak Sponge Function”. In: *Advances in Cryptology - EUROCRYPT 2015*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 733–761. ISBN: 978-3-662-46799-2.

[10] Itai Dinur and Adi Shamir. “Breaking Grain-128 with Dynamic Cube Attacks”. In: *Fast Software Encryption, FSE 2011*. Ed. by Antoine Joux. Vol. 6733. Lecture Notes in Computer Science. Springer, 2011, pp. 167–187. ISBN: 978-3-642-21701-2.

[11] Itai Dinur and Adi Shamir. “Cube Attacks on Tweakable Black Box Polynomials”. In: *Advances in Cryptology - EUROCRYPT 2009*. Ed. by Antoine Joux. Vol. 5479. Lecture Notes in Computer Science. Springer, 2009, pp. 278–299. ISBN: 978-3-642-01000-2.

[12] Thomas Jakobsen and Lars R. Knudsen. “The Interpolation Attack on Block Ciphers”. In: *Fast Software Encryption, FSE ’97*. Ed. by Eli Biham. Vol. 1267. Lecture Notes in Computer Science. Springer, 1997, pp. 28–40. ISBN: 3-540-63247-6.

[13] Philipp Jovanovic and Ilia Polian. “Fault-based attacks on the Bel-T block cipher family”. In: *DATE*. ACM, 2015, pp. 601–604.

[14] Lars R. Knudsen. “Truncated and Higher Order Differentials”. In: *Fast Software Encryption: Second International Workshop. 1994*. Ed. by Bart Preneel. Vol. 1008. Lecture Notes in Computer Science. Springer, 1995, pp. 196–211.

[15] Lars R. Knudsen and David Wagner. “Integral Cryptanalysis”. In: *Fast Software Encryption,FSE 2002*. Ed. by Joan Daemen and Vincent Rijmen. Vol. 2365. Lecture Notes in Computer Science. Springer, 2002, pp. 112–127. ISBN: 3-540-44009-7.

[16] Xuejia Lai. “Higher Order Derivatives and Differential Cryptanalysis”. In: *Communications and Cryptography, Two Sides of One Tapestry*. Ed. by Richard E. Blahut, Jr. Daniel J. Costello, Ueli Maurer, and Thomas Mittelholzer. Kluwer Academic Publishers, 1994, pp. 227–233. ISBN: 978-1-4613-6159-6.

[17] Fen Liu, Wen Ji, Lei Hu, Jintai Ding, Shuwang Lv, Andrei Pyshkin, and Ralf-Philipp Weinmann. “Analysis of the SMS4 Block Cipher”. In: *ACISP*. Vol. 4586. Lecture Notes in Computer Science. Springer, 2007, pp. 158–170.

[18] Nicky Mouha, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel, and Ingrid Verbauwhede. “Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers”. In: *Selected Areas in Cryptography*. Vol. 8781. Lecture Notes in Computer Science. Springer, 2014, pp. 306–323.

[19] Shahram Rasoolzadeh and Håvard Raddum. “Faster Key Recovery Attack on Round-Reduced PRINCE”. In: *LightSec*. Vol. 10098. Lecture Notes in Computer Science. Springer, 2016, pp. 3–17.

[20] Ling Sun, Wei Wang, Ru Liu, and Meiqin Wang. *MILP-Aided Bit-Based Division Property for ARX-Based Block Cipher*. Cryptology ePrint Archive, Report 2016/1101. `http://eprint.iacr.org/2016/1101`. 2016.

[21] Ling Sun, Wei Wang, and Meiqin Wang. *Automatic Search of Bit-Based Division Property for ARX Ciphers and Word-Based Division Property*. Cryptology ePrint Archive, Report 2017/860. `https://eprint.iacr.org/2017/860`. 2017.

[22] Ling Sun, Wei Wang, and Meiqin Wang. “MILP-Aided Bit-Based Division Property for Primitives with Non-Bit-Permutation Linear Layers”. In: *IACR Cryptology ePrint Archive* 2016 (2016), p. 811. URL: `http://eprint.iacr.org/2016/811`.

[23] Siwei Sun, David Gerault, Pascal Lafourcade, Qianqian Yang, Yosuke Todo, Kexin Qiao, and Lei Hu. “Analysis of AES, SKINNY, and Others with Constraint Programming”. In: *IACR Transactions on Symmetric Cryptology* 2017.1 (2017).

[24] Yosuke Todo. “Integral Cryptanalysis on Full MISTY1”. In: *Advances in Cryptology - CRYPTO 2015*. Ed. by Rosario Gennaro and Matthew Robshaw. Vol. 9215. Lecture Notes in Computer Science. Springer, 2015, pp. 413–432. ISBN: 978-3-662-47988-9.

[25] Yosuke Todo. “Integral Cryptanalysis on Full MISTY1”. In: *J. Cryptology* 30.3 (2017), pp. 920–959.

[26] Yosuke Todo. “Structural Evaluation by Generalized Integral Property”. In: *Advances in Cryptology - EUROCRYPT 2015*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 287–314. ISBN: 978-3-662-46799-2.

[27] Yosuke Todo, Takanori Isobe, Yonglin Hao, and Willi Meier. "Cube Attacks on Non-Blackbox Polynomials Based on Division Property". In: *CRYPTO (3)*. Vol. 10403. Lecture Notes in Computer Science. Springer, 2017, pp. 250–279.

[28] Yosuke Todo and Masakatu Morii. "Bit-Based Division Property and Application to Simon Family". In: *Fast Software Encryption, FSE 2016*. Ed. by Thomas Peyrin. Vol. 9783. Lecture Notes in Computer Science. Springer, 2016, pp. 357–377. ISBN: 978-3-662-52992-8.

[29] Michael Vielhaber. "Breaking ONE.FIVIUM by AIDA an Algebraic IV Differential Attack". In: *IACR Cryptology ePrint Archive* 2007 (2007), p. 413. URL: http://eprint.iacr.org/2007/413.

[30] Zejun Xiang, Wentao Zhang, Zhenzhen Bao, and Dongdai Lin. "Applying MILP Method to Searching Integral Distinguishers Based on Division Property for 6 Lightweight Block Ciphers". In: *Advances in Cryptology - ASIACRYPT 2016*. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Vol. 10031. Lecture Notes in Computer Science. 2016, pp. 648–678. ISBN: 978-3-662-53886-9.

[31] Wenying Zhang and Vincent Rijmen. *Division Cryptanalysis of Block Ciphers with a Binary Diffusion Layer*. Cryptology ePrint Archive, Report 2017/188. https://eprint.iacr.org/2017/188. 2017.

A Implementation of Present

The following example shows how one can implement the PRESENT cipher in our framework to analyse its properties against bit-based division property attacks.

```
from cipher_description import CipherDescription

present_sbox = [0xC, 0x5, 0x6, 0xB, 0x9, 0x0, 0xA, 0xD,
                0x3, 0xE, 0xF, 0x8, 0x4, 0x7, 0x1, 0x2]
present_permutations = [\
  ['s1', 's16', 's4'], ['s2', 's32', 's8'],
  ['s3', 's48', 's12'], ['s5', 's17', 's20'],
  ['s6', 's33', 's24'], ['s7', 's49', 's28'],
  ['s9', 's18', 's36'], ['s10', 's34', 's40'],
  ['s11', 's50', 's44'], ['s13', 's19', 's52'],
  ['s14', 's35', 's56'], ['s15', 's51', 's60'],
  ['s22', 's37', 's25'], ['s23', 's53', 's29'],
  ['s26', 's38', 's41'], ['s27', 's54', 's45'],
  ['s30', 's39', 's57'], ['s31', 's55', 's61'],
  ['s43', 's58', 's46'], ['s47', 's59', 's62']]

present = CipherDescription(64)
present.add_sbox('S-box', present_sbox)
```

```
for i in range(16):
    bits = ["s{}".format(4*i + 0),
            "s{}".format(4*i + 1),
            "s{}".format(4*i + 2),
            "s{}".format(4*i + 3)]
    present.apply_sbox('S-box', bits, bits)
for p in present_permutations:
    present.apply_permutation(p)
```

Using this description of the PRESENT block cipher we can mount our analysis. The following code checks whether no bit-based division property distinguisher exists for 10 rounds of PRESENT.

```
from itertools import combinations
from solvatore import Solvatore
from cipher_description import CipherDescription
from ciphers import present

cipher = present.present
rounds = 10

solver = Solvatore()
solver.load_cipher(cipher)
solver.set_rounds(rounds)

# Look over all combination for one non active bit
for bits in combinations(range(64), 1):
    nonactive_bits = bits
    active_bits = {i for i in range(64)
                   if i not in nonactive_bits}

    # Find all balanced bits
    balanced_bits = []
    for i in range(cipher.state_size):
        if solver.is_bit_balanced(i, rounds, active_bits):
            balanced_bits.append(i)

    if len(balanced_bits) > 0:
        print("Found distinguisher!")
        print(active_bits, balanced_bits)
```

B Overview of Distinguishers

In the following we list some of the new distinguishers we found.

B.1 ChaCha

$$\overline{\{0\}} \xrightarrow{\text{9-round}} \{32-68, 192-223, 352-415, 424-428\} \tag{13}$$

B.2 Chaskey

$$\overline{\{64-127\}} \xrightarrow{\text{3-round}} \{80-85\} \tag{14}$$

$$\overline{\{96\}} \xrightarrow{\text{4-round}} \{80-81\} \tag{15}$$

B.3 DES

$$\overline{\{50-52, 63\}} \xrightarrow{\text{7-round}} \{0, 3, 9, 10, 18, 19, 25, 28\} \tag{16}$$

B.4 GIFT-64

$$\overline{\{0-2\}} \xrightarrow{\text{9-round}} \{3, 7, 27, 43, 59\} \tag{17}$$

B.5 LBlock

$$\overline{\{34\}} \xrightarrow{\text{17-round}} \{2, 3, 30, 31\} \tag{18}$$

B.6 Mantis

$$\overline{\{0-7, 16-23, 40-47, 56-63\}} \xrightarrow{3+2 \text{ rounds}} \begin{aligned}\{&2, 6, 10, 14, 18, 22, 26, 30\\ &34, 38, 42, 46, 50, 54, 58, 62\}\end{aligned} \tag{19}$$

B.7 Qarma

QARMA-64/σ_0

$$\overline{\{0-3, 20-23, 40-43, 60-63\}} \xrightarrow{2+2 \text{ rounds}} \begin{aligned}\{&1, 5, 9, 13, 17, 21, 25, 29,\\ &33, 37, 41, 45, 49, 53, 57, 61\}\end{aligned} \tag{20}$$

QARMA-64/σ_1

$$\overline{\{0-3, 20-23, 40-43\}} \xrightarrow{2+2 \text{ rounds}} \{0-63\} \tag{21}$$

QARMA-64/σ_2

$$\overline{\{0-3, 20-23, 40-43\}} \xrightarrow{2+2 \text{ rounds}} \{0-63\} \tag{22}$$

QARMA-128/σ_0

$$\overline{\{0-15, 32-47\}} \xrightarrow{2+2 \text{ rounds}} \{0-127\} \tag{23}$$

QARMA-128/σ_1

$$\overline{\{0-15, 32-47\}} \xrightarrow{2+2 \text{ rounds}} \{0-127\} \tag{24}$$

QARMA-128/σ_2

$$\overline{\{0-7\}} \xrightarrow{2+2 \text{ rounds}} \{0-127\} \tag{25}$$

B.8 RoadRunner

$$\overline{\{0,1,8,9,16,17\}} \xrightarrow{5\text{-round}} \{32,33,40,41,48,49,56,57\} \tag{26}$$

B.9 Salsa

$$\overline{\{0-31\}} \xrightarrow{6\text{-round}} \{128-255, 295\} \tag{27}$$

B.10 SM4

$$\overline{\{96,97\}} \xrightarrow{12\text{-round}} \{0-31\} \tag{28}$$

On Quantum Distinguishers for Type-3 Generalized Feistel Network Based on Separability

Publication Information

Samir Hodžić, Lars Ramkilde Knudsen, and Andreas Brasen Kidmose. "On Quantum Distinguishers for Type-3 Generalized Feistel Network Based on Separability". In: *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020, Paris, France, April 15-17, 2020, Proceedings*. Ed. by Jintai Ding and Jean-Pierre Tillich. Vol. 12100. Lecture Notes in Computer Science. Springer, 2020, pp. 461–480. DOI: 10.1007/978-3-030-44223-1_25. URL: https://doi.org/10.1007/978-3-030-44223-1_25

Contribution

- Main contributions in development of ideas, test implementations, figures, and writing.

Remarks

This publication has been slightly edited to fit the format.

On quantum distinguishers for Type-3 Generalized Feistel network based on separability

S. Hodžić[1], L. R. Knudsen[1], and A. B. Kidmose[1]

Technical University of Denmark, DTU Compute, Denmark, saho,lrkn,abki@dtu.dk

Abstract. In this work, we derive a method for constructing quantum distinguishers for GFNs (Generalized Feistel-like schemes with invertible inner functions and XORs), where for simplicity 4 branches are considered. The construction technique is demonstrated on Type-3 GFN, where some other cyclically inequivalent GFNs are considered as examples. Introducing the property of separability, we observe that finding a suitable partition of input blocks implies that some branches can be represented as a sum of functions with almost disjoint variables, which simplifies the application of Simon's algorithm. However, higher number of rounds in most of the cases have branches which do not satisfy the previous property, and in order to derive a quantum distinguisher for these branches, we employ Simon's and Grover's algorithm in combination with a suitable system of equations given in terms of input blocks and inner functions involved in the round function. As a result, we are able to construct a 5-round quantum distinguisher for Type-3 GFNs using only a quantum encryption oracle with query complexity $2^{N/4} \cdot \mathcal{O}(N/4)$, where N size of the input block.

Keywords: Simon's algorithm Grover's algorithm Generalized Feistel network Quantum cryptanalysis

1 Introduction

The interest in post-quantum cryptanalysis of block ciphers has significantly increased in the last decade after H. Kuwakado and M. Morii [17] proposed a polynomial time quantum distinguisher for 3-round Feistel Network based on Simon's algorithm [25]. The current state of quantum cryptanalysis of block ciphers encompasses several approaches which range from quantifying the known classical attacks, to applying the quantum algorithms which result in significant reduction of attack complexities, such as Simon's and Grover's [11] algorithms. In various settings, these two algorithms have been utilized in many recent works [3], [4], [5], [6], [8], [9], [10], [12], [14], [15], [16], [17], [19], [21], [23], [24], [27]. In general, the overall attention of researchers has been further motivated by total/partial breaks of several ciphers (or proofs that certain schemes provide much less security than expected) such as Even-Mansour [18],

AEZ cipher [3], LED cipher [27], FX construction [4], [19], and certain authenticated encryption schemes [16].

The previous works demonstrate that Simon's and Grover's algorithm can be utilized for key-recovery attacks and construction of quantum distinguishers. The main idea is to construct a function, f, which has a hidden shift, that can be recovered using Simon's algorithm. The shift can then either reveal the secret key or be used to distinguish the cipher from a random permutation. A problem that may arise is that f may have unwanted collisions, however, Kaplan et al. [16] showed this can be solved by performing sufficient measurements.

An interesting application of combining Simon's and Grover's algorithm has been shown in [19] on the FX construction, where G. Leander and A. May showed that key-whitening method does not provide the same increase of the key space as in the classical environment. The combination of these two algorithms has been used in the context of amplitude amplification technique derived by G. Brassard et al. [7]. The work [19] initialized the series of papers [4], [8], [9], [10], [15] which have been combining Simon's and Grover's algorithm in a similar way, resulting in quantum distinguishers and key recovery attacks.

In this paper, we derive a method for constructing quantum distinguishers based on Simon's/Grover's algorithm for Generalized Feistel networks. Recently, iterated classical Feistel schemes in combination with advanced slide attacks have been analysed in [8]. A distinguisher for 5-round Feistel networks is provided in [10], which is then turned into a key-recovery attack with the Simon/Grover combination. Moving to more general schemes, Dong et al. [9] provides a polynomial time quantum distinguisher for Type-1 GFN with d branches on $2d - 1$ rounds. In the same work, the authors provide a $(2d + 1)$-round polynomial time quantum distinguisher for $2d$-branch Type-2 GFN. Ito and Iwata [15] improved the Type-1 distinguisher to cover $3d - 3$ rounds.

We notice that the previous distinguishers are based on finding a suitable partition of the input block, such that, some branch can be written as a sum of functions with almost disjoint variables. Here, 'almost' means that one of the functions contain one specific variable that is not involved in other functions, and 'disjoint' refers to its suitable placement in the context of Simon's algorithm. We call this property *separability* and refer to Definition 4 for the details. The construction of f, that we use in our work, is based on the concatenation of two suitable functions whose values are obtained by querying the encryption oracle in superposition and truncating the output. Therefore, the attacks belong to the so-called **Q2** attack model. According to A. Hosoyamada and Y. Sasaki [13, Section 5.2] truncation is possible in the quantum world, i.e., it has been shown that computing a truncated function can be done as efficiently as computing the complete function.

However, the propagation of separability, which we consider in strong and weak settings, is limited throughout the cipher. In order to demonstrate our approach, we analyze the Type-3 GFN [28], which has not been addressed in previous works. We notice that one can consider a specific set of terms which is actually present in almost the whole cipher, which gives rise to a system of equations given in terms

of input blocks and inner functions employed in the definition of underlying GFN. The solvability of this system, in combination with Simon's and Grover's algorithm, then implies suitable parameters upon which the distinguisher is constructed. For instance, we show that for 5-round Type-3 GFN one needs $2^n \cdot \mathcal{O}(n)$ query complexity and $2n + 2(2n + 1)(n + 1 + \sqrt{n + 1})$ qubits, in order to extract suitable parameters required for a distinguisher. Here, n is the branch size of a given GFN, since for simplicity we fix the number of branches to be 4 for all GFNs that we consider.

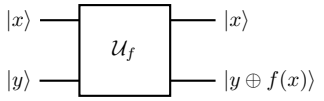
In general, we note that a successful construction of a distinguisher (by our method) highly depends on the solvability of the observed system of equations, along with the employed technique for constructing the Simon's function f. Recall that in our work the concatenation method is used, but other methods can be utilized as well. Further consideration of the presented methods on other GFNs in terms of the design and security, is left as an open problem.

This paper is organized as follows. In Section 2 we provide an overview of Simon's and Grover's algorithm (along with the basic notation). In Section 3 we demonstrate the presence of the separability property in 3-round Feistel cipher, Type-1 and Type-2 GFNs, which we formalize by considering three different types, namely (semi-) strong and weak separability. In the same section we analyze the Type-3 GFN (precisely, the 4-th branch of 5-th round). In Section 4 we employ Simon's and Grover's algorithm to derive necessary parameters that will enable us to construct a quantum distinguisher. Some concluding remarks are given in Section 5.

2 Preliminaries

The vector space $\mathbb{F}_2^n$ is the space of all n-tuples $x = (x_1, \ldots, x_n)$, where $x_i \in \mathbb{F}_2$. For $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ in $\mathbb{F}_2^n$, the usual scalar (or dot) product over $\mathbb{F}_2$ is defined as $x \cdot y = x_1 y_1 \oplus \cdots \oplus x_n y_n$, where by '$\oplus$' we denote the modulo 2 computation in $\mathbb{F}_2^n$. A qubit is a superposition of the classical basis states, $|0\rangle$ and $|1\rangle$, i.e. $\mu|0\rangle + \nu|1\rangle$ $(\mu, \nu \in \mathbb{C})$ with $|\mu|^2 + |\nu|^2 = 1$, which represents the normalization condition. A quantum register is a collection of n qubits, and formally we denote it as $|x\rangle = |c_1\rangle \otimes \ldots \otimes |c_n\rangle$, where $c_i \in \mathbb{F}_2$ (and thus $x \in \mathbb{F}_2^n$). The process of measuring a quantum state, say $|\vartheta\rangle$ (which is a superposition of classical states $|x\rangle$, $x \in \mathbb{F}_2^n$) is a non-reversible physical process in which we collapse the state $|\vartheta\rangle$ into a classical state.

Since all computations in the quantum circuit are done in the Hilbert space, it is necessary that any operator (and every single gate) implemented in the quantum environment is invertible. To implement a function $f : \mathbb{F}_2^n \to \mathbb{F}_2^r$ quantumly, one uses an ancilla register (i.e., a register with auxiliary qubits), say y, as



Here, $\mathcal{U}_f$ denotes the quantum oracle (as unitary operator) that provides the values of the function f. If $|y\rangle$ is the all-zero register of size τ, then the oracle $\mathcal{U}_f$ actually provides the mapping $|x\rangle \to |f(x)\rangle$, where the output $|x\rangle$ after the evaluation of $\mathcal{U}_f$ is simply neglected.

In relation to quantum algorithms described later on, we will use the Hadamard transform $H^{\otimes n}$, $n \geq 1$ (also known as the Sylvester-Hadamard matrix), which is defined recursively as

$$H^{\otimes 1} = 2^{-1/2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}; \quad H^{\otimes n} = 2^{-1/2} \begin{pmatrix} H^{\otimes(n-1)} & H^{\otimes(n-1)} \\ H^{\otimes(n-1)} & -H^{\otimes(n-1)} \end{pmatrix}.$$

Throughout the article, by $|\mathbf{0}\rangle$ we will denote the all-zero quantum register whose size will be clear from the context. Throughout the work we will use the following notation related to GFNs.

Notations:

x_j^{r-1} the input of j-th branch in r-th round, $r \geq 1$, $j \geq 1$.

$x_j^0 = x_j$ the input to j-th branch (first round). At some places x_j denote input variables to a function, which is clear from the context.

$F_i^r = F_i^r(k_r, \cdot)$ the inner function (in the round function) at r-th round of a GFN, which involves the secret round key k_r.

$F_i = F_i^1$ the inner functions in the first round. Only in relation (5), F^j denotes a single inner function (in the round function) which involves a secret round key.

$RF_r^{(t)}$ the function which denotes the t-th branch at r-th round of a GFN.

2.1 A brief overview of Simon's and Grover's algorithm

Simon's algorithm [25] solves the following problem in polynomial time.

Simon's problem: Given a function $f : \mathbb{F}_2^n \to \mathbb{F}_2^\tau$ ($\tau \geq 1$) and the promise that there exists a vector $s \in \mathbb{F}_2^n$ such that for all $(x, y) \in \mathbb{F}_2^n \times \mathbb{F}_2^n$ it holds that

$$f(x) = f(y) \;\Leftrightarrow\; x \oplus y \in \{\mathbf{0}, s\}, \tag{1}$$

the goal is to find s.

The property (1) means that for an arbitrary vector $x \in \mathbb{F}_2^n$ the value $f(x)$ will repeat *only* when the input x is shifted by s (the case when $y = x$ is trivial). The value s is called the *shift* or the *period*. Classically, one can find s in time complexity $\mathcal{O}(2^{n/2})$. However, with the following quantum algorithm it is possible to solve this problem with quantum complexity $\mathcal{O}(n)$.

Simon's algorithm [25]:

1) Prepare the state $2^{-n/2}\sum_{x\in\mathbb{F}_2^n}|x\rangle|\mathbf{0}\rangle$, where the second all-zero register is of size τ ($x \in \mathbb{F}_2^n$).

2) Apply the operator $\mathcal{U}_f$ which implements the function $f : \mathbb{F}_2^n \to \mathbb{F}_2^\tau$ in order to obtain the state $2^{-n/2}\sum_{x\in\mathbb{F}_2^n}|x\rangle|f(x)\rangle$.

3) Measuring the second register (the one with values of f), the previous state is collapsed to $|\Omega_a|^{-1/2}\sum_{x\in\Omega_a}|x\rangle$, where $\Omega_a = \{x \in \mathbb{F}_2^n : f(x) = a\}$, for some $a \in Im(f)$.

4) Apply the Hadamard transform $H^{\otimes n}$ to the state $|\Omega_a|^{-1/2}\sum_{x\in\Omega_a}|x\rangle$ in order to obtain $|\varphi\rangle = |\Omega_a|^{-1/2}2^{-n/2}\sum_{y\in\mathbb{F}_2^n}\sum_{x\in\Omega_a}(-1)^{x\cdot y}|y\rangle$.

5) Measure the state $|\varphi\rangle$:

   i) If f does not have any period, the output of the measurement are random values $y \in \mathbb{F}_2^n$.

   ii) If f has a period s, the output of measurement are vectors y which are strictly orthogonal to s, since the amplitudes of y are given by $2^{-(n+1)/2}\sum_{x\in\Omega_a}(-1)^{x\cdot y} = 2^{-(n+1)/2}[(-1)^{x'\cdot y} + (-1)^{(x'\oplus s)\cdot y}]$, where we use the assumption that $\Omega_a = \{x', x' \oplus s\}$ ($x' \in \mathbb{F}_2^n$, $f(x') = f(x' \oplus s) = a$), for any $a \in \mathbb{F}_2^n$.

6) If f has a period, repeat the previous steps until one collects $n-1$ linearly independent vectors y_i. Then, solve the homogeneous system of equations $y_i \cdot s = 0$ (for collected values y_i) in order to extract the unique period s.

Throughout the paper, a Simon's function f will be constructed using the well-known concatenation technique which is given as follows.

Proposition 1. *Let the function $f : \mathbb{F}_2 \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be defined as*

$$f(b,x) = \begin{cases} g(x), & b = 0 \\ h(x), & b = 1 \end{cases}, \tag{2}$$

where $g, h : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Then:

i) If s is a period of both g and h, then $f(b, x \oplus s) = f(b, x)$ holds for all $(b, x) \in \mathbb{F}_2 \times \mathbb{F}_2^n$, i.e., f has period $(0, s) \in \mathbb{F}_2 \times \mathbb{F}_2^n$.

ii) If $h(x \oplus s) = g(x)$ holds for all $x \in \mathbb{F}_2^n$, then f has period $(1, s) \in \mathbb{F}_2 \times \mathbb{F}_2^n$.

Remark 2.1. Note that using the same computation as in the proof of Lemma 1 in [17], one can show that if $g(x)$ and $h(x) = g(x \oplus s)$ are permutations in Proposition 1-(*ii*), then the period $(1, s)$ of f is unique.

Example 2. [17] analyses the distinguishability of 3-round Feistel cipher (Figure 1), and in Section III-A the function $f : \mathbb{F}_2 \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ is defined by

$$f(b,x) = \begin{cases} F_2(x \oplus F_1(\alpha)) \oplus (\alpha \oplus \beta), & b = 0 \\ F_2(x \oplus F_1(\beta)) \oplus (\alpha \oplus \beta), & b = 1 \end{cases}, \tag{3}$$

where $\alpha, \beta \in \mathbb{F}_2^n$ are different fixed vectors, where the restrictions of the function $f(b,x)$ are utilizing the second branch $y_1 = x_0 \oplus F_2(x_1 \oplus F_1(x_0))$.

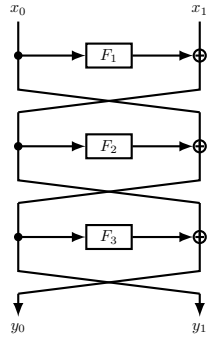


Figure 1: The 3-round Feistel network with keyed inner functions F_i.

Denoting by $g(x) = f(0,x)$ and $h(x) = f(1,x)$, it is easily verified that $h(x \oplus (F_1(\alpha) \oplus F_1(\beta))) = g(x)$, and thus f has the shift $(1,s) = (1, F_1(\alpha) \oplus F_1(\beta)) \in \mathbb{F}_2 \times \mathbb{F}_2^n$, and thus this construction corresponds to the case (ii) of Proposition 1.

Remark 2.2. Note that several works use the idea of Proposition 1 to construct the Simon's function f, e.g. [8], [16], [27]. On the other hand, there are constructions which are not based on the concatenation method, such as the analysis of the Even-Mansour and LRW ciphers in [16].

Another quantum algorithm that will be used in our work is Grover's algorithm [11] which finds a target vector in a given set of vectors, without assuming any structure on the set. Formally, the following problem is considered.

Grover's problem: Let X denote the search set whose elements are represented on $\lceil \log_2 |X| \rceil$ qubits, such that the superposition $\sum_{x \in X} |x\rangle$ is computable in $\mathcal{O}(1)$. Given oracle access to a function $\mathcal{B} : X \to \mathbb{F}_2$, called the classifier, find $x \in X$ such that $\mathcal{B}(x) = 1$.

If there are 2^u preimages of 1, then the procedure of the Grover's algorithm is based on applying approximately $\sqrt{|X|/2^u}$ times the operator Q (which queries $\mathcal{U}_\mathcal{B}$ as a subroutine), in order to amplify the target amplitudes towards the preimages of 1. If $\mathcal{B}$ has a value 1 only at one vector x, then Grover's algorithm runs in time $\mathcal{O}(\sqrt{|X|})$.

Our method of constructing a quantum distinguisher for GFNs will be based on the combination of Simon's and Grover's algorithm (similarly to [9], [10], [19]) in framework of the following result derived by G. Brassard et al. [7].

Theorem 3. *[7] Let $\mathcal{A}$ be any quantum algorithm on q qubits that uses no measurement. Let $\mathcal{B} : \mathbb{F}_2^q \to \mathbb{F}_2$ be a function that classifies outcomes of $\mathcal{A}$ as good or bad. Let $p > 0$ be the initial success probability that a measurement of $\mathcal{A}|\boldsymbol{0}\rangle$ is good. Set $t = \lceil \frac{\pi}{4\theta} \rceil$, where is defined via $\sin^2(\theta) = p$. Moreover, define the unitary operator $Q = -\mathcal{A}S_0\mathcal{A}^{-1}S_{\mathcal{B}}$, where the operator $S_{\mathcal{B}}$ changes the sign of the good state*

$$|x\rangle \to \begin{cases} -|x\rangle, & \text{if } \mathcal{B}(x) = 1 \\ |x\rangle, & \text{if } \mathcal{B}(x) = 0 \end{cases},$$

while S_0 changes the sign of the amplitude only for the zero state $|\boldsymbol{0}\rangle$. Then after the computation of $Q^t\mathcal{A}|\boldsymbol{0}\rangle$, a measurement yields good with probability at least $\max\{p, 1-p\}$.

3 Analysis of GFNs

In this section, we present a new approach for constructing distinquishers for GFNs. In Section 3.1 we illustrate the main idea behind the quantum distinguishers constructed for Type-1 and Type-2 GFNs. Introducing the property of separability, which is the main focus of our approach, in Section 3.2 we analyze the Type-3 scheme with 4 branches. Later on in Section 4 we present our approach in full detail.

3.1 On quantum distinquishers for Type-1 and Type-2 GFN

Recall that quantum distinquishers for classical 3-round Feistel network were presented in [17], and for Type-1/Type-2 GFNs have been constructed in [9], [15] (see also [8], [10], [14]). In this section we discuss the main idea behind the distinguishers on these schemes.

Let us consider the classical 3-round Feistel network (Figure 1), denoted by $FN_3(x_0, x_1) = (y_0, y_1)$, where y_0 and y_1 are given by

$$\begin{cases} y_0 = x_1 \oplus F_1(x_0) \oplus F_3(x_0 \oplus F_2(x_1 \oplus F_1(x_0))), \\ y_1 = x_0 \oplus F_2(x_1 \oplus F_1(x_0)). \end{cases} \tag{4}$$

As shown in [17], by fixing the input x_0 two times, first by α and then by β, then using the right branch of FN_3 one can construct a function $f : \mathbb{F}_2 \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ as in (3), and consequently verify that $f(b, x) = f(b \oplus 1, x \oplus s)$ holds, where $s = F_1(\alpha) \oplus F_1(\beta)$. Moreover, they showed that $(1, s)$ is the only non-trivial period of f. Since this is highly unlikely to hold for a random permutation (FN_3 is proved to be indistinguishable from a random permutation in classical environment under exponential number of queries), this automatically provides a quantum distinguisher which is based on Simon's algorithm.

Similarly, for d-branch Type-1 GFN (CAST256-like scheme), whose r-th round function is given by Figure 2, where x_j^{r-1} is j-th branch in the input for r-th round, one can construct the function $f : \mathbb{F}_2 \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ [9] by

$$f : (b, x) \to \alpha_b \oplus x_1^{2d-1} = F^d(h(\alpha_b) \oplus x), \tag{5}$$

where $(x_0^{2d-1}, \ldots, x_{d-1}^{2d-1}) = E_k(\alpha_b, x)$ and

$$h(\alpha_b) = F^{d-1}(F^{d-2}(\ldots(F^2(F^1(\alpha_b) \oplus x_1^0) \oplus x_2^0) \ldots \oplus x_{d-3}^0) \oplus x_{d-2}^0),$$

with blocks $x_0^0, \ldots, x_{d-2}^0$ taken to be constants and $x_{d-1}^0 = x$. It is not difficult to see that the period of the function f is given by $(1, s) = (1, h(\alpha_0) \oplus h(\alpha_1)) \in \mathbb{F}_2 \times \mathbb{F}_2^n$, which provides the quantum distinguisher for $2d - 1$ rounds based on Simon's algorithm. In a similar way, in [9] the distinguisher of Type-2 GFN (RC6/CLEFIA-like scheme) for $2d + 1$ rounds has been provided for $2d$ branches.

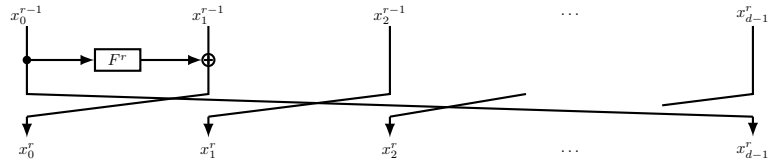


Figure 2: The round function of Type-1 GFN with d branches involving keyed function F^r.

An improvement of the distinguisher (5) has been given in [15], where the same idea of fixing suitable inputs results in a periodic function f has been applied to $(3d - 3)$ and $(d^2 - d + 1)$-round Type-1 GFN. In general, our main observation on previously mentioned distinguishers is that the construction of the Simon's function f is based on the property that we call *separability*, which is defined as follows.

Definition 4. Let $RF_r^{(t)} : (x_0, \ldots, x_{d-1}) \in \mathbb{F}_2^{dn} \to \mathbb{F}_2^n$ denotes the t-th branch at round r of a given GFN. In addition, assume that the input plaintext $(x_0, \ldots, x_{d-1})$ can be written with disjoint variables as $(x_0, \ldots, x_{d-1}) = (x, y, z) \in \mathbb{F}_2^{e_1} \times \mathbb{F}_2^{e_2} \times \mathbb{F}_2^{e_3}$ $(e_1 + e_2 + e_3 = dn,\ e_1 \geq 1)$. Then:

i) The GFN satisfies the *strong separability* property if $RF_r^{(t)}(x_0, \ldots, x_{d-1})$ can be represented as

$$RF_r^{(t)}(x_0, \ldots, x_{d-1}) = RF_r^{(t)}(x, y, z) = \lambda(y, z) \oplus G(x \oplus H(y, z)),$$

where λ is a known function (i.e., implementable by adversary[1]).

[1]Or due to existence of an oracle to provide its values, and being available to be queried by any inputs.

ii) The GFN satisfies the *semi-strong separability* property if

$$RF_r^{(t)}(x_0, \ldots, x_{d-1}) = RF_r^{(t)}(x, y, z) = y \oplus \lambda(z) \oplus G(x \oplus H(y), z),$$

where λ is not available (to be implemented or queried).

iii) The GFN satisfies the *weak separability* property if $RF_r^{(t)}(x, y, z) = \lambda(y, z) \oplus G(x \oplus H(y, z), y, z)$ and λ eventually is not available (e.g. due to involvement of secret keys, or non-existence of the oracle to provide its values).

In all three cases, $G, H : \mathbb{F}_2^n \to \mathbb{F}_2^n$ are some vectorial functions such that G depends on the whole term '$x \oplus H(\cdot)$' (i.e., x is appearing *only* in the term '$x \oplus H(\cdot)$' and nowhere else outside of this term), the function $H(\cdot)$ depends on variables y and/or z only (depending on the case), but not on x.

Remark 3.1. Note that in general, we are not necessarily limited to only three disjoint variables x, y, z, but more can be employed. It just appears that the separability of many GFNs schemes is captured by Definition 4 which uses three disjoint variables.

To provide more clarification, we remark the following points:

- In the case of FN_3 (construction (3)), we notice that the variable $x = x_1$ is appearing only in the term '$x_1 \oplus F_1(x_0)$' and nowhere else outside of this term. Here we consider that $(x_0, x_1) = (y, x)$ and $H(y) = F_1(y)$. Considering y_1 in relation (4), we see that FN_3 satisfies the strong separability, since $y = x_0$ is appearing in $H(y)$ and also outside of it as a summand (where $G = F_2$ and λ is the identity mapping in variable y only).
- In (5) we have that the input block $x_d^0 = x$ is appearing only in the term '$x \oplus h(x_1^0, \ldots, x_{d-1}^0) = x \oplus H(y)$', i.e., it is not involved in the definition of the function H, nor in any other term outside of this one.

In general, if a given GFN satisfies the strong or semi-strong separability property, then one can always construct a Simon's function directly by querying the quantum encryption oracle with suitable constant/variable input blocks x, y, z and applying Proposition 1-(ii). We have the following results.

Proposition 5. *Let $RF_r^{(t)} : (x_0, \ldots, x_{d-1}) \in \mathbb{F}_2^{dn} \to \mathbb{F}_2^n$ ($x_i \in \mathbb{F}_2^n$) denotes the output of the t-th branch at round r of a given GFN, which satisfies the semi-strong separability, that is $RF_r^{(t)}(x_0, \ldots, x_{d-1}) = RF_r^{(t)}(x, y, z) = y \oplus \lambda(z) \oplus G(x \oplus H(y), z)$, for some functions $G, H, \lambda : \mathbb{F}_2^n \to \mathbb{F}_2^n$ (λ is not available to the adversary). Then the Simon's function $f : \mathbb{F}_2 \times \mathbb{F}_2^n$ defined by*

$$f(b, x) = \begin{cases} \alpha_1 \oplus RF_r^{(t)}(x, \alpha_0, \beta) & b = 0 \\ \alpha_0 \oplus RF_r^{(t)}(x, \alpha_1, \beta) & b = 1 \end{cases},$$

has the period $(1, s) = (1, H(\alpha_0) \oplus H(\alpha_1))$, were α_b and β are fixed and known.

Proposition 6. *Let $RF_r^{(t)} : (x_0, \ldots, x_{d-1}) \in \mathbb{F}_2^{dn} \to \mathbb{F}_2^n$ ($x_i \in \mathbb{F}_2^n$) denotes the output of the t-th branch at round r of a given GFN, which satisfies the strong separability, that is $RF_r^{(t)}(x_0, \ldots, x_{d-1}) = RF_r^{(t)}(x, y, z) = \lambda(y, z) \oplus G(x \oplus H(y, z))$, for some functions $G, H, \lambda : \mathbb{F}_2^n \to \mathbb{F}_2^n$ (λ is available to the adversary). Then the Simon's function $f : \mathbb{F}_2 \times \mathbb{F}_2^n$ defined by*

$$f(b, x) = \begin{cases} \lambda(\alpha_1, \beta) \oplus RF_r^{(t)}(x, \alpha_0, \beta) & b = 0 \\ \lambda(\alpha_0, \beta) \oplus RF_r^{(t)}(x, \alpha_1, \beta) & b = 1 \end{cases},$$

has the period $(1, s) = (1, H(\alpha_0, \beta) \oplus H(\alpha_1, \beta))$, were α_b and β are fixed and known.

Recall that $RF_r^{(t)}(\alpha_b, x)$ are obtained by querying and truncating the encryption oracle $\mathcal{U}_{GFN}$, and thus we assume that f can be implemented. Note that in the context of [4] and the asymmetric-query approach (related to **Q2** attack model), we are not able to reduce the query complexity, since our function f is using the quantum encryption oracle $\mathcal{U}_{GFN}$ for both restrictions of the function $f(b, x)$.

Remark 3.2. Proposition 6 shows that semi-strong separability is in fact strong separability (regardless of wether λ is known or not in the representation of $RF_r^{(t)}$). Essentially, this is due to the fact that the functions H and λ are disjoint variable functions, i.e., they do not depend on same variables.

In the next subsection we analyze the Type-3 GFN and present the main idea of our approach (which is further described in details in Section 4) that applies to certain rounds which satisfy the weak separability property.

3.2 On separability of the Type-3 GFN

In general, one can notice that for all GFNs provided on [2, Figure 3] (and other schemes that one can find, for instance in [1], [22]) the weak separability property is appearing after several initial rounds only, and thus makes the application of Simon's algorithm very difficult. In what follows, we consider 4 rounds of the so-called Type-3 scheme which satisfies the weak separability property, and we demonstrate a new approach that potentially allows us to analyze a somewhat increased number of rounds (at some branches).

Recall that the Type-3 GFN is given by Figure 3, where the round function of Type-3 GFN (shortly RF) is defined as

$$RF : (x_0, \ldots, x_3) \to (x_1 \oplus F_1^r(x_0), x_2 \oplus F_2^r(x_1), x_3 \oplus F_3^r(x_2), x_0), \tag{6}$$

where $x_i \in \mathbb{F}_2^n$ and $F_i^r : \mathbb{F}_2^n \to \mathbb{F}_2^n$.

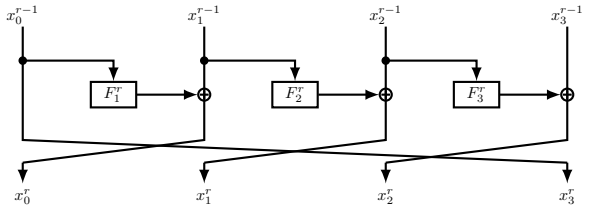


Figure 3: The round function of Type-3 GFN.

Notice that, if we consider the 4-th branch of 4-round Type-3 (shortly GFN_3), which is given by

$$RF_4^{(4)}(x_0,\ldots,x_3) = \\ x_3 \oplus \underbrace{F_1^3(x_2 \oplus F_1^2(x_1 \oplus F_1(x_0)) \oplus F_2(x_1)) \oplus F_2^2(x_2 \oplus F_2(x_1)) \oplus F_3(x_2)}_{\text{Does not involve } x_3},$$

then it actually satisfies the strong separability. This observation allows that we fix blocks (x_0, x_1, x_2) to be constants, say to take two arbitrary (different) values $\alpha_0 = (\alpha_0^{(0)}, \alpha_1^{(0)}, \alpha_2^{(0)})$, $\alpha_1 = (\alpha_0^{(1)}, \alpha_1^{(1)}, \alpha_2^{(1)}) \in \mathbb{F}_2^{3n}$, and set $x_3 = x$. This way (having an encryption oracle), we can construct the functions $g_0(x) = RF_4^{(4)}(\alpha_0, x)$ and $g_1(x) = RF_4^{(4)}(\alpha_1, x)$ and consequently the function $f : \mathbb{F}_2 \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ defined by

$$f(b,x) = \begin{cases} g_0(x), & b = 0 \\ g_1(x), & b = 1 \end{cases}, \tag{7}$$

is a periodic function in vector $(1, s) = (1, H(\alpha_0) \oplus H(\alpha_1))$, where $H(x_0, x_2, x_3) = x_3 \oplus RF_4^{(4)}(x_0, \ldots, x_3)$. Note that the periodicity follows from Proposition 1-(ii), since $g_0(x \oplus s) = g_1(x)$ holds for all $x \in \mathbb{F}_2^n$. Thus we obtain a quantum distinguisher for 4 rounds of GFN_3 with polynomial-time complexity, due to the possibility of applying Simon's algorithm directly. In general, at some other branches of RF_4 one can apply the same idea by utilizing some other suitable inputs.

Remark 3.3. Note that in the case of Type-3 GFN with d-branches, we have that the branch $RF_d^{(d)}(x_0, \ldots, x_{d-1})$ can be written as $RF_d^{(d)}(x_0, \ldots, x_{d-1}) = x_{d-1} \oplus G(x_0, \ldots, x_{d-2})$, for some function G given in terms of inner functions F_i^r, where $i, r \in \{1, \ldots, d-1\}$. This means that $RF_d^{(d)}$ satisfies the strong separability, and thus one can construct Simon's function f similarly as in (7) providing the attack complexity $\mathcal{O}(n)$.

Attacking more rounds: If we consider the GFN_3 with many rounds (say ≥ 5), then we are facing the weak separability property. Let us consider the 5-round (4-th branch), which is given by

$$\begin{aligned} RF_5^{(4)}(x_0,\ldots,x_3) = x_0 \oplus F_3^2(x_3 \oplus F_3(x_2)) \oplus F_2^3[x_3 \oplus F_3(x_2) \oplus F_2^2(x_2 \oplus F_2(x_1))] \\ \oplus F_1^4[x_3 \oplus F_3(x_2) \oplus F_1^3(x_2 \oplus F_2(x_1) \oplus F_1^2(x_1 \oplus F_1(x_0)))] \\ \oplus F_2^2(x_2 \oplus F_2(x_1))]. \end{aligned} \tag{8}$$

Note that other branches of the 5-th round are even more complex. In this case we do not have the separability of input blocks x_i, since all of them appear at many places. However, the main idea is to notice a "term" which has a potential to be periodic, as for instance "$x_3 \oplus F_3(x_2)$". The choice of this term is motivated by the fact that input block x_3 in round function RF is not input of any inner function F_i^r.

Let us fix $x_3 = x$ to be a variable, and let $(x_0, x_1, x_2) = (\alpha^{(0)}, \alpha^{(1)}, \alpha^{(2)})$ be a constant vector from $\mathbb{F}_2^{3n}$. Now, if in the place of x_2 we consider two values, say $\alpha_0^{(2)}$ and $\alpha_1^{(2)}$, then in the form of $RF_5^{(4)}(\alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)}, x)$ the value $\alpha_0^{(2)}$ will appear in the term "$x \oplus F_3(\alpha_0^{(2)})$", and it will appear at other places as well. Considering the fixed inputs, we have

$$\begin{aligned} RF_5^{(4)}(\alpha_0, \boldsymbol{x}) = \alpha^{(0)} \oplus F_3^2(\boldsymbol{x} \oplus \boldsymbol{F_3(\alpha_0^{(2)})}) \oplus F_2^3[\boldsymbol{x} \oplus \boldsymbol{F_3(\alpha_0^{(2)})} \oplus F_2^2(\alpha_0^{(2)} \oplus F_2(\alpha^{(1)}))] \\ \oplus F_1^4[\boldsymbol{x} \oplus \boldsymbol{F_3(\alpha_0^{(2)})} \oplus F_1^3(\alpha_0^{(2)} \oplus F_2(\alpha^{(1)}) \oplus F_1^2(\alpha^{(1)} \oplus F_1(\alpha^{(0)})))] \\ \oplus F_2^2(\alpha_0^{(2)} \oplus F_2(\alpha^{(1)}))]. \end{aligned}$$

where $\alpha_0 = (\alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)})$. In general, the function $RF_5^{(4)}(\alpha_0, \boldsymbol{x})$ can be written in terms of some function $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ as

$$g_{00}(x) = RF_5^{(4)}(\alpha_0, \boldsymbol{x}) = G(\boldsymbol{x} \oplus \boldsymbol{F_3(\alpha_0^{(2)})}, \alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)}),$$

which is shortly denoted by $g_{00}(x)$ with respect to variable x, where the index '00' stands for the index of $\alpha_0^{(2)}$ that is appearing inside $F_3(\cdot)$ and outside of it.

Remark 3.4. Note that we here do not consider that any part of $RF_5^{(4)}$ represents a function λ, since it does not make much difference in the case of weak separability (considering also the definition of the round function RF given by (6)).

At this step, we cannot point out some value $s \in \mathbb{F}_2^n$ which will enable us to construct a periodic function by concatenating two functions (and thus applying Proposition 1), since the variable $\alpha_b^{(2)}$ $(b = 0, 1)$ is appearing at other places outside '$x \oplus F_3(\alpha_b^{(2)})$' as well. Since one may consider $s = F_3(\alpha_0^{(2)}) \oplus F_3(\alpha_1^{(2)})$ (which is not known due to the secret key k_3), we have that

$$g_{00}(x \oplus s) = RF_5^{(4)}(\alpha_0, \boldsymbol{x} \oplus \boldsymbol{s}) = G(\boldsymbol{x} \oplus \boldsymbol{F_3(\alpha_1^{(2)})}, \alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)}).$$

In order to construct a periodic function f, it is convenient to query the function $RF_5^{(4)}(x_0, x_1, x_2, x_3)$ with suitable inputs such that the obtained function matches the function $g_{10}(x) = g_{00}(x \oplus s)$.

Considering the function $RF_5^{(4)}(x_0, x_1, x_2, x_3)$ (relation (8)), we notice that the terms $x_3 \oplus F_3(x_2)$, $x_2 \oplus F_2(x_1)$, and $x_1 \oplus F_1(x_0)$ are related such that they have maximally one common variable. Namely, $x_3 \oplus F_3(x_2)$ and $x_2 \oplus F_2(x_1)$ have x_2 in common, $x_2 \oplus F_2(x_1)$ and $x_1 \oplus F_1(x_0)$ have x_1 in common. Since in our functions $g_{00}(x)$ and $g_{10}(x)$ we are fixing the term $x_3 \oplus F_3(x_2) = x \oplus F_3(\alpha_b^{(2)})$ $(b = 0, 1)$, if we then query the encryption oracle with parameters $(\beta, \gamma, \alpha_1^{(2)}, x)$, where $\alpha_1^{(2)}$ is an arbitrary fixed value, we have

$$RF_5^{(4)}(\beta, \gamma, \alpha_1^{(2)}, x) = G(\boldsymbol{x \oplus F_3(\alpha_1^{(2)})}, \beta, \gamma, \alpha_1^{(2)})$$

Due to the structure of the function $RF_5^{(4)}$ in (8) it is not difficult to see that if for fixed vectors $\alpha^{(1)}$ and $\alpha_1^{(2)}$ it holds that

$$\begin{cases} \alpha_1^{(2)} \oplus F_2(\gamma) = \alpha_0^{(2)} \oplus F_2(\alpha^{(1)}) \\ \gamma \oplus F_1(\beta) = \alpha^{(1)} \oplus F_1(\alpha^{(0)}) \end{cases}, \tag{9}$$

then

$$\begin{aligned} (\alpha^{(0)} \oplus \beta) \oplus RF_5^{(4)}(\beta, \gamma, \alpha_1^{(2)}, x) &= G(\boldsymbol{x \oplus F_3(\alpha_1^{(2)})}, \alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)}) \\ &= g_{10}(x). \end{aligned} \tag{10}$$

In (10), $G(\boldsymbol{x \oplus F_3(\alpha_1^{(2)})}, \alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)})$ is denoted by g_{10}, since we have $\alpha_1^{(2)}$ inside $F_3(\cdot)$, and $\alpha_0^{(2)}$ outside of the term $x \oplus F_3(\alpha_1^{(2)})$. Once we find the vectors β and γ for which (9) holds, the Simon's function f is constructed as

$$f(b, x) = \begin{cases} g_{00}(x), & b = 0 \\ g_{10}(x) = g_{00}(x \oplus s), & b = 1. \end{cases}, \tag{11}$$

which has the period $(1, s) = (1, F_3(\alpha_0^{(2)}) \oplus F_3(\alpha_1^{(2)}))$, extractable with Simon's algorithm in time $\mathcal{O}(n)$.

Assuming that F_1 and F_2 are invertible mappings, from (9) we have that

$$\begin{cases} \gamma = F_2^{-1}(\alpha_0^{(2)} \oplus \alpha_1^{(2)} \oplus F_2(\alpha^{(1)})) \\ \beta = F_1^{-1}(\alpha^{(1)} \oplus F_1(\alpha^{(0)}) \oplus \gamma) \end{cases}, \tag{12}$$

which means that system (9) has a unique solution pair (β, γ), where β depends on γ.

Remark 3.5. It is important to note that the assumption on invertibility of F_i implies that the period $(1, s)$ of f given by (11), is unique (Remark 2.1). More importantly, for fixed $\alpha^{(0)}, \alpha^{(1)}, \alpha_b^{(2)}$ $(b = 0, 1)$, the solution pair (β, γ) actually *guarantees* that the construction of f given by (11) has a non-zero period.

From the aspect of periodicity of f, which is imposed by solving the system (9), we note that (9) may have more solution pairs (β, γ), since all these pairs are

involved in the function g_{10} and considered to be a valid pairs as long as it holds that $g_{10}(x) = g_{00}(x \oplus s)$ for all $x \in \mathbb{F}_2^n$. Hence, the solution (12) ensures the periodicity of f, but does not describe all solution pairs (β, γ) that impose the periodicity of f.

Since oracles for F_1 and F_2 may not be available (due to unknown keys k_1, k_2), we are unable to compute γ and β directly. Now, our approach (further analysed in Section 4) is based on combining the Simon's and Grover's algorithm [19] (see also [9], [10]).

Remark 3.6. For the Type-3 GFN with d-branches, the branch $RF_{d+1}^{(d)}(x_0, \ldots, x_{d-1})$ can be written as $RF_{d+1}^{(d)}(x_0, \ldots, x_{d-1}) = x_0 \oplus G(x_{d-1} \oplus F_{d-1}(x_{d-2}), x_0, \ldots, x_{d-2})$, for some function G given in terms of inner functions F_i^r, where $i \in \{1, \ldots, d-1\}$ and $r \in \{1, \ldots, d\}$. Since $RF_{d+1}^{(d)}$ satisfies the weak separability, by choosing suitable inputs x_i (as constants or variables) one can apply the Simon-Grover algorithm presented in Section 4 (similarly as for $RF_5^{(4)}$). In this case, the underlying system of equations is larger and its solvability can be deduced by using the same arguments as for (9). Note that the attack complexity, with respect to the combination of Simon's and Grover's algorithm, would depend on how many constant and variables x_i have been chosen.

4 Combining Simon's and Grover's algorithm

The exhaustive search for pairs (β, γ) that provide the periodicity of f, given by (11), is highly inefficient (for larger n). In this section we combine Simon's and Grover's algorithm in order to find such pairs, where the main focus is the strong scenario which assumes the randomness of inner functions F_i^r (denoted as **CASE I**). We also discuss the weakened scenario (denoted as **CASE II**) in which there may exist many (β, γ) which result in periodicity of f in s which are not necessarily of the form $s = F_3(\alpha_0^{(2)}) \oplus F_3(\alpha_1^{(2)})$. In terms of the Simon-Grover algorithm this scenario has even smaller complexity, since in this case the underlying classifier $\mathcal{B}$ has larger preimage set (good inputs) of the output value 1 (as described in Section 2.1, in the part related to Grover's algorithm).

Recall that as a part of Grover's algorithm, we will have to define the classifier $\mathcal{B}$. For this purpose we will use the function $f : \mathbb{F}_2^{2n+(n+1)} \to \mathbb{F}_2^n$, which is defined as

$$f(\beta, \gamma, b, x) = \begin{cases} g_{00}(x) = RF_5^{(4)}(\alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)}, x), & b = 0 \\ g_{10}(x) = (\alpha^{(0)} \oplus \beta) \oplus RF_5^{(4)}(\beta, \gamma, \alpha_1^{(2)}, x), & b = 1 \end{cases}, \tag{13}$$

where the values of $g_{00}(x)$ we directly take from the truncated encryption oracle $\mathcal{U}_{GFN_3}$, in which the keys involved in F_i^r are known to the oracle. On the other hand, the values of $g_{10}(x)$ also use $\mathcal{U}_{GFN_3}$, with additional inputs γ and β, where vectors $\alpha^{(0)}, \alpha^{(1)}, \alpha_b^{(2)}$ $(b = 0, 1)$ are known and fixed in advance.

Now, let us define the function $\xi : (\mathbb{F}_2^n)^2 \times (\mathbb{F}_2^{n+1})^\ell \to \mathbb{F}_2^{\ell n}$ by

$$\xi : (\beta, \gamma, y_1, \ldots, y_\ell) \to f(\beta, \gamma, y_1) || \ldots || f(\beta, \gamma, y_\ell),$$

where the parameter ℓ ($\approx \mathcal{O}(n)$) implements the parallelized application of Simon's algorithm [19]. Denoting by $\mathcal{U}_\xi$ the quantum oracle defined by

$$\mathcal{U}_\xi : (\beta, \gamma, y_1, \ldots, y_\ell, \mathbf{0}, \ldots, \mathbf{0}) \to |\beta, \gamma, y_1, \ldots, y_\ell, \xi(\beta, \gamma, y_1, \ldots, y_\ell)\rangle,$$

where $\mathbf{0}$ is the all-zero quantum state of length n, then we define:

The quantum algorithm $\mathcal{A}$:

1) Prepare the initial all-zero state $|\mathbf{0}\rangle$ of size $(2n + \ell(n+1) + \ell n)$.

2) Apply the Hadamard transform $H^{\otimes(2n+\ell(n+1))}$ (on the first $\upsilon = 2n + \ell(n+1)$ qubits) and the operator $\mathcal{U}_\xi$ in order to obtain the state

$$2^{-\upsilon/2} \sum_{\substack{\beta,\gamma \in \mathbb{F}_2^n, \\ y_1,\ldots,y_\ell \in \mathbb{F}_2^{n+1}}} |\beta, \gamma\rangle |y_1\rangle \cdots |y_1\rangle |\xi(\beta, \gamma, y_1, \ldots, y_\ell)\rangle.$$

3) Apply the Hadamard transform to states $|y_1\rangle \cdots |y_\ell\rangle$, to get the state

$$|\varphi\rangle = 2^{-\frac{\upsilon+(n+1)\ell}{2}} \sum_{\substack{\beta,\gamma \in \mathbb{F}_2^n, \\ u_1,\ldots,u_\ell \in \mathbb{F}_2^{n+1},\ y_1,\ldots,y_\ell \in \mathbb{F}_2^{n+1}}} |\beta, \gamma\rangle (-1)^{u_1 \cdot y_1} |u_1\rangle \cdots (-1)^{u_\ell \cdot y_\ell} |u_\ell\rangle |\xi(\beta, \gamma, y_1, \ldots, y_\ell)\rangle.$$

At this moment, we are not doing any measurement on $|\varphi\rangle$, since we need to apply Grover's algorithm in addition. Now, we define:

The classifier $\mathcal{B} : (\beta, \gamma, u_1, \ldots, u_\ell) \in \mathbb{F}_2^{2n+\ell(n+1)} \to \mathbb{F}_2$:

1) If dimension of the linear span of $u_1, \ldots, u_\ell \in \mathbb{F}_2^{n+1}$ is not equal to n, i.e., $dim(\langle u_1, \ldots, u_\ell \rangle) \neq n$, then we set $\mathcal{B}(\beta, \gamma, u_1, \ldots, u_\ell) = 0$. Otherwise, use [19, Lemma 2] to compute a candidate period $s' \in \mathbb{F}_2^{n+1}$.

2) Then, check whether $f(\beta, \gamma, y) = f(\beta, \gamma, y \oplus s')$ holds for some amount of (randomly) provided y. If all equalities hold, then the output of $\mathcal{B}$ is 1, otherwise 0.

Remark 4.1. Notice that the definition of the classifier $\mathcal{B}$ actually implements the idea of Simon's algorithm, by finding a test period s' in the first step, and testing it then in the second step.

Thus, we say that a state $(\beta, \gamma, u_1, \ldots, u_\ell)$ is good if and only if $\mathcal{B}(\beta, \gamma, u_1, \ldots, u_\ell) = 1$, that is, when both tests 1) and 2) are satisfied. The classifier $\mathcal{B}$ partitions the state $|\varphi\rangle$ into a good and bad spaces $|\varphi_0\rangle$ and $|\varphi_1\rangle$ respectively, as $|\varphi\rangle = |\varphi_0\rangle + |\varphi_1\rangle$. Here $|\varphi_0\rangle$ and $|\varphi_1\rangle$ denote the projection onto the good and bad subspace respectively. In order to discuss the probability p of obtaining a good state (after measuring the state $|\varphi\rangle$), we distinguish the following cases:

CASE I: Let us assume that F_i^r behave as pseudo-random permutations. Then with very high probability we are expecting that not many pairs (β, γ), different than those given by (12), are implying the periodicity of the function f. This expectation is partly supported in a weak setting presented in Appendix - Table 1, in which we set that $F_i^r(x) = S(x \oplus k_i^r)$, $x \in \mathbb{F}_2^n$, where S is the S-box used in TWINE [26] ($n = 4$) and SMS4 [20] ($n = 8$). Even in these weak settings, the experiments indicate that periodicity of f in $s = F_3(\alpha_0^{(2)}) \oplus F_3(\alpha_1^{(2)})$ is mainly related to existence of two pairs (β, γ). Formally, we have the following result which is quite similar to [19, Lemma 5], where a short proof is provided for self-completeness.

Lemma 7. *Let $\sigma = (\beta', \gamma', u_1, \ldots, u_\ell)$ be an observed state. If a candidate pair (β', γ') is given by $(\beta', \gamma') = (\beta, \gamma)$, where (β, γ) is given by (12), then $\mathcal{B}(\sigma) = 1$ holds with probability at least $\frac{1}{5}$. On contrary, assume that probability of (β, γ) to be a wrong pair, that is when $(\beta', \gamma') \neq (\beta, \gamma)$ is not given by (12), is upper bounded by 2^{-z} with $z > 2n - 4$ (based on the pseudorandomness of F_i^r). If the output of $\mathcal{B}$ is equal to 1, then $(\beta', \gamma') = (\beta, \gamma)$ holds probability $> 1 - \frac{1}{2^{z-2n+4}}$.*

Proof. Using the same arguments as in the first part of the proof of [19, Lemma 5], one obtains that $p_0 = Pr[\mathcal{B}(\sigma) = 1 | \beta' = \beta, \gamma' = \gamma] \geq \frac{1}{5}$. Regarding the second part, let $\sigma = (\beta', \gamma', y_1, \ldots, y_\ell)$ be an input for which it holds that $\mathcal{B}(\sigma) = 1$. Assume that probability of the pair (β', γ') to be a wrong one (that is $(\beta', \gamma') \neq (\beta, \gamma)$) is upper bounded by $q = 2^{-z}$, where $z > 2n - 4$. Here by a wrong pair we mean that at least one of the keys $\beta', \gamma' \in \mathbb{F}_2^n$ is not correct. By law of total probability, it is not difficult to see that $Pr[\mathcal{B}(\sigma) = 1] \leq 2^{-2n} \cdot p_0 + 2^{-n-z+1} + 2^{-z}$, which consequently (using the first part of the proof) gives that

$$Pr[\beta' = \beta, \gamma' = \gamma | \mathcal{B}(\sigma) = 1] \geq 1 - \frac{2^{2n-z+4} - 2^{2(2n-z+4)}}{1 - 2^{2(2n-z+4)}}.$$

Note that the previous two inequalities are obtained using the same computational steps as in [19, Lemma 5]. Assuming that $\nu = z - 2n + 4 > 0$ (i.e., $z > 2n - 4$), we have that $\frac{2^{-\nu} - 2^{-2\nu}}{1 - 2^{-2\nu}} < \frac{1}{2^\nu}$, which consequently (by $-\nu = 2n - z - 4$) gives $Pr[\beta' = \beta, \gamma' = \gamma | \mathcal{B}(\sigma) = 1] \geq 1 - \frac{2^{2n-z-4} - 2^{2(2n-z-4)}}{1 - 2^{2(2n-z-4)}} > 1 - \frac{1}{2^{z-2n+4}}$. □

Consequently, by law of total probability the value p (without loss of generality) is estimated as

$$\begin{aligned} p &= Pr[|\beta', \gamma', u_1, \ldots, u_\ell\rangle \text{ is good}] \\ &\approx Pr[(\beta', \gamma') = (\beta, \gamma)] \cdot Pr[\mathcal{B}(\beta', \gamma', u_1, \ldots, u_\ell) = 1 | (\beta', \gamma') = (\beta, \gamma)] \approx 2^{-2n}, \end{aligned}$$

where (β, γ) are given by (12). Let us consider the second part of definition of $\mathcal{B}$, where we observe ρ randomly taken vectors $y \in \mathbb{F}_2^n$ and $s' = (c, \overline{s}) \in \mathbb{F}_2 \times \mathbb{F}_2^n$ as some candidate period. If $c = 0$ and $y = (0, x)$, then testing the equality $f(\beta, \gamma, y) = f(\beta, \gamma, y \oplus s')$ is equivalent to $g_{00}(x) = g_{00}(x \oplus \overline{s})$, which is highly

unlikely if we assume that F_i^r are random permutations (or if we assume that $RF_r^{(t)}$ behaves as a random permutation for considered number of rounds). Let us consider the case when $g_{00}(x) = g_{10}(x \oplus \overline{s})$, where $g_{00}(x)$ are values coming from $\mathcal{U}_{GFN_3}$ (involving all correct keys) and g_{10} takes the candidate values β, γ as inputs (by design (13)). Then it is reasonable to assume that the probability of satisfying $g_{00}(x) = g_{10}(x \oplus \overline{s})$ (for many random inputs x) with respect to β, γ is equal to $2^{-\varepsilon}$, for sufficiently large ε.

Consequently, the probability of guessing a right pair (β, γ), for which all ρ equalities $g_{00}(x) = g_{10}(x \oplus \overline{s})$ hold, is given by $2^{-\rho \cdot \varepsilon}$, and thus the value z from Lemma 7 is taken to be $z = \rho \cdot \varepsilon$. Thus, we simply observe $\rho = \lceil \frac{2n-4}{\varepsilon} \rceil$ vectors y in order to ensure a high probability of having correct β, γ, if we observe an input $\sigma = (\beta, \gamma, y_1, \ldots, y_\ell)$ for which $\mathcal{B}(\sigma) = 1$. For instance, if $\varepsilon = 2$ (which is very low considering our assumptions), then $\rho = \lceil n - 2 \rceil$, which is always possible.

Hence, $\mathcal{B}$ defines a unitary operator $S_{\mathcal{B}}$ that changes the signs of states as

$$S_{\mathcal{B}} : \ \sigma = |\beta, \gamma, u_1, \ldots, u_\ell\rangle \rightarrow \begin{cases} -|\beta, \gamma, u_1, \ldots, u_\ell\rangle, & \mathcal{B}(\sigma) = 1 \\ |\beta, \gamma, u_1, \ldots, u_\ell\rangle, & \mathcal{B}(\sigma) = 0 \end{cases} .$$

The application of Grover's algorithm is realized by applying the operator $\mathcal{Q} = -\mathcal{A}S_0\mathcal{A}^{-1}S_{\mathcal{B}}$ consecutively t times to the state $|\varphi\rangle = \mathcal{A}|\mathbf{0}\rangle$. Thus, measuring the state $\mathcal{Q}^t\mathcal{A}|\mathbf{0}\rangle$ we obtain a good state with probability p_{good}, which is estimated as follows.

By Theorem 3, we have $\sin^2(\theta) = p = \langle\varphi_1|\varphi_1\rangle$ which implies $\theta \approx \arcsin(\sqrt{p}) \approx 2^{-n}$. Therefore, the number of iterations (by Theorem 3) given by $t = \lceil \frac{\pi}{4\theta} \rceil = \lceil \frac{\pi}{4 \cdot 2^{-n}} \rceil \approx 2^n$ is sufficient to result in angle $\pi/2$ between the resulting state $\mathcal{Q}^t\mathcal{A}|\mathbf{0}\rangle$ and the bad subspace. Consequently, the success probability $p_{good} = 1 - 2^{-n}$ is very close to 1 for somewhat larger n.

CASE II: The experiments considered in weak settings given in Appendix - Section A.1 indicate that it may be the case that for various periods $s \in \mathbb{F}_2^n$ there may exist pairs (β, γ) (used as inputs in g_{10}) which imply the periodicity of f. By testing the periodicity for all vectors $s \in \mathbb{F}_2^n$, for the TWINE S-box on average we find 3 vectors s for which there exist two pairs of (β, γ) such that f (given by (11)) is periodic. In rare cases, for certain key sets taken randomly for 5 rounds, we may have that for almost all $s \in \mathbb{F}_2^4$ (TWINE S-box is of size 4) one finds at least two pairs of (β, γ) giving the periodicity of f.

If the underlying GFN has this property (at observed branch), i.e., that there exist different periods s which are ensured with more pairs (β, γ), then the classifier $\mathcal{B}$ has the initial probability p much larger than 2^{-2n}, and consequently, the final query complexity is significantly less than $2^n \cdot \mathcal{O}(n)$. In this context, it is important to emphasize that a good outcome $(\beta, \gamma, u_1, \ldots, u_\ell)$ is useful as long as its values β, γ which imply the periodicity of f (where clearly the corresponding period can be computed from vectors $u_1, \ldots, u_\ell$). Also, the probability p_{good} of obtaining this outcome is still satisfying, since the space φ_1 is larger. Note that by [7, Theorem

3], one can still run the Grover's algorithm even when the value p is not known in advance, where due to existence of many periods and corresponding pairs (β, γ), our algorithm will not run forever, but it will be significantly faster than $2^n \cdot \mathcal{O}(n)$.

To summarize: Considering the strong scenario (when F_i^r are pseudorandom permutations), the overall procedure of obtaining a pair $(\beta, \gamma) \in \mathbb{F}_2^{2n}$ for $RF_5^{(4)} : \mathbb{F}_2^{4n} \to \mathbb{F}_2^n$ requires $2n + \ell(2n+1) = 2n + 2(2n+1)(n+1+\sqrt{n+1})$ qubits and approximatively $2^n \cdot \mathcal{O}(n)$ quantum queries, where the value $\ell = 2(n+1+\sqrt{n+1})$ is chosen according to [19, Lemma 4] (for vectors $y_i \in \mathbb{F}_2^{n+1}$ in the algorithm $\mathcal{A}$). In addition, a good outcome provides also a set of vectors $y_1, \ldots, y_\ell$ which we can use to find a period $s = F_3(\alpha_0^{(2)}) \oplus F_3(\alpha_1^{(2)})$ with very high probability, as discussed earlier. In the weak scenario (when different periods are possible with many (β, γ) pairs), the query complexity tends to be lower, which clearly depends on the observed GFN.

5 Conclusions

For the classical environment, many works have been devoted to improve the diffusion properties of GFN, since they depend on inner round functions. However, in the quantum environment the cryptanalytic methods are taking different direction, in which the quantum algorithms (such as Simon's, Grover's, and others) are playing an important role (for which in many cases the inner functions are not important that much). The method presented in this work focuses on two elements, namely a suitable construction of a Simon's function and collection of a specific equations in terms of inner functions (with suitably fixed inputs). We show that the solvability of the system of collected equations may imply a construction of a quantum distinguisher for (almost) any number of rounds if the considered system has solutions. Unfortunately, our method (being generic) does not run in polynomial time, but in exchange indicates on which specific inner round functions the security may rely. As an interesting research direction, we leave our method for further investigation in the context of: other (cyclically inequivalent) GFNs used in some well-known block ciphers, GFNs with non-binary group operations, unbalanced GFN, design criterions, combination with other attacks, and so on.

References

[1] Thierry P. Berger, Marine Minier, and Gaël Thomas. "Extended Generalized Feistel Networks Using Matrix Representation". In: *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*. Ed. by Tanja Lange, Kristin E. Lauter, and Petr Lisonek. Vol. 8282. Lecture Notes in Computer Science. Springer, 2013, pp. 289–305. DOI: 10.1007/978-3-662-43414-7_15. URL: https://doi.org/10.1007/978-3-662-43414-7%5C_15.

[2] Andrey Bogdanov and Kyoji Shibutani. "Generalized Feistel networks revisited". In: *Des. Codes Cryptogr.* 66.1-3 (2013), pp. 75–97. DOI: 10.1007/s10623-012-9660-z. URL: https://doi.org/10.1007/s10623-012-9660-z.

[3] Xavier Bonnetain. "Quantum Key-Recovery on Full AEZ". In: *Selected Areas in Cryptography - SAC 2017 - 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers*. Ed. by Carlisle Adams and Jan Camenisch. Vol. 10719. Lecture Notes in Computer Science. Springer, 2017, pp. 394–406. DOI: 10.1007/978-3-319-72565-9_20. URL: https://doi.org/10.1007/978-3-319-72565-9%5C_20.

[4] Xavier Bonnetain, Akinori Hosoyamada, María Naya-Plasencia, Yu Sasaki, and André Schrottenloher. "Quantum Attacks Without Superposition Queries: The Offline Simon's Algorithm". In: *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I*. Ed. by Steven D. Galbraith and Shiho Moriai. Vol. 11921. Lecture Notes in Computer Science. Springer, 2019, pp. 552–583. DOI: 10.1007/978-3-030-34578-5_20. URL: https://doi.org/10.1007/978-3-030-34578-5%5C_20.

[5] Xavier Bonnetain and María Naya-Plasencia. "Hidden Shift Quantum Cryptanalysis and Implications". In: *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part I*. Ed. by Thomas Peyrin and Steven D. Galbraith. Vol. 11272. Lecture Notes in Computer Science. Springer, 2018, pp. 560–592. DOI: 10.1007/978-3-030-03326-2_19. URL: https://doi.org/10.1007/978-3-030-03326-2%5C_19.

[6] Xavier Bonnetain, María Naya-Plasencia, and André Schrottenloher. "On Quantum Slide Attacks". In: *Selected Areas in Cryptography - SAC 2019 - 26th International Conference, Waterloo, ON, Canada, August 12-16, 2019, Revised Selected Papers*. Ed. by Kenneth G. Paterson and Douglas Stebila. Vol. 11959. Lecture Notes in Computer Science. Springer, 2019, pp. 492–519. DOI: 10.1007/978-3-030-38471-5_20. URL: https://doi.org/10.1007/978-3-030-38471-5%5C_20.

[7] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. "Quantum amplitude amplification and estimation". In: *Contemporary Mathematics* 305 (2002), pp. 53–74.

[8] Xiaoyang Dong, Bingyou Dong, and Xiaoyun Wang. "Quantum Attacks on Some Feistel Block Ciphers". In: *IACR Cryptol. ePrint Arch.* (2018), p. 504. URL: https://eprint.iacr.org/2018/504.

[9] Xiaoyang Dong, Zheng Li, and Xiaoyun Wang. "Quantum cryptanalysis on some generalized Feistel schemes". In: *Sci. China Inf. Sci.* 62.2 (2019), 22501:1–22501:12. DOI: 10.1007/s11432-017-9436-7. URL: https://doi.org/10.1007/s11432-017-9436-7.

[10] Xiaoyang Dong and Xiaoyun Wang. “Quantum key-recovery attack on Feistel structures”. In: *Sci. China Inf. Sci.* 61.10 (2018), 102501:1–102501:7. DOI: 10.1007/s11432-017-9468-y. URL: https://doi.org/10.1007/s11432-017-9468-y.

[11] Lov K. Grover. “A Fast Quantum Mechanical Algorithm for Database Search”. In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*. Ed. by Gary L. Miller. ACM, 1996, pp. 212–219. DOI: 10.1145/237814.237866. URL: https://doi.org/10.1145/237814.237866.

[12] Akinori Hosoyamada and Kazumaro Aoki. “On Quantum Related-Key Attacks on Iterated Even-Mansour Ciphers”. In: *Advances in Information and Computer Security - 12th International Workshop on Security, IWSEC 2017, Hiroshima, Japan, August 30 - September 1, 2017, Proceedings*. Ed. by Satoshi Obana and Koji Chida. Vol. 10418. Lecture Notes in Computer Science. Springer, 2017, pp. 3–18. DOI: 10.1007/978-3-319-64200-0_1. URL: https://doi.org/10.1007/978-3-319-64200-0%5C_1.

[13] Akinori Hosoyamada and Yu Sasaki. “Quantum Demiric-Selçuk Meet-in-the-Middle Attacks: Applications to 6-Round Generic Feistel Constructions”. In: *Security and Cryptography for Networks - 11th International Conference, SCN 2018, Amalfi, Italy, September 5-7, 2018, Proceedings*. Ed. by Dario Catalano and Roberto De Prisco. Vol. 11035. Lecture Notes in Computer Science. Springer, 2018, pp. 386–403. DOI: 10.1007/978-3-319-98113-0_21. URL: https://doi.org/10.1007/978-3-319-98113-0%5C_21.

[14] Gembu Ito, Akinori Hosoyamada, Ryutaroh Matsumoto, Yu Sasaki, and Tetsu Iwata. “Quantum Chosen-Ciphertext Attacks Against Feistel Ciphers”. In: *Topics in Cryptology - CT-RSA 2019 - The Cryptographers’ Track at the RSA Conference 2019, San Francisco, CA, USA, March 4-8, 2019, Proceedings*. Ed. by Mitsuru Matsui. Vol. 11405. Lecture Notes in Computer Science. Springer, 2019, pp. 391–411. DOI: 10.1007/978-3-030-12612-4_20. URL: https://doi.org/10.1007/978-3-030-12612-4%5C_20.

[15] Gembu Ito and Tetsu Iwata. “Quantum Distinguishing Attacks against Type-1 Generalized Feistel Ciphers”. In: *IACR Cryptol. ePrint Arch.* (2019), p. 327. URL: https://eprint.iacr.org/2019/327.

[16] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. “Breaking Symmetric Cryptosystems Using Quantum Period Finding”. In: *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9815. Lecture Notes in Computer Science. Springer, 2016, pp. 207–237. DOI: 10.1007/978-3-662-53008-5_8. URL: https://doi.org/10.1007/978-3-662-53008-5%5C_8.

[17] Hidenori Kuwakado and Masakatu Morii. “Quantum distinguisher between the 3-round Feistel cipher and the random permutation”. In: *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings*. IEEE, 2010, pp. 2682–2685. DOI: 10.1109/ISIT.2010.5513654. URL: https://doi.org/10.1109/ISIT.2010.5513654.

[18] Hidenori Kuwakado and Masakatu Morii. “Security on the quantum-type Even-Mansour cipher”. In: *Proceedings of the International Symposium on Information Theory and its Applications, ISITA 2012, Honolulu, HI, USA, October 28-31, 2012*. IEEE, 2012, pp. 312–316. URL: https://ieeexplore.ieee.org/document/6400943/.

[19] Gregor Leander and Alexander May. “Grover Meets Simon - Quantumly Attacking the FX-construction”. In: *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Vol. 10625. Lecture Notes in Computer Science. Springer, 2017, pp. 161–178. DOI: 10.1007/978-3-319-70697-9_6. URL: https://doi.org/10.1007/978-3-319-70697-9%5C_6.

[20] Fen Liu, Wen Ji, Lei Hu, Jintai Ding, Shuwang Lv, Andrei Pyshkin, and Ralf-Philipp Weinmann. “Analysis of the SMS4 Block Cipher”. In: *Information Security and Privacy, 12th Australasian Conference, ACISP 2007, Townsville, Australia, July 2-4, 2007, Proceedings*. Ed. by Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson. Vol. 4586. Lecture Notes in Computer Science. Springer, 2007, pp. 158–170. DOI: 10.1007/978-3-540-73458-1_13. URL: https://doi.org/10.1007/978-3-540-73458-1%5C_13.

[21] Boyu Ni, Gembu Ito, Xiaoyang Dong, and Tetsu Iwata. “Quantum Attacks Against Type-1 Generalized Feistel Ciphers and Applications to CAST-256”. In: *Progress in Cryptology - INDOCRYPT 2019 - 20th International Conference on Cryptology in India, Hyderabad, India, December 15-18, 2019, Proceedings*. Ed. by Feng Hao, Sushmita Ruj, and Sourav Sen Gupta. Vol. 11898. Lecture Notes in Computer Science. Springer, 2019, pp. 433–455. DOI: 10.1007/978-3-030-35423-7_22. URL: https://doi.org/10.1007/978-3-030-35423-7%5C_22.

[22] Kaisa Nyberg. “Generalized Feistel Networks”. In: *Advances in Cryptology - ASIACRYPT ’96, International Conference on the Theory and Applications of Cryptology and Information Security, Kyongju, Korea, November 3-7, 1996, Proceedings*. Ed. by Kwangjo Kim and Tsutomu Matsumoto. Vol. 1163. Lecture Notes in Computer Science. Springer, 1996, pp. 91–104. DOI: 10.1007/BFb0034838. URL: https://doi.org/10.1007/BFb0034838.

[23] Martin Rötteler and Rainer Steinwandt. “A note on quantum related-key attacks”. In: *Inf. Process. Lett.* 115.1 (2015), pp. 40–44. DOI: 10.1016/j.ipl.2014.08.009. URL: https://doi.org/10.1016/j.ipl.2014.08.009.

[24] Thomas Santoli and Christian Schaffner. "Using Simon's algorithm to attack symmetric-key cryptographic primitives". In: *Quantum Inf. Comput.* 17.1&2 (2017), pp. 65–78. DOI: 10.26421/QIC17.1-2-4. URL: https://doi.org/10.26421/QIC17.1-2-4.

[25] Daniel R. Simon. "On the Power of Quantum Computation". In: *SIAM J. Comput.* 26.5 (1997), pp. 1474–1483. DOI: 10.1137/S0097539796298637. URL: https://doi.org/10.1137/S0097539796298637.

[26] Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. "TWINE: A Lightweight Block Cipher for Multiple Platforms". In: *Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers*. Ed. by Lars R. Knudsen and Huapeng Wu. Vol. 7707. Lecture Notes in Computer Science. Springer, 2012, pp. 339–354. DOI: 10.1007/978-3-642-35999-6_22. URL: https://doi.org/10.1007/978-3-642-35999-6%5C_22.

[27] Linhong Xu, Jiansheng Guo, Jingyi Cui, and Mingming Li. "Key-recovery attacks on LED-like block ciphers". In: *Tsinghua Science and Technology* 24.5 (2019), pp. 585–595. DOI: 10.26599/TST.2018.9010130.

[28] Yuliang Zheng, Tsutomu Matsumoto, and Hideki Imai. "On the Construction of Block Ciphers Provably Secure and Not Relying on Any Unproved Hypotheses". In: *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*. Ed. by Gilles Brassard. Vol. 435. Lecture Notes in Computer Science. Springer, 1989, pp. 461–480. DOI: 10.1007/0-387-34805-0_42. URL: https://doi.org/10.1007/0-387-34805-0%5C_42.

A Appendix

A.1 Experimental results related to system (9)

In Table 1 we consider the number of pairs (β, γ) which are implying the periodicity of f exactly in $s = F_3(\alpha_0^{(2)}) \oplus F_3(\alpha_1^{(2)})$, where f is given by (11) as

$$f(b,x) = \begin{cases} g_{00}(x) = RF_5^{(4)}(\alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)}, x), & b = 0, \\ g_{10}(x) = (\alpha^{(0)} \oplus \beta) \oplus RF_5^{(4)}(\beta, \gamma, \alpha_1^{(2)}, x), & b = 1. \end{cases} \tag{14}$$

We take that F_i^r are defined by $F_i^r(x) = S(x \oplus k_i^r)$, $x \in \mathbb{F}_2^n$, with S being the S-box used in TWINE [26] ($n = 4$) and SMS4 [20] ($n = 8$). Since we are considering only 5 rounds, the keys supplied to inner functions F_i^r are taken to be arbitrary, and thus we are considering in total $5 \times 3 = 15$ random keys (effectively it is needed 4×3, since $RF_4^{(1)} = RF_5^{(4)}$). In addition, with respect to these random sets of keys, we are also taking random quadruples $(\alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)}, \alpha_0^{(2)})$.

S-box	Number of random quadruples $(\alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)}, \alpha_0^{(2)})$	Number of key sets for the first 5 rounds	Number of pairs (β, γ) which imply periodicity of f in s
TWINE ($n = 4$)	10	10	2
SMS4 ($n = 8$)	10	10	2

Table 1: The number of pairs (β, γ) that imply a periodic function f given by (14).

We notice that for different random key sets, one may obtain the same pairs of (β, γ), or eventually the first/second values are equal. Unfortunately, we did not find why in almost all cases one obtains exactly 2 different pairs of (β, γ). In rare cases, for the TWINE S-box, for certain instances of keys and quadruples $(\alpha^{(0)}, \alpha^{(1)}, \alpha_0^{(2)}, \alpha_0^{(2)})$ there exist 16 pairs (β, γ) which imply periodicity of f. However, recall that this is a weak setting of the inner functions F_i^r considered only on 5 rounds (if one would relate these to the presence of weak keys).

A Formal Analysis of Boomerang Probabilities

Publication Information

Andreas B. Kidmose and Tyge Tiessen. "A Formal Analysis of Boomerang Probabilities". In: *IACR Transactions on Symmetric Cryptology* 2022.1 (Mar. 2022), pp. 88–109. DOI: 10.46586/tosc.v2022.i1.88-109. URL: https://tosc.iacr.org/index.php/ToSC/article/view/9528

Contribution

- Main contributions in experimental verification, writing, and comparison with existing work.

Remarks

This publication has been slightly edited to fit the format.

A Formal Analysis of Boomerang Probabilities

Andreas B. Kidmose[1] and Tyge Tiessen[1]

Technical University of Denmark, Kgs. Lyngby, Denmark abki,tyti@dtu.dk

Keywords: boomerang attack cryptanalysis independence Serpent

Abstract. In the past 20 years since their conception, boomerang attacks have become an important tool in the cryptanalysis of block ciphers. In the classical estimate of their success probability, assumptions are made about the independence of the underlying differential trails that are not well-founded. We underline the problems inherent in these independence assumptions by using them to prove that for any boomerang there exists a differential trail over the entire cipher with a higher probability than the boomerang.
While cryptanalysts today have a clear understanding that the trails can be dependent, the focus of previous research has mostly gone into using these dependencies to improve attacks but little effort has been put into giving boomerangs and their success probabilities a stronger theoretical underpinning. With this publication, we provide such a formalization.
We provide a framework which allows us to formulate and prove rigorous statements about the probabilities involved in boomerang attacks without relying on independence assumptions of the trails. Among these statements is a proof that two-round boomerangs on SPNs with differentially 4-uniform S-boxes always deviate from the classical probability estimate to the largest degree possible.
We applied the results of this formalization to analyze the validity of some of the first boomerang attacks. We show that the boomerang constructed in the amplified boomerang attack on Serpent by Kelsey, Kohno, and Schneier has probability zero. For the rectangle attack on Serpent by Dunkelman, Biham, and Keller, we demonstrate that a minuscule fraction of only $2^{-43.4}$ of all differential trail combinations used in the original attack have a non-zero probability. In spite of this, the probability of the boomerang is in fact a little higher than the original estimate suggests as the non-zero trails have a vastly higher probability than the classical estimate predicts.

1 Introduction

One of the most important developments in block cipher cryptanalysis was the invention of differential cryptanalysis by Biham and Shamir [4]. Any block cipher

proposed today must be argued secure against differential attacks. Several ways to do this have been tried over the years, mainly focused on bounding the maximal probability of a single differential trail. The idea was that if the maximal probability was p then at least p^{-1} texts would be needed. Several ciphers with guaranteed security against simple differential attacks were proposed and later broken by more sophisticated methods.

One example is the KN-cipher [20], which is a 6-round Feistel cipher that uses x^3 as the nonlinear part of the round function,. The authors had proven this construction secure against ordinary differentials. However, the low degree meant it could be broken by higher-order differentials [14] with very low complexity.

Another example is COCONUT98 [23] which used a decorrelation technique to separate the upper and lower halves of the cipher. The upper and lower halves are weak 4-round Feistel networks and the decorrelation module is an addition and a multiplication with key material in a finite field. The multiplication with a secret value makes it impossible to push a difference through the module.

This design caused Wagner [24] to propose the boomerang attack as a clever way to connect unrelated high-probability differentials for the top and bottom half. The basic idea is to "throw" a pair of plaintexts through the cipher, add a difference to the resulting ciphertexts, and observe how they return (see Figure 1). A second-order differential in the middle of the cipher connects the differentials for the upper and lower halves causing the boomerang to return. The classical analysis of the probability of the boomerang returning (see subsection 2.1) assumes the differentials act independently.

This attack worked exceptionally well on the COCONUT98 cipher since the 4-round Feistel cipher admit differentials with probability of $4^{-4.3}$. The important observation is that for a fixed key the decorrelation module is affine. This means there is a probability-one transition, however, predicting the output difference is impossible without knowledge of the key. Knowing the exact difference is not needed for the attack, only that any second-order derivative is zero with probability one.

Some variants to the basic boomerang attack have been proposed over the years. The idea of the amplified boomerang attack by Kohnu et al. [15] is to turn the boomerang attack into a chosen-plaintext attack instead the original adaptively chosen-plaintext/ciphertext attack. The rectangle attack by Biham et al. [3] builds up on the amplified boomerang by making use of the fact that the differences in the middle need not be fixed but can take any value, as long as the sum is 0. The sandwich attack by Dunkelman et al. [11], [12] proposes a framework where the two differentials are separated in the middle, like two pieces of bread with a thin slice of meat hence the name. The differentials for the upper and lower halves are then connected via ad-hoc methods through the middle round. As a framework it has proven to be a good basis for investigating the dependencies of the differentials involved in a boomerang attack.

The boomerang attack and its variants have proven themselves to be effective on a wide variety of ciphers. Notable examples include the attack on AES by Biryukov and Khovratovich [5], and on KASUMI by Dunkelman et al. [11]. More recently

the retracing boomerang attack was introduced by Dunkelman et al. [10], which improved the best attack on 5-round AES by discarding some data and forcing the boomerang back along the same trajectory.

Related work

The differentials in a boomerang attack were usually assumed to be independent, however there is no a priori justification for that assumption. Several techniques, commonly known as boomerang switches, have been proposed to take advantage of dependencies to boost the probability of the boomerang. The *Feistel switch*, which bypasses a round for free, was already implemented by Wagner in [24] in the attack on Khufu. The *ladder switch* and the *S-box switch* were introduced by Biryukov and Khovratovich [5]. In the ladder switch the attacker chooses the boundary of the two differentials such that it does not necessarily align with the rounds of the cipher. When putting an S-box in a differential where it is inactive, instead of active, it does not add to the probability. The S-box switch is the fact that, if the output difference for the upper differential matches the difference from the lower differential, then the pairs are just swapped and we only pay for the probability in one direction.

While the switches were used to aid the attacker, Murphy [18] pointed out that the differentials might in fact be incompatible. In the middle of the boomerang, where the upper and lower differentials meet, we have a pair of pairs. The upper differential defines the distance between the pairs and the lower defines the difference for the differential transition. It may be the case that there are several pairs that follow the transition for the lower differential but none with the distance dictated by the upper differential. Murphy in particular showed an example for DES, where the required transition over S_2 in round 4 is impossible, and therefore that particular boomerang never comes back.

Cid et al. [8] proposed the boomerang connectivity table (BCT) as a unified approach to calculate the boomerang switching probability in SPN-ciphers when the middle part is one S-box layer. It includes the previously mentioned switches and as well as a new switching property they discovered, and it can be used to show when two differentials are incompatible.

The BCT depends on the inverse S-box, however since the inverse S-box is not used in a Feistel cipher, a different approach is needed here. The FBCT, for Feistel boomerang connectivity table, was introduced by Boukerrou et al. [6] to solve this issue. The authors also show the invalidity of the related-key boomerang attack on LBlock by Liu et al. [17].

The BCT and FBCT work at the S-box level and as such only consider one round; the dependency can, however, span many rounds. Two concurrent papers looked at this problem with different approaches. Wang and Peyrin [25] took a table-based approach, where the DDT and BCT are combined in a table which they call the boomerang difference table (BDT). This table can be used to evaluate the probability of a boomerang switch over 2 rounds. Song et al. [21] instead proposed a way to determine the length of the middle part, where the dependency exists.

The classical way to find a good boomerang distinguisher would be to choose the best differentials for the upper and lower part separately, and then just hope that they are compatible. The problem is that the best differentials might not have a high probability of connecting in the middle, and therefore choosing a lower probability differential might result in a higher probability boomerang. Recently several MILP models have been proposed to search for boomerang distinguishers, e.g., [9] and [13], which will take the switching probability into account for multiple rounds.

Motivation

Since the inception of the boomerang attack, we have come to appreciate some of the difficulties involved in estimating the probabilities of boomerang distinguishers. While there is a general understanding that the naive method of estimating boomerang probabilities as the product of the individual involved trails is incorrect, and while dependencies between the trails have been put to good use in attacks such as the sandwich attack, we still lack a consistent model of describing boomerang probabilities.

With this work, we want to fill this gap by creating a mathematical model that allows us to precisely formulate the probabilities of boomerang attacks. The only assumptions that we want to rely on are those commonly made in differential cryptanalysis.

Our contribution

In this paper, we take a close look at the probability estimates classically made in boomerang attacks and which assumptions are being made. We show that using these assumptions we can prove that for any boomerang there would necessarily exists a differential over the entire cipher with higher probability than the boomerang. While this is clearly not the case, it underlines the need for a better formal underpinning of boomerang success probabilities.

Building up on a notation that extends the notions from differential cryptanalysis to take a quartet of messages into account, we are able to rigorously prove several results regarding the probability of boomerang attacks. Among these are compact expressions of the boomerang probabilities as well as results on the applicability and limitation of the classical estimates of boomerang probabilities. In particular we are able to prove that two-round boomerangs on SPNs with differentially 4-uniform S-boxes never adhere to the classical probability estimate.

We furthermore apply our results to two classic boomerang attacks on the block cipher Serpent. The first one, the amplified boomerang attack on Serpent [15], is shown to be invalid as stated. For the second one, the rectangle attack [2], we show that of all boomerang trails considered in the attack only a fraction of $2^{-43.4}$ have a non-zero probability. For the remaining boomerang trails we demonstrate that their probability is much higher than a classical estimate would suggest, leaving the total combined probability of all boomerang trails close to the original estimate. By

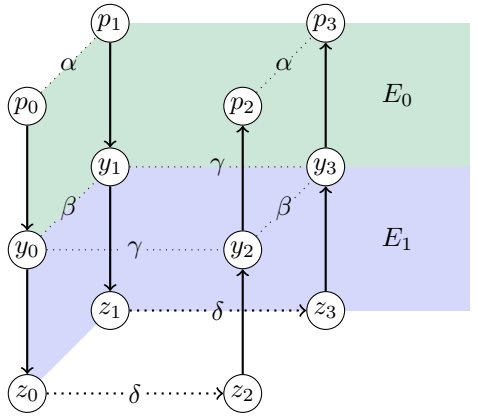


Figure 1: Outline of a basic boomerang attack.

including even more trails in a refined analysis we are able to improve this original estimate by a factor of $2^{4.3}$.

Outline

In Section 2 we introduce the basic boomerang attack. In Section 3 we prove that the independence assumption is an inherently flawed assumption. In Section 4 we introduce d-differences which will be used in Section 5 to prove some statements about boomerang probabilities. In Section 7 we look at two boomerang attacks on Serpent, and finally Section 8 concludes the paper.

2 Boomerang attacks

Before we can start our investigation into formalizing the boomerang attack probabilities, we need a proper exposition of this attack and its variants.

2.1 Basic boomerangs

Let us start by properly introducing boomerang attacks, developed by David Wagner [24] and set the notation which is used throughout this paper.

Let Enc denote a block cipher that maps n-bit plaintexts bijectively to n-bit ciphertexts. For the purpose of the attack, we assume that the cipher can be decomposed into two parts E_0 and E_1 such that

$$\text{Enc} = E_1 \circ E_0 \ . \tag{1}$$

We are now interested in the scenario where we have a good differential $\alpha \xrightarrow{E_0} \beta$ over E_0 that holds with probability p, as well as a good differential $\gamma \xrightarrow{E_1} \delta$ over E_1

that holds with probability q. These two differentials can now be used to construct a distinguisher over the whole cipher as follows.

To construct the distinguisher, we start with a pair of plaintexts x_0 and x_1 with a difference α. When encrypting these two plaintexts, we expect the corresponding intermediate texts $y_0 := E_0(x_0)$ and $y_1 := E_0(x_1)$ to have a difference β with probability p (at least according to the standard assumptions of differential cryptanalysis). For orientation, see also Fig. 1.

With $z_0 := E_1(y_0)$ and $z_1 := E_1(y_1)$ being the respective ciphertexts, we now construct two more ciphertexts $z_2 := z_0 \oplus \delta$ and $z_3 := z_1 \oplus \delta$ by adding the difference δ to each of z_0 and z_1. Then the pairs (z_0, z_2) and (z_1, z_3) both have a difference of δ, the ciphertext difference in the second differential.

Decrypting these two ciphertexts, provides us with two more intermediate texts, $y_2 := E_1^{-1}(z_2)$ and $y_3 := E_1^{-1}(z_3)$, and two more plaintexts, $x_2 := E_0^{-1}(y_2)$ and $x_3 := E_0^{-1}(y_3)$. Assuming independence of the two ciphertext pairs (z_0, z_2) and (z_1, z_3), both of their respective intermediate pairs (y_0, y_2) and (y_1, y_3) will have a differences of γ with probability q^2.[1]

Combining this with the probability that (x_0, x_1) follows the first differential, we have with probability pq^2 that $y_0 \oplus y_1 = \beta$, $y_0 \oplus y_2 = \gamma$, and $y_1 \oplus y_3 = \gamma$. This forces the difference between y_2 and y_3 to be β. Again assuming independence from the other pairs, the pair (y_2, y_3) will follow the first differential with probability p, resulting in a plaintext difference of α between x_2 and x_3.

Taking all of these steps together, we estimate that the probability to see a difference α between x_2 and x_3 is equal to p^2q^2.

Over a random permutation, the probability for x_2 and x_3 to have a difference of α is $(2^n - 1)^{-1}$. We therefore expect such a boomerang distinguisher to be successful as long as p^2q^2 is sufficiently larger than $(2^n - 1)^{-1}$, since we can then use the above technique to distinguish the cipher from a random permutation in a chosen plaintext/chosen ciphertext attack.

In accordance with the above notation, we make the following definition:

Definition 1. We call a tuple of four plaintexts $(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n}$ a *right quartet* with respect to the input difference $\alpha \in \mathbb{F}_2^n$ and the output difference $\delta \in \mathbb{F}_2^n$ and a fixed key if and only if $x_0 \oplus x_1 = \alpha$, $x_2 \oplus x_3 = \alpha$, and $\mathrm{Enc}(x_0) \oplus \mathrm{Enc}(x_2) = \delta$ and $\mathrm{Enc}(x_1) \oplus \mathrm{Enc}(x_3) = \delta$.

Since the only non-deterministic part of the method is our initial choice of x_0, it is straightforward to see that the probability to detect the output difference α between x_2 and x_3 using the boomerang attack is equal to the number of right quartets divided by 2^n.

As described above, classically this probability is estimated to be p^2q^2 by making mentioned independence assumptions. A focal point of this paper is to shed light upon how and why these independence assumptions fail and what the consequences are for the estimate of the boomerang probability.

[1] The probability of a differential is the same both in the forward and backward direction for any bijective function. Note that this is not true for truncated differentials.

Taking more differentials into consideration

When estimating the number of expected right quartets with the classical assumptions as $2^n p^2 q^2$, we clearly are estimating a lower bound as we are only considering two particular differentials. To get a more accurate classical estimate of the boomerang probability we need to consider all possible values for the differences β and γ in the middle of the cipher:

$$\sum_{\beta \in \mathbb{F}_2^n, \gamma \in \mathbb{F}_2^n} \Pr\left(\alpha \xrightarrow{E_0} \beta\right)^2 \Pr\left(\gamma \xrightarrow{E_1} \delta\right)^2. \tag{2}$$

As a matter of fact, there is no reason to restrict the differences $y_0 \oplus y_1$ and $y_2 \oplus y_3$ to be the same, and likewise the differences $y_0 \oplus y_2$ and $y_1 \oplus y_3$ to be the same. Thus an even better classical approximation of the boomerang probability is thus

$$\sum_{\substack{\beta_0, \beta_1, \gamma_0, \gamma_1 \in \mathbb{F}_2^n, \\ \beta_0 \oplus \beta_1 \oplus \gamma_0 \oplus \gamma_1 = 0}} \Pr\left(\alpha \xrightarrow{E_0} \beta_0\right) \Pr\left(\alpha \xrightarrow{E_0} \beta_1\right) \Pr\left(\gamma_0 \xrightarrow{E_1} \delta\right) \Pr\left(\gamma_1 \xrightarrow{E_1} \delta\right). \tag{3}$$

For references of these classical estimates, see for example Wagner [24, Section 6] or Biham et al. [3, Section 4].

Practical restrictions

To find a good approximation of the boomerang probability, one would ideally like to determine the expressions in Eq. (3) or Eq. (2). This is in most cases impossible in practice. As a matter of fact, even when we only work with two differentials for each cipher part, it is usually computationally infeasible to determine the probabilities of just these differentials. What we tend to do in practice then, is to restrict ourselves to the most promising differential trails for the upper and lower parts of the cipher and take their probabilities to determine a good approximation for the boomerang probability.

2.2 Amplified boomerangs

One inherent limitation of the original boomerang attack is that it requires both chosen plaintexts and adaptively-chosen ciphertexts. The attack can be adapted to only requiring chosen plaintexts albeit with a much higher data complexity. This method was developed by Kelsey, Kohno, and Schneier [15]. The complexity of this method also inherently depends on the number of expected right quartets, i.e., the boomerang probability.

The idea is to choose two values x_0, x_2 and generate $x_1 = x_0 \oplus \alpha$ and $x_3 = x_2 \oplus \alpha$. Then like in the standard boomerang attack the differences in the middle will be β for both pairs with probability p^2, that is, $y_0 \oplus y_1 = y_2 \oplus y_3 = \beta$. The distance will be γ with a probability of 2^{-n} since if $y_0 \oplus y_2 = \gamma$ then so is $y_1 \oplus y_3 = \gamma$.

Finally the transition for the lower half is again q^2 and therefore the probability that $z_0 \oplus z_2 = z_1 \oplus z_3 = \delta$ is $p^2q^2 2^{-n}$. By creating a large set of pairs of input texts with difference α, the large number of possible combinations of input pairs allows the success probability to be higher than 2^{-n} then.

3 Independence assumptions in boomerang attacks

3.1 Independence of rounds and trails

The assumptions used in boomerang attacks can generally be put into two categories: those assumptions that stem from the theory of differential cryptanalysis and those assumptions that are specific to the boomerang attack.

The most important assumption from differential cryptanalysis is the assumption that the rounds of a cipher can be treated independently when determining differential probabilities, thereby allowing us to multiply the probabilities of differential round transitions to obtain the probability of a trail. We will briefly discuss this assumption when creating our model for calculating boomerang probabilities, so for now let us leave it by saying that this assumption works sufficiently well in practice.

The additional assumption made in the classical estimate of the boomerang probability though is about the independence of trails and cannot be derived from the standard assumptions of differential cryptanalysis. In the classical estimate of the boomerang probability, we simply multiply the probabilities of the four individual differential transitions. To be able to do this, we implicitly assume that these four differentials can be regarded as independent. We can state this assumption as follows:

Assumption 1. For any two text pairs (x_0, x_1) and (x_2, x_3) that both have a difference α, i.e., $x_0 \oplus x_1 = x_2 \oplus x_3 = \alpha$, the probability that both pairs are mapped to a difference β is the same as the square of the probability of the differential $\alpha \longrightarrow \beta$ for any choice of α and β.

Clearly this assumption does not hold when the text pairs coincide as the second pair then always follows the same differential as the first. In the following, we show that even assuming that this assumption holds closely for the case where the text pairs are distinct leads to contradictions.

3.2 An inherent problem

The idea and purpose of boomerang attacks is to provide an attack in scenarios where we might not find a good differential that covers the entire cipher. There seems to be little reason to use a boomerang attack if we already have a differential of higher probability over the entire cipher. The more surprising it might be that we can prove that there always exists a differential with probability higher than the boomerang probability when we rely on the assumption that we can treat the trail probabilities independently, as we do in the classical estimate.

Using the notation for the boomerang attack established in Section 2.1, we thus would expect that no differentials $\beta \xrightarrow{E_1} \varepsilon$ or $\eta \xrightarrow{E_0} \gamma$ exist for which

$$\Pr\left(\beta \xrightarrow{E_1} \varepsilon\right) \geq q^2 \quad \text{or} \quad \Pr\left(\eta \xrightarrow{E_0} \gamma\right) \geq p^2. \tag{4}$$

If this were not the case, at least one of the differential trails $\alpha \xrightarrow{E_0} \beta \xrightarrow{E_1} \varepsilon$ and $\eta \xrightarrow{E_0} \gamma \xrightarrow{E_1} \delta$ would have a probability higher than p^2q^2 and thus would be better suited as a distinguisher than the boomerang distinguisher.

Theorem 2. *Assume that we have a boomerang as described above of probability p^2q^2 and assume that the assumption of the independence of differentials holds (Assumption 1). Then there exist differentials $\alpha \xrightarrow{\text{Enc}} \varepsilon$ and $\eta \xrightarrow{\text{Enc}} \delta$ over the whole cipher with probabilities at least pq^2 and qp^2, respectively.*

Proof. We only prove here that there exists a differential $\alpha \xrightarrow{\text{Enc}} \varepsilon$ of probability at least pq^2. To show that a differential $\eta \xrightarrow{\text{Enc}} \delta$ of probability at least p^2q exists goes analogously.

We start by showing that if Assumption 1 holds for E_1, then

$$\sum_{\varepsilon} \Pr\left(\beta \xrightarrow{E_1} \varepsilon\right)^2 = \sum_{\delta} \Pr\left(\gamma \xrightarrow{E_1} \delta\right)^2. \tag{5}$$

By choosing some arbitrary fixed γ, we get that

$$\begin{aligned}
&\sum_{\varepsilon} \Pr\left(\beta \xrightarrow{E_1} \varepsilon\right)^2 \\
&= \sum_{\varepsilon} \Pr\Big(E_1(x_0 \oplus \beta) \oplus E_1(x_0) = \varepsilon \text{ and } E_1(x_0 \oplus \gamma \oplus \beta) \oplus E_1(x_0 \oplus \gamma) = \varepsilon\Big) \\
&= \Pr\Big(E_1(x_0 \oplus \beta) \oplus E_1(x_0) = E_1(x_0 \oplus \gamma \oplus \beta) \oplus E_1(x_0 \oplus \gamma)\Big) \\
&= \Pr\Big(E_1(x_0 \oplus \gamma) \oplus E_1(x_0) = E_1(x_0 \oplus \gamma \oplus \beta) \oplus E_1(x_0 \oplus \beta)\Big) \\
&= \sum_{\delta} \Pr\Big(E_1(x_0 \oplus \gamma) \oplus E_1(x_0) = \delta \text{ and } E_1(x_0 \oplus \gamma \oplus \beta) \oplus E_1(x_0 \oplus \beta) = \delta\Big) \\
&= \sum_{\delta} \Pr\left(\gamma \xrightarrow{E_1} \delta\right)^2,
\end{aligned}$$

where we used Assumption 1 in the first and last steps of the derivation. This concludes the proof of Eq. (5).

The probability that both these β differences are mapped to the same value thus gives us an upper bound for the probability that both δ differences are mapped to γ

differences:

$$\sum_{\varepsilon\in\mathbb{F}_2^n} \Pr\left(\beta \xrightarrow{E_1} \varepsilon\right)^2 \geq \Pr\left(\gamma \xrightarrow{E_1} \delta\right)^2 = q^2. \tag{6}$$

Let s now be the maximal value for any of the differentials $\beta \xrightarrow{E_1} \varepsilon$:

$$s := \max_{\varepsilon\in\mathbb{F}_2^n} \Pr\left(\beta \xrightarrow{E_1} \varepsilon\right). \tag{7}$$

Using the fact that the trail probabilities sum to one $\left(\sum_{\varepsilon\in\mathbb{F}_2^n} \Pr\left(\beta \xrightarrow{E_1} \varepsilon\right) = 1\right)$, we then have that

$$q^2 \leq \sum_{\varepsilon\in\mathbb{F}_2^n} \Pr\left(\beta \xrightarrow{E_1} \varepsilon\right)^2 \leq \sum_{\varepsilon\in\mathbb{F}_2^n} s \cdot \Pr\left(\beta \xrightarrow{E_1} \varepsilon\right) = s \cdot \sum_{\varepsilon\in\mathbb{F}_2^n} \Pr\left(\beta \xrightarrow{E_1} \varepsilon\right) = s \tag{8}$$

Thus there exists a differential $\beta \xrightarrow{E_1} \varepsilon$ with probability at least q^2 and thus there exists a differential $\alpha \xrightarrow{\text{Enc}} \varepsilon$ of probability at least pq^2.

Can we conclude from this that there always exist differentials that beat boomerang distinguishers and that we only need to find them? Certainly not (see for example Corollary 11). It much rather demonstrates that we must be very careful when unconditionally assuming independence of differentials as done in the classical estimate of the boomerang probability.

4 Generalized differences and their transitions

Before we look in more detail into the probabilities of boomerangs, let us introduce some notation and a model that allows us to formally discuss boomerang attacks. Parts of the notation that we are using are taken from [22].

In differential cryptanalysis, we are usually not interested in the absolute position of texts in the state space but only in their relative positions, i.e., their relative differences. The relative positions of a tuple of $d+1$ texts are uniquely defined by the differences of the d last texts with respect to the first text. To capture this notion, we define:

Definition 3 (d-difference)**.** For a tuple of $(d+1)$ texts $(m_0, m_1, m_2, \ldots, m_d)$, we describe their relative differences by the d-tuple

$$(m_1 \oplus m_0, m_2 \oplus m_0, \ldots, m_n \oplus m_0). \tag{9}$$

We refer to such a d-tuple as the *d-difference* of the $(d+1)$-tuple of messages. We refer to the first text of the $(d+1)$-tuple of messages as the *anchor* of the d-difference.

Note that d-differences thus describe the translation-invariant equivalence class of $(d+1)$-tuples in the state space. Thus a $(d+1)$-tuple is uniquely identified by its d-difference together with its anchor. For the remainder, we will refer to a $(d+1)$-tuple only as a tuple if the value d is clear from the context.

Extending the notion of differentials to d-differences, we make the following definition:

Definition 4 (Transition with fixed anchor)**.** Let $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)$ be two d-differences over $\mathbb{F}_2^n$. By the transition $\boldsymbol{\alpha} \xrightarrow[x]{f} \boldsymbol{\beta}$, we denote the event that f maps the tuple of messages corresponding to the d-difference $\boldsymbol{\alpha}$ with anchor x to a tuple of messages with d-difference $\boldsymbol{\beta}$. More precisely, we say that $\boldsymbol{\alpha} \xrightarrow[x]{f} \boldsymbol{\beta}$ holds if and only if

$$\begin{aligned} f(x \oplus \alpha_1) \oplus f(x) &= \beta_1, \\ f(x \oplus \alpha_2) \oplus f(x) &= \beta_2, \\ &\ldots \\ f(x \oplus \alpha_d) \oplus f(x) &= \beta_d. \end{aligned}$$

Example 5. Let (m_0, m_1, m_2, m_3) be a plaintext tuple and let (c_0, c_1, c_2, c_3) be the corresponding tuple of ciphertexts. Set $\boldsymbol{\alpha} = (m_1 \oplus m_0, m_2 \oplus m_0, m_3 \oplus m_0)$ and let $\boldsymbol{\beta}$ be some 3-difference. Then $\boldsymbol{\alpha} \xrightarrow[m_0]{f} \boldsymbol{\beta}$ holds if and only if $\boldsymbol{\beta} = (c_0 \oplus c_1, c_0 \oplus c_2, c_0 \oplus c_3)$.

To define the probability of such transitions where the anchor is not fixed, we take the same route as standard differential cryptanalysis and assume that the first text is chosen uniformly at random.

Definition 6 (Probability of transitions)**.** Let f, $\boldsymbol{\alpha}$, and $\boldsymbol{\beta}$ again be as in Definition 4. The probability of the transition $\boldsymbol{\alpha} \xrightarrow{f} \boldsymbol{\beta}$ is then defined as:

$$\Pr\left(\boldsymbol{\alpha} \xrightarrow{f} \boldsymbol{\beta}\right) := \Pr_{\mathbf{X}}\left(\boldsymbol{\alpha} \xrightarrow[\mathbf{X}]{f} \boldsymbol{\beta}\right) \tag{10}$$

where $\mathbf{X}$ is a random variable, distributed uniformly on $\mathbb{F}_2^n$.

For simple differences (1-differences) this definition coincides with the definition of differentials.

Example 7. Let f be the AES S-box, and consider the 3-difference transition $(\alpha_1, \alpha_2, \alpha_3) \xrightarrow{f} (\beta_1, \beta_2, \beta_3)$, where $\alpha_1 = 7$, $\alpha_2 = 25$, and $\alpha_3 = \alpha_1 \oplus \alpha_2$ and $\beta_1 = 166$, $\beta_2 = 183$, and $\beta_3 = \beta_1 \oplus \beta_2$. To calculate the probability we simple count all values of x for which $f(x \oplus \alpha_1) \oplus f(x) = \beta_1$ and $f(x \oplus \alpha_2) \oplus f(x) = \beta_2$ and $f(x \oplus \alpha_3) \oplus f(x) = \beta_3$. There are exactly 4 values (0, 7, 25, and 30) which means that the probability is $2^2 \cdot 2^{-8} = 2^{-6}$. If we instead change $\beta_2 = 1$, then there are no values for x for which it holds and the probability is therefore 0. Note that this also illustrates Lemma 15.

To allow us to lower bound the probability of a transition over several rounds of a cipher, we need the ability to split transition into a collection of trails. To this end, we define what a trail is in this context.

Definition 8 (A d-difference trail)**.** Let $f = f_l \circ \cdots \circ f_2 \circ f_1$ be a composition of n-bit to n-bit functions, let $\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_l \in \mathbb{F}_2^{dn}$ be a sequence of d-differences, and let $x \in \mathbb{F}_2^n$. We refer to the sequence $(\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_l)$ as a trail over f and denote the event that this trail is followed as $\boldsymbol{\alpha}_0 \xrightarrow[x]{f_0} \boldsymbol{\alpha}_1 \xrightarrow[f_0(x)]{f_1} \boldsymbol{\alpha}_2 \cdots \xrightarrow[f_{l-1}\circ\cdots\circ f_0(x)]{f_l} \boldsymbol{\alpha}_l$. That is, we say that the $(d+1)$-tuple corresponding to $\boldsymbol{\alpha}_0$ with the anchor x follows the trail $(\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_l)$ if and only if all the transitions

$$\begin{aligned} \boldsymbol{\alpha}_0 &\xrightarrow[x]{f_0} \boldsymbol{\alpha}_1, \\ \boldsymbol{\alpha}_1 &\xrightarrow[f_0(x)]{f_1} \boldsymbol{\alpha}_2, \\ &\cdots \\ \boldsymbol{\alpha}_{l-1} &\xrightarrow[f_{l-1}\circ\cdots\circ f_0(x)]{f_l} \boldsymbol{\alpha}_l \end{aligned}$$

are adhered to.

To be able to efficiently determine the probability of these trails, we run into the same problem that one encounters when formalizing the probability of differentials trails, namely that the transitions that make up a trail are generally not independent. To solve this issue we make the same assumption that is conventionally made in differential cryptanalysis, namely that we can reasonably well approximate the probability of a trail by considering the individual transitions as independent. This independence is achieved by assuming that the anchor used in each of the transitions is distributed uniformly randomly. This can for example be modelled by saying that a uniformly random constant is added onto the state after every found/round (for the classic example see [16]).

Assumption 2 (Hypothesis of stochastic equivalence)**.** We assume that treating the individual transitions of a trail as independent gives a reasonably good approximation of the real trail probability. Using the previously established notation, we write this as

$$\begin{aligned} \Pr_{\mathbf{X}}\left(\boldsymbol{\alpha}_0 \xrightarrow[\mathbf{X}]{f_0} \boldsymbol{\alpha}_1 \xrightarrow[f_0(\mathbf{X})]{f_1} \boldsymbol{\alpha}_2 \cdots \xrightarrow[f_{l-1}\circ\cdots\circ f_0(\mathbf{X})]{f_l} \boldsymbol{\alpha}_l\right) \\ \approx \Pr\left(\boldsymbol{\alpha}_0 \xrightarrow{f_0} \boldsymbol{\alpha}_1\right) \cdots \Pr\left(\boldsymbol{\alpha}_{l-1} \xrightarrow{f_l} \boldsymbol{\alpha}_l\right) \quad . \end{aligned} \tag{11}$$

Note that for the case of $d = 2$, this corresponds exactly to the standard assumption made in differential cryptanalysis.

We will now state some rules without proof that are useful when working with these transitions (or when working with standard differentials for that matter). For

the purpose of readability, and as we are mostly concerned with quartets of messages here, we fix $d = 3$. Let now $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$ and $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3)$ be two 3-differences and let f be a bijective function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$.

Rule 1. $$\boldsymbol{\alpha} \xrightarrow[x]{f} \boldsymbol{\beta} \iff \boldsymbol{\beta} \xrightarrow[f(x)]{f^{-1}} \boldsymbol{\alpha}$$

Rule 2.

$$(\alpha_1, \alpha_2, \alpha_3) \xrightarrow[x]{f} (\beta_1, \beta_2, \beta_3)$$
$$\iff (\alpha_1, \alpha_1 \oplus \alpha_2, \alpha_1 \oplus \alpha_3) \xrightarrow[x \oplus \alpha_1]{f} (\beta_1, \beta_1 \oplus \beta_2, \beta_1 \oplus \beta_3) \quad (12)$$

In accordance with the common definition of truncated differentials, we also define:

Definition 9 (Transitions of truncated d-differences)**.** A truncated d-difference is an affine subspace in the linear space of d-differences. A truncated d-difference transition is a pair (A, B) of truncated d-differences denoted as $A \xrightarrow{f} B$. The probability of such a truncated d-difference transition $A \xrightarrow{f} B$ is then defined as the probability that an input d-difference chosen uniformly at random from A maps to a d-difference in B:

$$\Pr\left(A \xrightarrow{f} B\right) := |A|^{-1} \sum_{\substack{\boldsymbol{\alpha} \in A \\ \boldsymbol{\beta} \in B}} \Pr\left(\boldsymbol{\alpha} \xrightarrow{f} \boldsymbol{\beta}\right). \quad (13)$$

From this definition and Rule 1 follows immediately the following rule:

Rule 3.

$$|A| \Pr\left(A \xrightarrow{f} B\right) = |B| \Pr\left(B \xrightarrow{f^{-1}} A\right). \quad (14)$$

5 Rigorous statements for boomerang probabilties

As we will see in the following, d-differences and truncated d-difference transitions provide a notation that allows us to formalize statements about the probability of boomerangs in a consistent model.

Let us start by looking at a tuple of four texts (x_0, x_1, x_2, x_3) where the differences of the pairs (x_0, x_1) and (x_2, x_3) are α each. Using 3-differences, we can say that this 4-tuple (x_0, x_1, x_2, x_3) corresponds to a 3-difference $(\alpha, \eta, \alpha \oplus \eta)$ for some $\eta \in \mathbb{F}_2^n$. With this view, we can think of the relationships between the texts in the boomerang attack as 3-differences. This alternative view is depicted in Fig. 2 (see also Fig. 1 for comparison).

Looking at the boomerang attack from this 3-difference perspective, we can state the probability of the return of the boomerang:

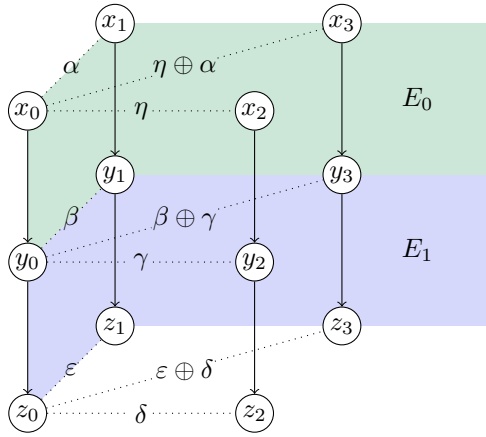


Figure 2: Outline of a simple boomerang attack in the d-difference view. The differences v and w are allowed to take any value here.

Theorem 10. *Let A be the affine subspace of all 3-differences which correspond to an input quartet:*

$$A := \left\{(\alpha, \eta, \alpha \oplus \eta) \in \mathbb{F}_2^{3n} \;\middle|\; \eta \in \mathbb{F}_2^n\right\}. \tag{15}$$

Let B be the set of all 3-differences which correspond to a right ciphertext quartet:

$$B := \left\{(\varepsilon, \delta, \varepsilon \oplus \delta) \in \mathbb{F}_2^{3n} \;\middle|\; \varepsilon \in \mathbb{F}_2^n\right\}. \tag{16}$$

The probability of the return of the boomerang is then equal to the probability of the truncated 3-difference transition $A \xrightarrow{\text{Enc}} B$ multiplied by 2^n:

$$\Pr\left(\textit{Boomerang returns}\right) = 2^n \cdot \Pr\left(A \xrightarrow{\text{Enc}} B\right). \tag{17}$$

Proof. In this proof $\mathbf{X}$ denotes a random variable which is uniformly distributed over $\mathbb{F}_2^n$. The random variable $\mathbf{Y}$ denotes the image of $\mathbf{X}$ under Enc. We know that the boomerang returns if and only if both δ differences are mapped to the same difference in the decryption direction. Using this in the second step and rules 1 and 2 in subsequent steps, we get:

$$\Pr\left(\text{Boomerang returns}\right) \tag{18}$$

$$= \Pr_{\mathbf{X}}\Big(\text{Enc}^{-1}\big(\text{Enc}(\mathbf{X}) \oplus \delta\big) \oplus \text{Enc}^{-1}\big(\text{Enc}(\mathbf{X} \oplus \alpha) \oplus \delta\big) = \alpha\Big) \tag{19}$$

$$= \sum_{\eta \in \mathbb{F}_2^n} \Pr_{\mathbf{X}}\left(\delta \xrightarrow[\text{Enc}(\mathbf{X})]{\text{Enc}^{-1}} \eta \;\text{ and }\; \delta \xrightarrow[\text{Enc}(\mathbf{X}\oplus\alpha)]{\text{Enc}^{-1}} \eta\right) \tag{20}$$

Using the fact that $\mathrm{Enc}(\mathbf{X} \oplus \alpha) = \mathbf{Y} \oplus \varepsilon$ whenever $\varepsilon \xrightarrow[\mathbf{Y}]{\mathrm{Enc}^{-1}} \alpha$, we continue with

$$= \sum_{\substack{\eta \in \mathbb{F}_2^n \\ \varepsilon \in \mathbb{F}_2^n}} \Pr_{\mathbf{Y}} \left(\delta \xrightarrow[\mathbf{Y}]{\mathrm{Enc}^{-1}} \eta \text{ and } \delta \xrightarrow[\mathbf{Y} \oplus \varepsilon]{\mathrm{Enc}^{-1}} \eta \text{ and } \varepsilon \xrightarrow[\mathbf{Y}]{\mathrm{Enc}^{-1}} \alpha \right) \tag{21}$$

Using the third expression, we can now simplify the second by changing the anchor

$$= \sum_{\substack{\eta \in \mathbb{F}_2^n \\ \varepsilon \in \mathbb{F}_2^n}} \Pr_{\mathbf{Y}} \left(\delta \xrightarrow[\mathbf{Y}]{\mathrm{Enc}^{-1}} \eta \text{ and } \delta \oplus \varepsilon \xrightarrow[\mathbf{Y}]{\mathrm{Enc}^{-1}} \eta \oplus \alpha \text{ and } \varepsilon \xrightarrow[\mathbf{Y}]{\mathrm{Enc}^{-1}} \alpha \right) \tag{22}$$

Applying Rule 1 to all three subexpressions

$$= \sum_{\substack{\eta \in \mathbb{F}_2^n \\ \varepsilon \in \mathbb{F}_2^n}} \Pr_{\mathbf{X}} \left(\eta \xrightarrow[\mathbf{X}]{\mathrm{Enc}} \delta \text{ and } \eta \oplus \alpha \xrightarrow[\mathbf{X}]{\mathrm{Enc}} \delta \oplus \varepsilon \text{ and } \alpha \xrightarrow[\mathbf{X}]{\mathrm{Enc}} \varepsilon \right) \tag{23}$$

Collecting everything into 3-differences

$$= \sum_{\substack{\eta \in \mathbb{F}_2^n \\ \varepsilon \in \mathbb{F}_2^n}} \Pr_{\mathbf{X}} \left((\alpha, \eta, \eta \oplus \alpha) \xrightarrow[\mathbf{X}]{\mathrm{Enc}} (\varepsilon, \delta, \delta \oplus \varepsilon) \right) \tag{24}$$

$$= \sum_{\eta \in \mathbb{F}_2^n} \Pr_{\mathbf{X}} \left((\alpha, \eta, \eta \oplus \alpha) \xrightarrow[\mathbf{X}]{\mathrm{Enc}} B \right) \tag{25}$$

$$= 2^n \cdot \Pr \left(A \xrightarrow{\mathrm{Enc}} B \right) \tag{26}$$

which concludes the proof.

The sum (24) contains a single term where $\varepsilon = \delta$ and $\eta = \alpha$ which corresponds to the probability of the ordinary differential $\alpha \xrightarrow{\mathrm{Enc}} \delta$. This yields the following:

Corollary 11. *The probability of the return of the boomerang defined in Theorem 10 is greater than or equal to the probability of the ordinary differential with input difference α and output difference δ.*

We would like to point out that we did not need to use the Assumption 2 for the proof of Theorem 10. Using that assumption now though, we can make statements about the case where we split the encryption function into two parts E_0 and E_1.

Theorem 12. *Let A and B be again as in Eqs.* (15) *and* (16)*. The probability of the boomerang to follow the differentials $\alpha \xrightarrow{E_0} \beta$ and $\gamma \xrightarrow{E_1} \delta$ for the respective text pairs in the upper and lower halves is then equal to*

$$\Pr \left((\beta, \gamma, \beta \oplus \gamma) \xrightarrow{E_0^{-1}} A \right) \cdot \Pr \left((\beta, \gamma, \beta \oplus \gamma) \xrightarrow{E_1} B \right) \quad . \tag{27}$$

Proof. Along similar lines as the proof for Theorem 10.

Comparing with the classical estimate of the boomerang probability, we see that we classically estimate that the 3-difference $(\beta, \gamma, \beta \oplus \gamma)$ is mapped by E_0^{-1} to a 3-difference in A as p^2 and likewise estimate the probability for this 3-difference to be mapped by E_1 into B as q^2.

How well do these approximations hold? The following lemma sheds some light on that:

Lemma 13. *The average of the probability for a 3-difference $(\beta, \gamma, \beta \oplus \gamma)$ to be mapped by a function f to a 3-difference of type $(\alpha, \eta, \alpha \oplus \eta)$ for some $\eta \in \mathbb{F}_2^n$ over all $\gamma \in \mathbb{F}_2^n$ is equal to the square of the probability of the differential $\beta \xrightarrow{f} \alpha$:*

$$2^{-n} \sum_{\gamma,\eta \in \mathbb{F}_2^n} \Pr\left((\beta, \gamma, \beta \oplus \gamma) \xrightarrow{f} (\alpha, \eta, \alpha \oplus \eta)\right) = \left(\Pr\left(\beta \xrightarrow{f} \alpha\right)\right)^2. \tag{28}$$

Proof. Let $\mathbf{X}$ and $\mathbf{Y}$ denote two independent, uniformly distributed random variables on $\mathbb{F}_2^n$. We then have

$$2^{-n} \sum_{\gamma,\eta \in \mathbb{F}_2^n} \Pr\left((\beta, \gamma, \beta \oplus \gamma) \xrightarrow{f} (\alpha, \eta, \alpha \oplus \eta)\right) \tag{29}$$

$$= 2^{-n} \sum_{\gamma,\eta \in \mathbb{F}_2^n} \Pr_{\mathbf{X}}\left(\beta \xrightarrow[\mathbf{X}]{f} \alpha \text{ and } \gamma \xrightarrow[\mathbf{X}]{f} \eta \text{ and } \beta \oplus \gamma \xrightarrow[\mathbf{X}]{f} \alpha \oplus \eta\right) \tag{30}$$

$$= 2^{-n} \sum_{\gamma \in \mathbb{F}_2^n} \Pr_{\mathbf{X}}\left(\beta \xrightarrow[\mathbf{X}]{f} \alpha \text{ and } \beta \xrightarrow[\mathbf{X} \oplus \gamma]{f} \alpha\right) \tag{31}$$

$$= \Pr_{\mathbf{X},\mathbf{Y}}\left(\beta \xrightarrow[\mathbf{X}]{f} \alpha \text{ and } \beta \xrightarrow[\mathbf{Y}]{f} \alpha\right) \tag{32}$$

$$= \left(\Pr\left(\beta \xrightarrow{f} \alpha\right)\right)^2 \tag{33}$$

which concludes the proof.

This lemma could be described as stating that Assumption 1 holds on average over the possible differences in the middle of the boomerang. It should be stressed though that the actual differences used in the middle layer of a boomerang attack are fixed; the actual probability of a boomerang can thus deviate strongly from this average.

A direct consequence of Lemma 13 is the following theorem, which also appears as Proposition 1 in [19].

Theorem 14. *Let a boomerang be given as defined in Theorem 10. The average probability of the return of the boomerang taken over the ciphertext differences δ is equal to the probability that two randomly chosen plaintext pairs with difference α have equal ciphertext differences.*

The probability of two random pairs of texts, both having the same difference α, to be mapped to same differences is closely related to the non-uniformity of the probability distributions of the ciphertext differences resulting from the plaintext difference α. The extreme examples are the APN functions which have the most uniform possible distribution of ciphertext differences with half of the probabilities $\Pr\left(\alpha \xrightarrow{\text{Enc}} w\right)$ equal to 2^{1-n} and the other half equal to zero. It is easy to see that for APN functions the average probability of a boomerang to return is equal to 2^{1-n}. Due to a lack of suitable APN bijections, practical constructions use differentially 4-uniform S-boxes instead. Interestingly, they have two-valued probabilities for propagation of 3-differences as shown in the following lemma, which also follows from the proof of Proposition 4 in [7]:

Lemma 15. *Let $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a differentially 4-uniform bijection. Let $\alpha \xrightarrow{f} \beta$ be a differential of probability p over f and let $A = \left\{(\alpha, \eta, \alpha \oplus \eta) \in \mathbb{F}_2^{3n} \;\middle|\; \eta \in \mathbb{F}_2^n\right\}$. Then the probability that $(\beta, \gamma, \beta \oplus \gamma) \xrightarrow[y]{f^{-1}} A$ is either p or 0 for any $\gamma, y \in \mathbb{F}_2^n$.*

Proof. As f is differentially 4-uniform, we know that there are at most four different values of y such that $\beta \xrightarrow[y]{f^{-1}} \alpha$ holds. Let y_0, y_1, y_2, and y_3 be such four values and let us assume without loss of generality that $y_0 \oplus y_1 = y_2 \oplus y_3 = \beta$. Now for some fixed y, let us look at the probability that $(\beta, \gamma, \beta \oplus \gamma) \xrightarrow[y]{f^{-1}} A$ holds. This is then equivalent to both $\beta \xrightarrow[y]{f^{-1}} \alpha$ and $\beta \xrightarrow[y \oplus \gamma]{f^{-1}} \alpha$ holding simultaneously. Now let us suppose that γ is one of the values 0, β, $y_0 \oplus y_2$, or $y_0 \oplus y_2 \oplus \beta$. Then clearly both differentials hold if and only if y is one of y_0, y_1, y_2, and y_3. On the converse if γ is not equal to any of the above values, then both differentials can never hold simultaneously. Thus the probability that both hold is either 0 or p.

In words, the probability that two randomly chosen data pairs of the same difference follow the same differential strongly depends on the difference between the pairs. We now consider SPNs, that is, a cipher where the round function consists of a non-linear substitution layer comprised of S-boxes applied in parallel, a linear permutation layer, and a key addition. Lemma 15 then allows us to make the following statement:

Theorem 16. *Let* Enc *be a 2-round SPN using 4-uniform S-boxes in the substitution layer. Let* $\text{Enc} = E_1 \circ E_0$ *be such that E_0 and E_1 correspond to one round of the cipher each. Let further $\alpha \xrightarrow{E_0} \beta$ and $\gamma \xrightarrow{E_1} \delta$ be two differentials with probabilities p and q respectively. Then the probability of the boomerang constructed from these differentials is either 0 or pq.*

Proof. We determine the probability using Eq. (27). Let us evaluate the first factor. As E_0 consist only of an S-box layer and an affine layer, it is straightforward to see that $(\beta, \gamma, \beta \oplus \gamma) \xrightarrow[y]{f^{-1}} A$ holds if and only if the respective 3-difference transitions over

all single S-boxes hold. But as the S-boxes are 4-uniform, we know from Lemma 15 that the probabilities of the transitions over the S-boxes are either 0 or correspond to the probability of a single differential over the S-box. This property is thus lifted to the complete round and we thus know that the first factor in Eq. (27) thus evaluates either to 0 or p. The argument for the second factor is analogous.

We should note that it is enough for the boomerang to have a probability-zero transition in one S-box to set the total probability to zero. As a consequence any randomly chosen simple boomerang over such a cipher has a high probability of having a probability of zero. We give examples of this behavior on some well-known boomerang attacks in Section 7.

6 Comparison to the boomerang connectivity table (BCT)

The boomerang connectivity table (BCT) [8] is a tool that has been applied successfully in recent years. We would thus like to give a quick comparison between the techniques used in this paper and the BCT.

A BCT allows us to determine the probability that two trails connect successfully over an S-box when this S-box corresponds to the middle layer of a sandwich attack. The BCT for a given input difference α and output difference δ is defined as the number of right quartets over one n-bit S-box S:

$$\mathrm{BCT}(\alpha,\delta) := \left|\left\{x \in \mathbb{F}_2^n \;\middle|\; S^{-1}(S(x)\oplus\delta)\oplus S^{-1}(S(x\oplus\alpha)\oplus\delta) = \alpha\right\}\right|$$

Using Theorem 10, we can formulate this with our framework.

Theorem 17. *Given an S-box S, an input difference α and output difference δ, we can state the* BCT entry as

$$\mathrm{BCT}(\alpha,\delta) = \sum_{\substack{\eta\in\mathbb{F}_2^n\\ \varepsilon\in\mathbb{F}_2^n}} (\alpha,\eta,\alpha\oplus\eta) \xrightarrow{S} (\varepsilon,\delta,\varepsilon\oplus\delta)$$

We would like to point out that in contrast to Lemma 15, we need to sum over both η and ε. Thus the probabilities in the BCT are not limited by the highest entries in the difference distribution table (DDT). Thus ensuring that the trails do not connect directly in the middle but leave a middle round as in the sandwich attack can have a positive effect on the probability of the attack.

7 Boomerang attacks on Serpent

In this section, we will take a closer look at two of the most well-known applications of boomerang attacks, namely the amplified boomerang attack [15] and the rectangle attack [3] on the block cipher Serpent.

As an SPN with differentially 4-uniform S-boxes, Serpent is a prime test candidate for Theorem 16. Before we go into more detail into the specific boomerang attacks, let us first make one observation. Any simple boomerang constructed from a differential trail for the top part and another trail for the bottom part contains a 2-round boomerang at its core. This makes it necessary to take Theorem 16 into consideration also when the boomerang covers more than two rounds.

7.1 Short overview on Serpent

The block cipher Serpent [1] was an AES candidate and ranked second in the final evaluation. Serpent is constructed as a 32-round substitution-permutation network (SPN) and has been designed to offer a very effective bit-sliced implementation. One round of Serpent consists of an S-box layer in which the same four-bit S-box is applied to all four-bit nibbles of the state followed by an affine layer and a round key addition. We provide a full description of Serpent in Section A to make the paper self-contained.

7.2 Amplified boomerang attack [15]

In the amplified boomerang attack on Serpent [15], a simple boomerang distinguisher on seven rounds (rounds one to seven) with a single differential trail for the top and a single differential trail for the bottom is used. The top part (E_0 in the paper) consists of rounds one to four while the bottom part (E_1 in the paper) consists of rounds five to seven. When taking a closer look at the probability of the inner two-round boomerang over rounds four and five, it is straightforward to see that there are several probability-zero transitions over the S-boxes.

Our approach here is to look at the first S-box transition of the lower differential, $B'_5 \to Y'_5$ (see Fig. 3). We then look at the pairs that follow the differences for each S-box and check if two pairs have the difference from the upper B'_5. If there are no pairs with this requirement then the transition for that S-box is impossible and therefore the boomerang has probability 0. A similar analysis can be done using the BCT by checking the entry for upper B'_5 to lower Y'_5. In this case the lower B'_5 is disregarded as the boomerang is turned into a sandwich.

As an example consider S-box 24 in round 5 for which the attack requires two pairs such that $S(x) \oplus S(x \oplus c) = 4$ with a distance of 1. The only values for x are $\{4, 7, 8, b\}$, but since none of these have a distance of 1, the transition for this S-box is impossible. Any one of these is sufficient to give the inner and thus the complete boomerang a probability of zero. We can also see this from the BCT (Table 2) as BCT(1,4) = 0. This renders the attack invalid.

7.3 Rectangle attack [3]

Let us now have a look at a more refined boomerang attack on Serpent, namely the rectangle attack [3]. In this attack, a boomerang distinguisher is used on rounds one

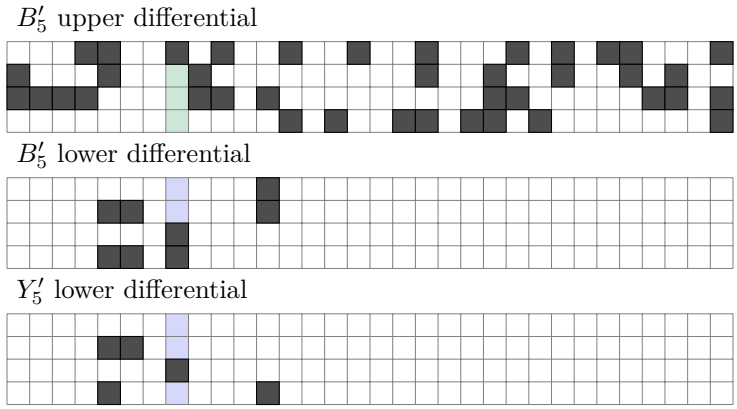


Figure 3: Differences involved in our analysis of the amplified boomerang attack on Serpent [15]. Black bits are active, and the green and blue column marks S-box 24 in the upper and lower differential respectively.

to eight and these rounds are split into two parts of four rounds each. Instead of using only one differential trail for each part, a set of different trails for both parts are used that share the same input or output difference (see Eq. (2) for the classical estimate). For the upper part $2^{13.4}$ trails are used. While the number of trails for the lower part is unspecified in the paper, the probability estimate from these is given. By considering the best $2^{40.0}$ trails of the trail type used in the paper, we get a classical estimate that is slightly higher than the one stated in the paper and we thus assume that these trails are a superset of the trails used in the original paper.[2] In total this leaves us with $2^{53.3}$ combinations of trails for the lower and upper part, giving us a classical estimate of $2^{-119.3}$ for the boomerang probability (this is slightly higher than the original estimate of $2^{-120.6}$ in [3]).

For all of these trail combinations, we calculated the accurate transition probabilities of the two-round inner boomerangs (rounds four and five). For the vast majority of trail combinations, the inner boomerang has a probability of zero, such that only 972 of the $2^{53.3}$ trail combinations are left with a non-zero probability. While this would leave the attack completely invalid if we stuck with the classical estimate (an estimate of $2^{-164.3}$ to be precise), we can apply the results of Theorem 16 positively on these remaining 972 trail combinations: as their inner boomerang has a non-zero probability, it has a much higher probability than the classical estimate would suggest. Interestingly combining the accurate probability for the inner boomerang with the classical estimates for the outer rounds of the boomerang, leaves us not only with a probability estimate that is close to the classical one, it even gives us a slightly higher one, namely $2^{-118.6}$.

[2] For this estimate we consider all trails from round five to eight of the type specified in the Appendix of [3] which activate at most 12 S-boxes in round five.

Unfortunately when we try to correctly determine the probability of a boomerang that exceeds two rounds, we are faced with the problem of the exponential growth in the number of trails that need to be considered. This is particularly true for the rectangle attack on Serpent where we were not able to apply our methodology to more than the inner two rounds. However, Lemma 13 gives us reason to assume that it is justified to apply the classical estimate for the outer rounds, in particular when the diffusion of the cipher is prohibitively large for more accurate methods. As we are in that case considering a large number of distances between the text pairs, it seems acceptable to assume that they reasonably closely estimate the average probability. And this average (as in Eq. (28)) corresponds exactly to the classical estimate.

To improve the quality of the boomerang, we also evaluated the effect of considering more trails for the lower part. We found that when we allowed all trails of the type used in the original attack that activate at most 15 S-boxes in round five, the probability of the boomerang distinguisher improved to $2^{-116.3}$ (as opposed to $2^{-119.0}$ for the classical estimate with the same number of trails). This improves the estimate of the boomerang probability in comparison to the original estimate by a factor of $2^{4.3}$.

8 Summary and conclusion

In this paper, we took a close look at boomerang attacks and the classical estimate of their probability. We explicitly stated the assumption underlying the classical estimates of boomerang probabilities and showed that an inherent contradiction arises when we take this assumption for granted. Using the notion of d-differences and their transitions, a generalization of differential cryptanalysis, we were able to express the probability of boomerang distinguishers precisely in a model that only relies on the independence of rounds instead of the independence of differentials. We then used this formalization to prove a number of results.

One of the most important results is that we could rigorously prove that two-round boomerangs on SPN ciphers with differentially 4-uniform S-boxes—including ciphers such as AES, Serpent, or PRESENT—deviate strongly from their expected classical probability estimate. This results in a very high likelihood for boomerangs that only make use of two differentials to have probability zero, even when covering more than two rounds. On the other hand, this also allows cleverly constructed boomerangs to beat the classical estimate by a large margin.

As an application of these results, we took a closer look at two classical applications of boomerangs on Serpent. For the first attack [15], we found that the boomerang, as constructed from two differentials, has in fact probability zero. For the second attack [2], we found that although only a fraction of $2^{-43.4}$ of all possible considered trail combinations had a non-zero probability, the total estimate of the boomerang was hardly altered as the remaining trails showed a much higher probability than the classical estimate would suggest. As a matter of fact by including some more trails, we were able to improve the classical estimate of the boomerang by a factor of $2^{4.3}$.

How come that the probability estimate for the rectangle attack was so little

influenced by the vast amount of probability-zero trail combinations? The explanation lies in the very large number of trails with comparable probabilities used in the attack. This allowed the classical estimate which only holds on average (as proven in Lemma 13) to describe the probability quite accurately.

From all this we conclude the following: probability estimates in boomerang attacks must be handled with the care. Simply combining two differentials to construct a boomerang can easily lead to the boomerang probability being zero, rendering the attack invalid. Detailed arguments and computer validations of the probability, where possible, should be a minimal requirement for all future boomerang attacks.

Acknowledgments

The authors would like to thank the anonymous reviewers and in particular Kaisa Nyberg for excellent feedback and comments that considerably improved the quality of the paper.

References

[1] Eli Biham, Ross J. Anderson, and Lars R. Knudsen. "Serpent: A New Block Cipher Proposal". In: *Fast Software Encryption, 5th International Workshop, FSE '98, Paris, France, March 23-25, 1998, Proceedings*. Ed. by Serge Vaudenay. Vol. 1372. Lecture Notes in Computer Science. Springer, 1998, pp. 222–238. DOI: 10.1007/3-540-69710-1_15. URL: https://doi.org/10.1007/3-540-69710-1%5C_15.

[2] Eli Biham, Orr Dunkelman, and Nathan Keller. "Related-Key Boomerang and Rectangle Attacks". In: *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*. Ed. by Ronald Cramer. Vol. 3494. Lecture Notes in Computer Science. Springer, 2005, pp. 507–525. DOI: 10.1007/11426639_30. URL: https://doi.org/10.1007/11426639%5C_30.

[3] Eli Biham, Orr Dunkelman, and Nathan Keller. "The Rectangle Attack - Rectangling the Serpent". In: *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*. Ed. by Birgit Pfitzmann. Vol. 2045. Lecture Notes in Computer Science. Springer, 2001, pp. 340–357. DOI: 10.1007/3-540-44987-6_21. URL: https://doi.org/10.1007/3-540-44987-6%5C_21.

[4] Eli Biham and Adi Shamir. "Differential Cryptanalysis of DES-like Cryptosystems". In: *J. Cryptol.* 4.1 (1991), pp. 3–72. DOI: 10.1007/BF00630563. URL: https://doi.org/10.1007/BF00630563.

[5] Alex Biryukov and Dmitry Khovratovich. "Related-Key Cryptanalysis of the Full AES-192 and AES-256". In: *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*. Ed. by Mitsuru Matsui. Vol. 5912. Lecture Notes in Computer Science. Springer, 2009, pp. 1–18. DOI: 10.1007/978-3-642-10366-7_1. URL: https://doi.org/10.1007/978-3-642-10366-7%5C_1.

[6] Hamid Boukerrou, Paul Huynh, Virginie Lallemand, Bimal Mandal, and Marine Minier. "On the Feistel Counterpart of the Boomerang Connectivity Table Introduction and Analysis of the FBCT". In: *IACR Trans. Symmetric Cryptol.* 2020.1 (2020), pp. 331–362. DOI: 10.13154/tosc.v2020.i1.331-362. URL: https://doi.org/10.13154/tosc.v2020.i1.331-362.

[7] Christina Boura and Anne Canteaut. "On the Boomerang Uniformity of Cryptographic Sboxes". In: *IACR Trans. Symmetric Cryptol.* 2018.3 (2018), pp. 290–310. DOI: 10.13154/tosc.v2018.i3.290-310. URL: https://doi.org/10.13154/tosc.v2018.i3.290-310.

[8] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. "Boomerang Connectivity Table: A New Cryptanalysis Tool". In: *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10821. Lecture Notes in Computer Science. Springer, 2018, pp. 683–714. DOI: 10.1007/978-3-319-78375-8_22. URL: https://doi.org/10.1007/978-3-319-78375-8%5C_22.

[9] Stéphanie Delaune, Patrick Derbez, and Mathieu Vavrille. "Catching the Fastest Boomerangs Application to SKINNY". In: *IACR Trans. Symmetric Cryptol.* 2020.4 (2020), pp. 104–129. DOI: 10.46586/tosc.v2020.i4.104-129. URL: https://doi.org/10.46586/tosc.v2020.i4.104-129.

[10] Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. "The Retracing Boomerang Attack". In: *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I*. Ed. by Anne Canteaut and Yuval Ishai. Vol. 12105. Lecture Notes in Computer Science. Springer, 2020, pp. 280–309. DOI: 10.1007/978-3-030-45721-1_11. URL: https://doi.org/10.1007/978-3-030-45721-1%5C_11.

[11] Orr Dunkelman, Nathan Keller, and Adi Shamir. "A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony". In: *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*. Ed. by Tal Rabin. Vol. 6223. Lecture Notes in Computer Science. Springer, 2010, pp. 393–410. DOI: 10.1007/978-3-642-14623-7_21. URL: https://doi.org/10.1007/978-3-642-14623-7%5C_21.

[12] Orr Dunkelman, Nathan Keller, and Adi Shamir. “A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony”. In: *J. Cryptol.* 27.4 (2014), pp. 824–849. DOI: 10.1007/s00145-013-9154-9. URL: https://doi.org/10.1007/s00145-013-9154-9.

[13] Hosein Hadipour, Nasour Bagheri, and Ling Song. “Improved Rectangle Attacks on SKINNY and CRAFT”. In: *IACR Trans. Symmetric Cryptol.* 2021.2 (2021), pp. 140–198. DOI: 10.46586/tosc.v2021.i2.140-198. URL: https://doi.org/10.46586/tosc.v2021.i2.140-198.

[14] Thomas Jakobsen and Lars R. Knudsen. “The Interpolation Attack on Block Ciphers”. In: *Fast Software Encryption, 4th International Workshop, FSE ’97, Haifa, Israel, January 20-22, 1997, Proceedings*. Ed. by Eli Biham. Vol. 1267. Lecture Notes in Computer Science. Springer, 1997, pp. 28–40. DOI: 10.1007/BFb0052332. URL: https://doi.org/10.1007/BFb0052332.

[15] John Kelsey, Tadayoshi Kohno, and Bruce Schneier. “Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent”. In: *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings*. Ed. by Bruce Schneier. Vol. 1978. Lecture Notes in Computer Science. Springer, 2000, pp. 75–93. DOI: 10.1007/3-540-44706-7_6. URL: https://doi.org/10.1007/3-540-44706-7%5C_6.

[16] Xuejia Lai, James L. Massey, and Sean Murphy. “Markov Ciphers and Differential Cryptanalysis”. In: *Advances in Cryptology - EUROCRYPT ’91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*. Ed. by Donald W. Davies. Vol. 547. Lecture Notes in Computer Science. Springer, 1991, pp. 17–38. DOI: 10.1007/3-540-46416-6_2. URL: https://doi.org/10.1007/3-540-46416-6%5C_2.

[17] Shusheng Liu, Zheng Gong, and Libin Wang. “Improved Related-Key Differential Attacks on Reduced-Round LBlock”. In: *Information and Communications Security - 14th International Conference, ICICS 2012, Hong Kong, China, October 29-31, 2012. Proceedings*. Ed. by Tat Wing Chim and Tsz Hon Yuen. Vol. 7618. Lecture Notes in Computer Science. Springer, 2012, pp. 58–69. DOI: 10.1007/978-3-642-34129-8_6. URL: https://doi.org/10.1007/978-3-642-34129-8%5C_6.

[18] Sean Murphy. “The Return of the Cryptographic Boomerang”. In: *IEEE Transactions on Information Theory* 57.4 (2011), pp. 2517–2521. DOI: 10.1109/TIT.2011.2111091. URL: https://doi.org/10.1109/TIT.2011.2111091.

[19] Kaisa Nyberg. “The Extended Autocorrelation and Boomerang Tables and Links Between Nonlinearity Properties of Vectorial Boolean Functions”. In: *IACR Cryptol. ePrint Arch.* (2019), p. 1381. URL: https://eprint.iacr.org/2019/1381.

[20] Kaisa Nyberg and Lars R. Knudsen. “Provable Security Against a Differential Attack”. In: *J. Cryptol.* 8.1 (1995), pp. 27–37. DOI: 10.1007/BF00204800. URL: https://doi.org/10.1007/BF00204800.

[21] Ling Song, Xianrui Qin, and Lei Hu. "Boomerang Connectivity Table Revisited. Application to SKINNY and AES". In: *IACR Trans. Symmetric Cryptol.* 2019.1 (2019), pp. 118–141. DOI: 10.13154/tosc.v2019.i1.118-141. URL: https://doi.org/10.13154/tosc.v2019.i1.118-141.

[22] Tyge Tiessen. "Polytopic Cryptanalysis". In: *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*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Vol. 9665. Lecture Notes in Computer Science. Springer, 2016, pp. 214–239. DOI: 10.1007/978-3-662-49890-3_9. URL: https://doi.org/10.1007/978-3-662-49890-3%5C_9.

[23] Serge Vaudenay. "Provable Security for Block Ciphers by Decorrelation". In: *STACS 98, 15th Annual Symposium on Theoretical Aspects of Computer Science, Paris, France, February 25-27, 1998, Proceedings*. Ed. by Michel Morvan, Christoph Meinel, and Daniel Krob. Vol. 1373. Lecture Notes in Computer Science. Springer, 1998, pp. 249–275. DOI: 10.1007/BFb0028566. URL: https://doi.org/10.1007/BFb0028566.

[24] David A. Wagner. "The Boomerang Attack". In: *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*. Ed. by Lars R. Knudsen. Vol. 1636. Lecture Notes in Computer Science. Springer, 1999, pp. 156–170. DOI: 10.1007/3-540-48519-8_12. URL: https://doi.org/10.1007/3-540-48519-8%5C_12.

[25] Haoyang Wang and Thomas Peyrin. "Boomerang Switch in Multiple Rounds. Application to AES Variants and Deoxys". In: *IACR Trans. Symmetric Cryptol.* 2019.1 (2019), pp. 142–169. DOI: 10.13154/tosc.v2019.i1.142-169. URL: https://doi.org/10.13154/tosc.v2019.i1.142-169.

A Description of Serpent

Here we give a description of the parts of Serpent needed to follow the analysis in Section 7. Serpent has a state of 128 bits split into 4 words of 32 bits to allow for an efficient bit-sliced implementation. The convention for Serpent is to use B_i as the state before round i, so B_0 is the plaintext and B_{32} is the ciphertext. Encryption then proceeds as follows:

$$\begin{aligned} Y_i &= S_i(B_i \oplus K_i) & \\ B_{i+1} &= L(Y_i) & i = 0, \ldots, 30 \\ B_{i+1} &= Y_i \oplus K_{i+1} & i = 31 \end{aligned}$$

Here L is the linear layer which in the bit sliced version is described as follows. If we call the state words for X_0, X_1, X_2, X_3 after the key addition and the S-box layer then the linear layer can be described as:

$$
\begin{aligned}
X_0, X_1, X_2, X_3 &= S_i(B_i \oplus K_i) \\
X_0 &= X_0 <<< 13 \\
X_2 &= X_2 <<< 3 \\
X_1 &= X_1 \oplus X_0 \oplus X_2 \\
X_3 &= X_3 \oplus X_2 \oplus (X_0 << 3) \\
X_1 &= X_1 <<< 1 \\
X_3 &= X_3 <<< 7 \\
X_0 &= X_0 \oplus X_1 \oplus X_3 \\
X_2 &= X_2 \oplus X_3 \oplus (X_1 << 7) \\
X_0 &= X_0 <<< 5 \\
X_2 &= X_2 <<< 22 \\
B_{i+1} &= X_0, X_1, X_2, X_3
\end{aligned}
$$

Here $<<<$ is a left rotation and $<<$ is a left shift.

Serpent uses 8 different S-boxes such that round i uses S-box $i \mod 8$. The S-boxes are applied to 1 bit from each word and the same S-box is used for all bits. The bit from X_0 is the least significant and X_3 is the most significant bit. This allows them to be applied in a bit-sliced fashion. The S-boxes are provided in Table 1 and Table 2 gives the BCT of S5 which is used in out analysis in Section 7.

The keyschedule is not relevant for the analysis so we will leave out the description.

Table 1: S-boxes used in Serpent

S0:	3	8	15	1	10	6	5	11	14	13	4	2	7	0	9	12
S1:	15	12	2	7	9	0	5	10	1	11	14	8	6	13	3	4
S2:	8	6	7	9	3	12	10	15	13	1	14	4	0	11	5	2
S3:	0	15	11	8	12	9	6	3	13	1	2	4	10	7	5	14
S4:	1	15	8	3	12	0	11	6	2	5	4	10	9	14	7	13
S5:	15	5	2	11	4	10	9	12	0	3	14	8	13	6	7	1
S6:	7	2	12	5	8	4	6	11	14	9	1	15	13	3	10	0
S7:	1	13	15	0	14	8	2	11	7	4	12	10	9	3	5	6

Table 2: BCT of S5

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	00	00	02	00	02	04	00	00	06	02	02	00	00	02	04
2	16	00	00	00	00	04	06	02	00	00	02	10	00	04	04	00
3	16	02	00	02	06	00	00	02	04	00	00	00	06	02	00	00
4	16	00	00	00	00	02	08	02	00	04	00	04	00	06	00	06
5	16	06	02	00	00	02	06	00	00	04	00	00	00	00	04	08
6	16	02	02	02	00	00	04	02	00	02	00	04	00	04	02	00
7	16	02	00	02	06	02	00	00	04	00	00	00	06	00	00	02
8	16	00	00	02	04	00	02	00	04	02	00	00	04	02	02	02
9	16	00	02	00	00	04	00	02	02	00	02	06	02	00	04	00
a	16	02	02	02	02	00	00	00	02	00	02	02	00	02	00	00
b	16	04	00	00	00	06	00	02	00	00	00	04	00	02	08	06
c	16	00	02	02	04	02	00	02	04	02	02	00	04	00	00	00
d	16	02	02	02	00	00	00	02	02	02	02	00	02	00	00	00
e	16	04	02	00	02	00	00	00	02	02	02	00	00	02	04	04
f	16	00	02	00	08	00	02	00	08	00	02	00	08	00	02	00

Pitfalls in data-masking techniques: Re-identification attacks

Publication Information

Samir Hodžić, Andreas Brasen Kidmose, and Lars Ramkilde Knudsen. "Pitfalls in data-masking techniques: Re-identification attacks". In: *Submitted to AFRICACRYPT*. 2022

Contribution

- Main contributions in developing research questions, assisting in development of experiments, and revision.

Remarks

This publication has been slightly edited to fit the format.

Pitfalls in data-masking techniques: Re-identification attacks

S. Hodžić[1], A. B. Kidmose[1], and L. R. Knudsen[1]

Technical University of Denmark, DTU Compute, Denmark, saho,abki,lrkn@dtu.dk

Abstract. In order to protect a data set which contains sensitive information, one employs various masking techniques which preserve the look/feel of the data. The question is how much information leaks if we consider only the formats of masked data. In this work, the previous question is considered in the context of the so-called re-identifications attacks, that is the identification of a customer's record in a given masked data set. In our approach we consider an industrial data set whose attribute lists are given separately. We provide two algorithms by which we construct different new custom data sets such that we assign different attribute formats to each customer. Our analysis shows that large portion of customers can be uniquely identified even in the worst possible realistic case, if more attributes are provided in the data set. Additionally, assuming the independency between certain attributes (due to their nature), the provided algorithms for merging can be used to estimate closely the real distributions of frequencies (sizes of anonymity sets) across more attributes in the data set, where the correctness of lower and upper bounds holds with high probability.

Keywords: Privacy Format preserving encryption Data masking Re-identification attacks

1 Introduction

The leakage of sensitive data is a huge problem today. As an example, the analysis of global data breaches by InfoWatch Analytical Center [3] shows that data leaks in H1 2018 increased by 12% compared to H1 2017. As further reported in [3],

> "The data leaks compromised 2.39 billion personal and payment data records, including social security numbers, bank card details, and other critical information. In H1 2018, there were 21 mega leaks, each resulting in the loss of over 10 million records. Inside companies, employees were responsible for 53.5% of the leaks, while executives and other privileged users caused over 2% of the cases"[1].

[1]For further details we refer to [3]

Another interesting example of data leakage, related to General Data Protection Regulation (GDPR), has been presented in [9]. More precisely, in an experiment which targeted 55 companies, the authors showed that from 15 companies they could obtain full access to data of certain employees of those companies. The approach was based on impersonating the employed individual by using some publicly available information, e.g., from social media and alike. This shows that the data leakage was possible due to the fact that policies and practices of GDPR data requests could be manipulated via social engineering techniques.

The previously mentioned data breaches clearly shows a need for protecting data, both in the case where it is stored on-site or if it is transmitted to some third party for research purposes. In any case, protection techniques depend on the type of the data and for which purpose it will be used. If we consider the later scenario in which the data is used for analytical/research purposes, the protection usually relies on so-called data-masking methods. The main goal in the masking process of the data is to preserve the look and/or feel of it [7].

With respect to the degree of exposure of the data and the amount of control maintained, there exist various data-masking techniques [7], [8], such as substitution, shuffling, null'ing out, masking out (see [1], [5], [6], [7], [8], [13], [14] for additional masking techniques). In what follows we briefly describe the so-called *Format Preserving Encryption* (FPE) data-masking technique, which is one of the widely used masking methods (see for instance [4], [10], [15]).

The FPE method is a masking approach in which symmetric-key encryption schemes are used in order to protect the data. This means that the owner of a secret key will be able to access to original data, which is not the case for masking techniques which are not invertible (for instance, those in which the real data characters are replaced by X, etc). Additional property of this method is that it preserves the format of the data, and thus makes the data look somewhat realistic in context of the format only. The FPE methods work such that it replaces the characters of the data with some other random-looking ones (letters or numbers) in an invertible way. In general, various FPE methods are utilized by many companies and for that purpose the National Institute of Standards and Technology (NIST) has published the Special Publication 800-38G [11] which defines two standard modes for FPE called FF1 and FF3, both of which are based on the well-known tweakable Feistel cipher construction with AES as the inner round function.

1.1 Motivation

Imagine we have a database encrypted using a format preserving encryption scheme. We might assume that the data is safe, after all, for a good encryption scheme, we expect that the ciphertext does not reveal any information about the plaintext. The problem here is that the format, and length in particular, is preserved. We can now imagine a scenario in which the attacker has gained access to the encrypted database and wants to figure which entry corresponds to a specified customer, for which the attacker knows some attributes, e.g., name and city. Intuitively, if only one customer

has the format the fits the target, the customer has been identified. This leads to the so-called re-identification attacks, which for example presents a serious threat to regulations for the protection of health related information [12].

The main goal of this work is to show statistically how much information is leaking from a given masked data set. The data set is encoded such that only the format is available to the attacker. More details related to the masking method applied to the data set considered in our analysis is given in Section 2.

Methodology: We are using a data set from a Danish company[2] (call it a data set $\mathcal{D}$) which contains the information of 80045 customers. In this data set we have several attributes given separately and organized in anonymity sets, that is, only the formats and how often they appear are given. In order to measure the leakage of the information in terms of probabilities, we are using this data set and construct several new custom data sets where we assign one format from every attribute to every customer. The method for assigning the attributes is performed by two algorithms (given in Section 2.1) in such a way that our constructed new custom data sets range from the sub-optimally lowest to the best probabilities for guessing a customer's record in the data set. Essentially we are recreating a range of data sets that could have produced $\mathcal{D}$.

The main conclusion of the work is that our experimental analysis indicates that any masked realistic data set (which uses any masking method that preserves the format) actually leaks a lot of information about an individual (in context of the re-identification attacks), if the data provides several attributes to every individual. In other words, we show that with high probability one can always guess the record of a given individual in a given data set (regardless of the utilized masking method, if the formats are preserved; which applies to FPE methods as well). On the other hand, our analysis shows that even if one has a data set whose attribute lists are given separately (such as $\mathcal{D}$), then the close estimate of real data (given with respect to more attributes) is possible with high probability.

Remark 1.1. We note that our analysis is demonstrated using only the set $\mathcal{D}$. We highly emphasize that the same analysis can be applied to any data set. The only difference which could make our approach less efficient is when a given data set contains anonymity sets which are less distinguishable even when more attributes are provided for each anonymity set.

There are several further motivational points for conducting the statistical analysis on the customized data sets (that we construct using the set $\mathcal{D}$). In the first place, it is quite reasonable to assume that an adversary may possess a masked data set which contains several attributes assigned to every individual. For instance, an example related to guessing certain secret target information one can find in [12, Section 1.2]. This indicates that even a simple statistical analysis may easily leak secret information. Another important point is related to the following question: How does the worst or the best possible data set look with respect to the given frequencies, if the attribute lists of customers are given separately?

[2]We are thankful to Martin Staal Boesgaard [2] for providing the masked data set.

The algorithms described in Section 2.1 provide acceptable answers to the previous question, and more importantly, we are also able to obtain sub-optimal lower bounds of probabilities for guessing a customer's record in the data set with respect to the re-identification attacks.

The rest of this paper is organized as follows. In Section 2, we fully describe the settings related to (custom) data sets (since our experiments are performed on these). In Section 2.1 we provide two algorithms for merging different attributes which will be assigned to every individual from the data set $\mathcal{D}$. In Section 3 we provide the statistical analysis of the constructed data sets. The concluding remarks are given in Section 4.

2 Constructing custom data sets

We start by describing the data set $\mathcal{D}$. For this data set, we have the data of 80045 customers for the following attributes: Full Name, E-mail, Address and City. These attributes are given separately and organized in anonymity sets with corresponding frequencies. As described in [2, Section II], the encoding scheme $\mathcal{E}$ which has been used to mask the data set $\mathcal{D}$ is given such that it encodes certain disjoint sets of characters (letters) {a,..., z, A,..., Z}, (digits) {0,..., 9} and (special characters) {æ, ø, å, ä, ö, Æ, Ø, Å, Ä, Ö}, to *a*, *d* and *x* respectively. Additionally, it is assumed that the special symbols *"@"* and *"."* are preserved by $\mathcal{E}$.

Clearly, $\mathcal{E}$ is not an invertible encoding scheme which preserves the format. For instance (recalling the examples from [2]), the full name of the customer "Sasha Adams" in $\mathcal{D}$ is encoded as *"aaaaa aaaaa"*, and the e-mail *"john.doe123@ example.com"* is encoded to *"aaaa.aaaddd@aaaaaaa.aaa"*.

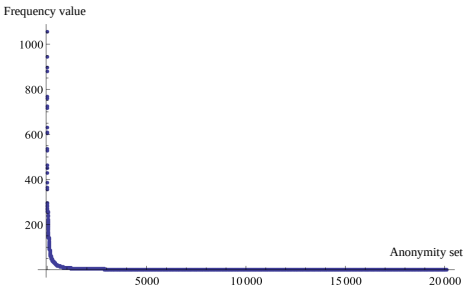
In order to get a proper feel of how the data set $\mathcal{D}$ looks like, we provide Figs. 2a to 2d

The number of anonymity sets for each attribute is given by Table 1. For instance, if we consider the Full Name attribute, from $\mathcal{D}$ we have that the largest anonymity set has the frequency 1054, and this number is approximately visible on Fig. 2a (on y-axis). Also, on Fig. 2a we see that most of the anonymity sets have very low frequency which are below 300 (in fact, only 21 anonymity sets have frequencies above 300). Similarly, on Figs. 2b to 2d one can read approximately the frequencies for other attributes in $\mathcal{D}$.

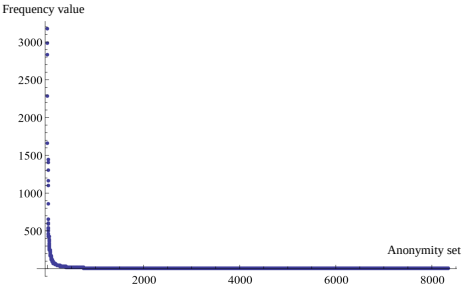
In order to proceed further with our analysis, we note that we do not have available information for certain attributes for all customers. In Table 1, for the E-mail attribute there is only masked data available for 55382 customers (with 7244 different formats) out of 80048 (which is the size of $\mathcal{D}$).

Next, we explain the main steps of our analysis. Our goal is to construct custom data sets using $\mathcal{D}$. The new custom data sets will be created such that we assign more attributes to every customer. For instance, if we consider a customer with the name format *"aaaaaaaa aaaaa"* (in $\mathcal{D}$ the frequency of such formats is 111), then we assign to it the Address format *"aaaaaaaa d"* (the frequency of such formats is

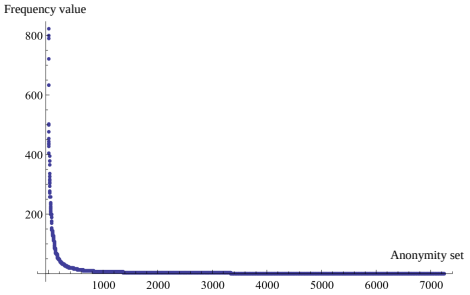
Figure 1: Frequency distributions in for the different attributes in $\mathcal{D}$



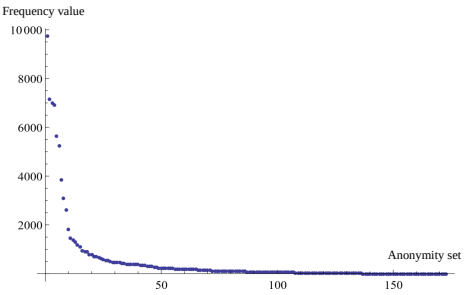
(a) The distribution for the "Full name" attribute



(b) The distribution for the "Address" attribute



(c) The distribution for the "E-mail" attribute



(d) The distribution for the "City" attribute

1310). In other words, we then assume that the customer *"aaaaaaaa aaaaa"* has the address *"aaaaaaaa d"*. Similarly, one further continues by adding other attributes to the same customer.

However, merging attributes for all customers is not directly possible due to the fact that attribute lists are not equal (Table 1), and thus one has to preprocess the attribute lists so that they are of the same length. More precisely, our approach is described as follows.

In our analysis (given in Section 3) we will be merging 2, 3 and 4 attributes per each customer, giving rise to different data sets with respect to the number of attributes, and then analyse those data sets statistically. In order to do this, we have to have the list of frequencies (i.e., sizes of anonymity sets) which are equal.

For instance, considering the Full name and Address attributes, one can notice that there are many anonymity sets (in the Address list) which are of size 1. These customers can be uniquely identified even without considering any other attribute in $\mathcal{D}$, and thus they are of no interest (as pointed our in [2]). Therefore, if we have to reduce the size of certain attribute list, then we will always reduce the larger list by dropping out the formats (which correspond to some customers) whose anonymity

Table 1: The total number of anonymity sets and the amount of available data for each attribute in $\mathcal{D}$.

Attributes	Full name	Address	City	E-mail
The number of anonymity sets	20085	8344	173	7244
The number of filled information	80045	80048	83475	55382

sets are the smallest.

As an example, if we consider two lists of frequencies $\{15, 8, 7, 5, 3, 2, 1, 1, 1\}$ and $\{20, 7, 4, 1, 1, 1, 1\}$ (with total sum of frequencies being 43 and 35 respectively), then we reduce the first one to $\{15, 8, 7, 5\}$ by dropping the customers from the smallest anonymity sets.

The methodology for merging different attributes (after making the length of frequency lists equal as previously described), is based on the algorithms **Alg-Min** and **Alg-Max** (given in Section 2.1), whose applications are described as follows:

i) By **Alg-Min** we are merging attributes such that the number of anonymity sets with respect to merged attributes is minimized. In general, if we assign more attributes to every customer, then the distinguishability between the customers is rising (as noted in [2]). The purpose of **Alg-Min** is to provide a sub-optimal minimization of the distinguishability between the customers. This process has the following two implications. First, by **Alg-Min** one can deduce sub-optimal lower bounds for probabilities of guessing a record for certain customer in a given data set. Secondly, it shows how bad (in terms of guessing) the data set can be given with respect to its frequencies.

ii) On the other hand, in order to make the spectrum of different custom data sets, we use **Alg-Max** to improve the distinguishability between the customers as follows. For instance, if we have a list where every customer has 4 different assigned attributes, and where the merging of these attributes has been done by **Alg-Min**, then **Alg-Max** is going to fix (for instance) the first 3 attributes and the last 4-th attribute list will be randomly merged to the previous 3 attributes (an example is given in Section 2.1). In this way, we have that the 4-th attribute is randomly assigned to the first 3 attributes (more details in Section 2.1).

Remark 2.1. In [2, Section IV] the independency between merged attributes has been pre-assumed in the discussion related to combined attributes, and as it will be visible in our experiments in Section 3, the random merging of the attributes provides the highest level of distinguishability between the customers (which justifies the probability analysis shown in Tables IV and V given in [2]).

Since the algorithms **Alg-Min** and **Alg-Max** are essential for our analysis of custom data sets in Section 3, we provide their detailed description in the next subsection.

2.1 Algorithms for merging attributes

In the first part we illustrate the main ideas behind **Alg-Min**, then we provide a formal description. Later on, we briefly describe **Alg-Max**, which is essentially related to randomized merging between formats of different attribute lists.

The algorithm Alg-Min:

We start with the following example which illustrates **Alg-Min**, whose purpose is to assign formats from different attribute lists to every customer such that the guessing probability of a customer's record is minimized as much as possible.

Let us consider the following two lists of frequencies which correspond to two different attributes $\mathbb{A}_1$ and $\mathbb{A}_2$, namely

$$L_1 = \{(F_1, 15), (F_2, 7), (F_3, 5), (F_4, 3), (F_5, 2)\},$$
$$L_2 = \{(G_1, 11), (G_2, 9), (G_3, 8), (G_4, 2), (G_5, 1), (G_6, 1)\}.$$

Remark 2.2. Note that the lists (corresponding to certain attribute) whose elements are anonymity sets with their frequencies, we will call attribute lists.

Here by $F_1, \ldots, F_5$ and $G_1, \ldots, G_6$ we denote the formats of the corresponding anonymity sets. For instance, in L_1 (corresponding to attribute $\mathbb{A}_1$) we have 15 customers with the format F_1, 7 customers with the format F_2, and so on (and similarly goes for L_2). Note that the sum of all frequencies in both lists is 32.

In order to merge lists L_1 and L_2 such that we decrease the guessing probabilities as much as possible, we consider the following important properties of distinguishability, and later we illustrate all details on lists L_i:

i) Clearly, in the worst possible case if we assume that an observed anonymity set has the same formats across all attributes (in some data set), then guessing the record of some specific customer is very hard, since we can not distinguish the customers in this anonymity set. For instance, consider an anonymity set with 400 customers with the same format across all attributes as $(Z_1, M_1, R_1, E_1, 400)$, where Z_1, M_1, R_1, E_1 are formats that correspond to some different attributes. Clearly, if we have a known customer whose formats across 4 attributes are given exactly as (Z_1, M_1, R_1, E_1), then one can not guess easily which record corresponds to this customer (probability of guessing would be $1/400$).

ii) It is natural to expect that most customers will differ in at least one attribute, and thus we expect this type of anonymity set to be unlikely. Therefore, if we have to merge one specific format (say, from attribute list $\mathbb{A}_1$) with some other format (from the different attribute list $\mathbb{A}_2$), then *we want to avoid as much as*

possible the merging between one fixed format from some big anonymity set in $\mathbb{A}_1$ *with many different formats from* $\mathbb{A}_2$.

iii) The previous conclusion implies the fact that *the overall distinguishability between the customers, with respect to more attributes, is decreased if merged formats from different attributes do not result in many anonymity sets.*

iv) In order to reduce the number of anonymity sets (considered across different attributes), one has to do merging of anonymity sets (with respect to more attributes) such that the new obtained anonymity sets are as big as possible (as described below in **Steps 1** and **2**). The necessity of this approximation is also visible from the fact that keeping large anonymity sets with the same format across many attributes actually *keeps the greater portion of the whole set of customers to be hardly identifiable.*

v) This simply means that *if we want to merge a format of some large anonymity set of one attribute with some format of another attribute, then we are looking to merge the formats between the largest possible anonymity sets in the attribute lists.*

Essentially, the previous observations imply the sub-optimality of the algorithm **Alg-Min** which is demonstrated in details on lists L_1 and L_2. Hence, the formats between the these lists, with respect to previously discussed observations, are merged as follows:

Step 1: Since the largest two anonymity sets in L_i are $(F_1, 15)$ and $(G_1, 11)$, then clearly at maximum 11 customers may have the two assigned attributes (F_1, G_1). The remaining $15 - 11 = 4$, which are corresponding to the format F_1, will be added to the list L_1. The reason for adding it to L_1 is actually the application of the observation (v) and the fact that we always want to merge the anonymity sets from L_1 and L_2 which are of the closest possible size. This means that we will not be merging $(F_1, 4)$ with $(G_2, 9)$, since in L_1 we already have the bigger anonymity set $(F_2, 7)$. Therefore, the element $(F_1, 4)$ is being placed in the list L_1, and at this moment the lists L_1 and L_2 are given as

$$L_1 = \{(F_2, 7), (F_3, 5), (F_1, 4), (F_4, 3), (F_5, 2)\},$$
$$L_2 = \{(G_2, 9), (G_3, 8), (G_4, 2), (G_5, 1), (G_6, 1)\}.$$

Remark 2.3. Notice that after adding an element to the list, we will always impose the ordering on L_i such that from left to right we always put the anonymity sets with higher frequencies (thus we have $(F_1, 4)$ in the middle of the list). In this way, we can keep track of the anonymity sets from L_i which are as closest as possible by size. After the merging from one anonymity set is completely done, we will remove that set from the corresponding list (for instance, we removed $(F_1, 15)$ and $(G_1, 11)$).

Remark 2.4. If the added element, say (F_j, t), has the same frequency t as some other element (F_r, t) in L_1, then the ordering of these two elements in L_1 will not matter

for the statistical analysis of the new data set $\tilde{\mathcal{D}}$, i.e., it does not matter whether L_1 is given as $L_1 = \{\ldots, (F_j, t), (F_r, t), \ldots\}$, or $L_1 = \{\ldots, (F_r, t), (F_j, t), \ldots\}$ (the same rule applies to L_2).

The new data set $\tilde{\mathcal{D}}$ that we are creating, with respect to two attributes $\mathbb{A}_1$ and $\mathbb{A}_2$, in this step has the first anonymity set given as

$$\tilde{\mathcal{D}} = \{(F_1, G_1, 11)\}.$$

Step 2: Repeating the previous procedure further, we merge $(F_2, 7)$ with $(G_2, 9)$, and there we have the anonymity set $(F_2, G_2, 7)$, and the remaining element $(G_2, 2)$ will be added to the list L_2. Thus L_i and $\tilde{\mathcal{D}}$ in this step are given as

$$\begin{aligned} L_1 &= \{(F_3, 5), (F_1, 4), (F_4, 3), (F_5, 2)\}, \\ L_2 &= \{(G_3, 8), (G_4, 2), (G_2, 2), (G_5, 1), (G_6, 1)\}, \\ \tilde{\mathcal{D}} &= \{(F_1, G_1, 11), (F_2, G_2, 7)\}. \end{aligned}$$

If we continue further, we will drain all elements from lists L_i since the total sum of their frequencies is 32, i.e., no formats from these lists will stay unmerged. Thus, the complete merging is providing the set $\tilde{\mathcal{D}}$ given as

$$\tilde{\mathcal{D}} = \begin{pmatrix} (F_1, G_1, 11) \\ (F_2, G_2, 7) \\ (F_3, G_3, 5) \\ (F_1, G_3, 3) \\ (F_4, G_4, 2) \\ (F_5, G_2, 2) \\ (F_4, G_5, 1) \\ (F_1, G_6, 1) \end{pmatrix}. \tag{1}$$

Regarding the observations (i)-(v), we see that in the list L_2 the probability of guessing a customer's record in the anonymity set $(G_1, 11)$ is $1/11$. However, in $\tilde{\mathcal{D}}$ we have preserved this probability, and moreover, consider the list L_2 the frequency 11 is constituting the largest portion of the whole set of customers. This means that approximately $34,4\%$ customers of L_2 (here $11/32 = 0.3437$) have preserved its probability of guessing after merging the attributes $\mathbb{A}_1$ and $\mathbb{A}_2$. In this way, we have completely applied the merging rule from the observation (v).

Now we provide the formal description of the algorithm **Alg-Min**, and then we discuss its properties.

Algorithm 1. [**Alg-Min**]
Input: Data sets $L_1 = \{(F_1, i_1), \ldots, (F_p, i_p)\}$ and $L_2 = \{(G_1, j_1), \ldots, (G_q, j_q)\}$ with frequency set $S = \sum_{k=1}^{p} i_k = \sum_{k=1}^{q} j_k$ (L_i correspond to attributes $\mathbb{A}_i$, $i = 1, 2$).
Output: Data set $\tilde{\mathcal{D}} = \{(F_{i_a}, G_{j_b}, t_{ab}) : \sum t_{ab} = S, i_a \in \{i_1, \ldots, i_p\}, j_b \in \{j_1, \ldots, j_q\}\}$ with merged attributes $\mathbb{A}_i$ $(i = 1, 2)$ with sub-optimally minimal number of anonymity sets in $\tilde{\mathcal{D}}$.

Set $\tilde{\mathcal{D}} = \emptyset$. While $\#L_1 \neq \emptyset$, compare the first elements of L_i (from left to right denote them by (F_k, i_k), (G_u, j_u), regardless of values of k, u[3]):

► If $i_k = j_u$, then add (F_k, G_u, i_k) to $\tilde{\mathcal{D}}$.

► If $i_k < j_u$, then add (F_k, G_u, i_k) to $\tilde{\mathcal{D}}$, add $(G_u, j_u - i_k)$ to L_2 and sort L_2 (from left to right with respect to frequencies).

► If $i_k > j_u$, then add (F_k, G_u, j_u) to $\tilde{\mathcal{D}}$, add $(F_k, i_k - j_u)$ to L_1 and sort L_1.

Drop (F_k, i_k) and (G_u, j_u) from L_i.

The algorithm **Alg-Min** has the following properties:

P1: It (sub-optimally) minimizes the number of anonymity sets in $\tilde{\mathcal{D}}$ along with maximization of their sizes. This means that the guessing probabilities of customer's records in $\tilde{\mathcal{D}}$ are minimized as much as possible, with respect to merged attributes. Note that this property reduces the overall distinguishability between the customers in $\tilde{\mathcal{D}}$ (the observation (iii)).

P2: The order of merging formats L_1 with L_2 does not affect the output of **Alg-Min**, regardless of the fact that the frequencies of L_1 and L_2 are different. This means that the same set $\tilde{\mathcal{D}}$ will be obtained at the end, whether we set that $L_2 = \{(F_1, i_1), \ldots, (F_p, i_p)\}$ and $L_1 = \{(G_1, j_1), \ldots, (G_q, j_q)\}$. This property applies even for merging multiple attributes.

To demonstrate further application of **Alg-Min** with multiple attributes, we provide the following example.

Example 1. Let us consider the attribute lists $L_3 = \{(H_1, 11), (H_2, 10), (H_3, 8), (H_4, 2), (H_5, 1)\}$ and $L_4 = \{(T_1, 17), (T_2, 8), (T_3, 5), (T_4, 2)\}$, which correspond to some attributes $\mathbb{A}_3$ and $\mathbb{A}_4$ respectively, where H_i and T_i are some formats in these attributes. Using **Alg-Min** we merge these two lists with the data set $\tilde{\mathcal{D}}$ given by (1). We obtain the following new data set:

$$\overline{\mathcal{D}} = \begin{pmatrix} (F_1, G_1, H_1, T_1, 11) \\ (F_2, G_2, H_2, T_2, 7) \\ (F_3, G_3, H_3, T_1, 5) \\ (F_1, G_3, H_3, T_3, 3) \\ (F_4, G_4, H_2, T_3, 2) \\ (F_5, G_2, H_4, T_4, 2) \\ (F_4, G_5, H_2, T_2, 1) \\ (F_1, G_6, H_5, T_1, 1) \end{pmatrix}. \tag{2}$$

[3]This depends in which stage the While loop the algorithm is, and thus the first elements of L_i will have different formats and frequencies, as in the example presented earlier.

Notice that the new data set $\overline{\mathcal{D}}$ keeps exactly the same amount of anonymity sets, as well as their sizes (which keeps the guessing probabilities fixed, which is the property **P1**). Also, regardless of the ordering by which we apply the algorithm **Alg-Min** to the sets $L_1, \ldots, L_4$ (i.e., whether we start with L_2 and L_4 firstly or not), we would still obtain the same set $\overline{\mathcal{D}}$ at the end (property **P2**).

The algorithm Alg-Max:

In order to construct a somewhat better data set than $\overline{\mathcal{D}}$ given by (2), in terms of the guessing probabilities per anonymity set, we apply the randomized merging for certain attributes. This method essentially represents the algorithm **Alg-Max**, which is formalized later on.

Hence, let us by $\mathcal{A}$ denote the first three columns in $\overline{\mathcal{D}}$ (those with formats F_i, G_i, H_i). If we randomly merge $\mathcal{A}$ and $L_4 = \{(T_1, 17), (T_2, 8), (T_3, 5), (T_4, 2)\}$, we get the new data set given as

$$\mathcal{B} = \begin{pmatrix} (F_1, G_1, H_1, T_1, 7) \\ (F_2, G_2, H_2, T_1, 4) \\ (F_5, G_2, H_4, T_1, 2) \\ (F_1, G_3, H_3, T_1, 2) \\ (F_3, G_3, H_3, T_1, 2) \\ (F_2, G_2, H_2, T_2, 2) \\ (F_1, G_1, H_1, T_3, 2) \\ (F_1, G_1, H_1, T_2, 2) \\ (F_1, G_6, H_5, T_3, 1) \\ (F_4, G_5, H_2, T_2, 1) \\ (F_4, G_4, H_2, T_2, 1) \\ (F_4, G_4, H_2, T_3, 1) \\ (F_1, G_3, H_3, T_2, 1) \\ (F_3, G_3, H_3, T_3, 1) \\ (F_3, G_3, H_3, T_2, 1) \\ (F_3, G_3, H_3, T_4, 1) \\ (F_2, G_2, H_2, T_4, 1) \end{pmatrix}.$$

Remark 2.5. It is important to note that the frequencies and merged formats in the data set $\mathcal{B}$ depend on random merging between $\mathcal{A}$ and L_4. This means if we randomize this merging more times, then one can verify that in most of the cases the number of anonymity sets in $\mathcal{B}$ is between 14 and 19. On average, $\mathcal{B}$ has two times more anonymity sets than $\mathcal{A}$.cp

The main observation here is that the random merging between the data set $\mathcal{A}$ and L_4 is imposing a significant increase in the number of anonymity sets in $\mathcal{B}$. This consequently increases the probabilities in guessing a customer's record in certain anonymity set, i.e., it increases the overall distinguishability between the customers across all attributes.

For instance, the probability of guessing a customer with the attributes (F_1, G_1, H_1, T_1), in the data set $\overline{\mathcal{D}}$ in (2) it is equal to $1/11 \approx 0.09$. However, this probability of guessing the same customer in the data set $\mathcal{B}$ is equal to $1/7 \approx 0.14$, and this is the increase by $\approx 155\%$ just after one attribute has been randomly merged. On the other hand, there is $28,1\%$ of the whole set of 32 customers (here $9/32 = 0.281$) who can be uniquely identified. In the most extreme case when all 4 lists $L_1, \ldots, L_4$ are merged randomly, one can verify that in most of the cases we get that the new obtained data set has between 28 and 32 anonymity set. *This means that almost all customers can be uniquely identified.* Now we formally but briefly state the algorithm **Alg-Max**.

Algorithm 2. [**Alg-Max**]
Input: Data set $\overline{\mathcal{D}}$ obtained by applying **Alg-Min** to lists $L_1, \ldots, L_{t-1}$ $(t \geq 2)$, and the list L_t.
Output: A data set with random merging of elements between $\overline{\mathcal{D}}$ and L_t, where L_i is a set of paired formats and frequencies $(i \in [1, t])$.

Keeping the set $\overline{\mathcal{D}}$ ordered (with respect to frequencies), merge every of its elements with randomly ordered L_t and organize it in anonymity sets with respect to formats.

3 Statistical analysis of the customized data sets

In this section, we consider the construction of various new data sets by merging 2, 3 and 4 attribute lists from $\mathcal{D}$, and provide their statistical analysis. Note that figures which are illustrating the probability distribution, are actually the probability values of guessing a specific customer in all anonymity sets for underlying custom data set. The custom data sets will be derived by applying the algorithms **Alg-Min** and **Alg-Max**. A somewhat detailed analysis will be provided for merging 2 attributes, while for merging 3 and 4 attributes we will be providing shorter analysis by emphasizing the main conclusions.

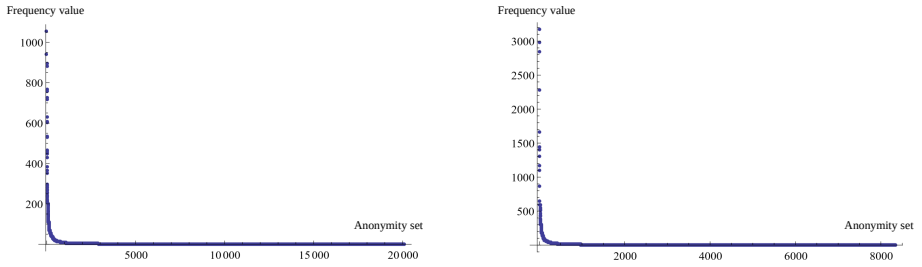
3.1 Customized data sets with 2 attributes

According to Table 1 we have that the attributes Full name, Address and City have around 80000 available data, and the E-mail list is of the size 55382. Due to the significant difference between the sizes of the attribute lists, we will consider only the case when the attribute lists Full name, Address and City are shortened to the size 80045 (by removing the smallest anonymity sets, as explained in Section 2).

The constructed data sets have been constructed by **Alg-Min** and thus have the minimized guessing probabilities of anonymity sets (equivalently, maximizes the sizes of anonymity sets). In Figs. 4a and 4b we show the frequency distribution of anonymity sets of the data sets $\mathcal{D}^{Min}_{Name/Address}$, $\mathcal{D}^{Min}_{Name/City}$ and $\mathcal{D}^{Min}_{Address/City}$.

Note that the total number of anonymity sets, corresponding to x-axis in Figs. 4a and 4b, are given by Table 2.

Figure 3: Frequency distributions in for the anonymity sets of $\mathcal{D}^{Min}_{Name/Address}$, $\mathcal{D}^{Min}_{Name/City}$, and $\mathcal{D}^{Min}_{Address/City}$



(a) for $\mathcal{D}^{Min}_{Name/Address}$ and $\mathcal{D}^{Min}_{Name/City}$

(b) for $\mathcal{D}^{Min}_{Address/City}$

Table 2: The total number of anonymity sets obtained by merging 2 attributes (**Alg-Min**).

Custom data sets	$\mathcal{D}^{Min}_{Name/Address}$, $\mathcal{D}^{Min}_{Name/City}$	$\mathcal{D}^{Min}_{Address/City}$
Number of anonymity sets	20085	8341

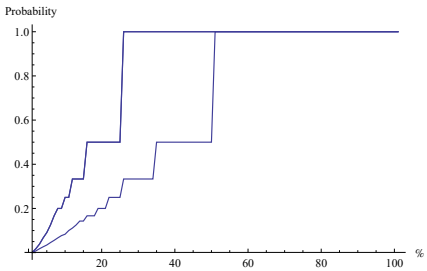
In Figs. 6a and 6b we show the guessing probabilities per anonymity sets of the data sets $\mathcal{D}^{Min}_{Name/Address}$, $\mathcal{D}^{Min}_{Name/City}$ and $\mathcal{D}^{Min}_{Address/City}$.

Note that the Address/City pair represents the lowest graph on both figures. The interpretation of the graphs is given as follows. For the data set $\mathcal{D}^{Min}_{Name/Address}$, in Fig. 6b we have that in the range between 3000 and 5000 anonymity sets the probability of guessing a customer's record is 0.5, which in Fig. 6acorresponds to the range (for quantiles) between 35% and 50%. Similarly one reads other values as well.

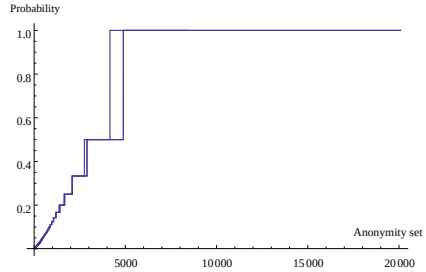
The main observation on these graphs is that in the worst case (that is for $\mathcal{D}^{Min}_{Address/City}$) we have $\approx 50\%$ anonymity sets whose anonymity sets are of size 1. Note that the probability in Fig. 6b on y-axis is showing the probability computed as $1/(\#Anonymity\ set)$, which means if the size of some anonymity set is equal to 1 (i.e., $\#Anonymity\ set = 1$), then the probability is exactly 1. Since the size of $\mathcal{D}^{Min}_{Address/City}$ is 8341, a precise measure gives exactly 4186 anonymity sets of size 1, which is on the graph approximately 50%. On the other hand, in Fig. 6b for $\mathcal{D}^{Min}_{Name/Address}$ and $\mathcal{D}^{Min}_{Name/City}$ we have exactly 15193 anonymity sets whose size is 1, which corresponds to $\approx 75\%$ of the totaly number of anonymity sets (that is 20085).

Regarding the probabilities of guessing a customer's record, one can see from Figs. 4a and 4b that they are very low if the customer belongs to larger anonymity sets. More precisely, for $\mathcal{D}^{Min}_{Name/Address}$ the largest anonymity set is of the size 1054, which gives the minimal guessing probability 1/1054. On the other hand for

Figure 5: Distributions for $\mathcal{D}^{Min}_{Name/Address}$, $\mathcal{D}^{Min}_{Name/City}$ and $\mathcal{D}^{Min}_{Address/City}$

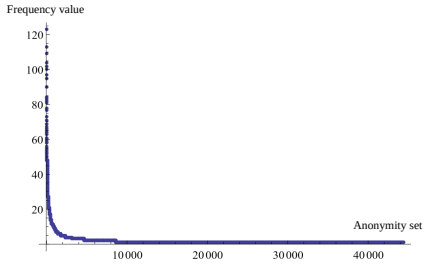


(a) The quantile distribution

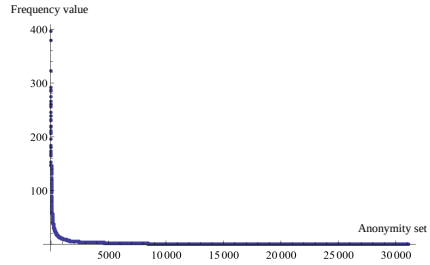


(b) The frequency distribution

Figure 7: The Frequency distributions



(a) for $\mathcal{D}^{Max}_{Name/Address}$ and $\mathcal{D}^{Max}_{Name/City}$



(b) for $\mathcal{D}^{Max}_{Address/City}$

$\mathcal{D}_{Name/Address}$ the largest anonymity set has the size 3171, which gives the probability 1/3171.

Note that the previous figures have been derived by **Alg-Min**, which represent the sub-optimal estimate of the lower guessing probabilities for considered pairs of merged attributes. Now, let us consider that the sets $\mathcal{D}^{Max}_{Name/Address}$, $\mathcal{D}^{Max}_{Name/City}$ and $\mathcal{D}^{Max}_{Address/City}$ are obtained by randomly merging the attributes (that is by **Alg-Max**). In Figs. 8a and 8b we show the sizes of the anonymity sets. We notice a huge decrease in values of the frequencies of anonymity sets and thus the corresponding guessing probabilities.

Furthermore, the number on the x-axis (Figs. 8a and 8b) shows that the number of anonymity sets has significantly increased (precisely given by Table 3), which gives a significant increase of the distinguishability between the customers with respect to 2 attributes. However, the most important observation (Fig. 9) is that the number of anonymity sets whose sizes are equal to 1 is significant. Table 3 shows precisely how many customers (out of 80045) can be uniquely identified, considering the facts that 35763 anonymity sets for $\mathcal{D}^{Max}_{Name/Address}$ and $\mathcal{D}^{Max}_{Name/City}$, and 22655 for $\mathcal{D}^{Max}_{Address/City}$, are of size 1.

Table 3: The total number of anonymity sets and probabilities of unique identifications of customer's records with respect to 2 attributes merged randomly (**Alg-Max**).

Custom data sets	$\mathcal{D}^{Max}_{Name/Address}$, $\mathcal{D}^{Max}_{Name/City}$	$\mathcal{D}^{Max}_{Address/City}$
Number of anonymity sets	44406	31101
Probabilities	44.6%	28.3%

Figure 9: The probability distribution for $\mathcal{D}^{Max}_{Name/Address}$, $\mathcal{D}^{Max}_{Name/City}$, and $\mathcal{D}^{Max}_{Address/City}$

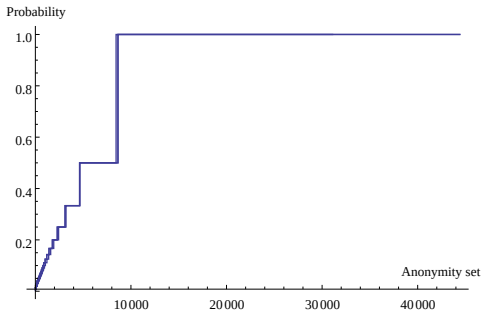
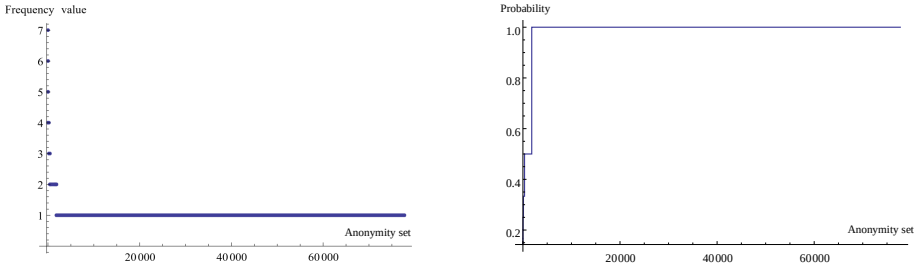


Figure 10: Distributions for $\mathcal{D}^{Max}_{Name/Address/City}$



(a) The frequency distribution

(b) The probability distribution

Remark 3.1. It is important to emphasize that the previous analysis has been performed for the attribute lists reduced to 80045. Although there is a slight analytical approach between our work and [2], we have shown that merging at least two attribute in a random way (**Alg-Max**) implies much greater distinguishability between the customers in general. More precisely, for the Name/City pair Table 3 provides a more accurate estimate than Table V given in [2]. Regarding other values given in [2, Tables V], we have the approximate matching regarding the maximal frequency in the data set $\mathcal{D}_{Name/City}$ (cf. Fig. 8a).

Remark 3.2. Although the algorithm **Alg-Min** provides the lower bounds on probabilities in terms of re-identification attacks, it is important to note that the assumption on independency between Name and City, or Address (or Email and City, etc.) is more realistic in general. This actually indicates that the realistic data sets provide a lot of distinguishability with respect to 2 attributes.

3.2 Customized data sets with 3 attributes

Let us consider the attribute lists Full name, Address and City reduced to the size 80045, and analyse the extremal/realistic merges of these attributes. Applying **Alg-Min** and merging all three attributes, we construct the data set $\mathcal{D}^{Min}_{Name/Address/City}$, which has the maximal sizes of anonymity sets. On the other hand, by $\mathcal{D}^{Max}_{Name/Address/City}$ we denote the set where the three attributes have been merged randomly (**Alg-Max**).

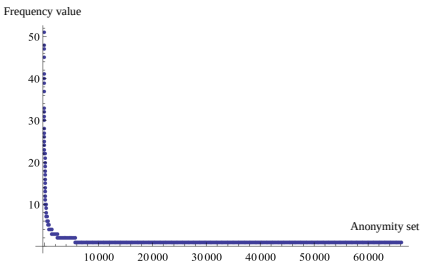
By checking the frequencies of $\mathcal{D}^{Min}_{Name/Address/City}$, we actually have that they are equal to the frequencies of $\mathcal{D}^{Min}_{Name/City}$, and thus the distribution for $\mathcal{D}^{Min}_{Name/Address/City}$ is given by Fig. 4a. On the other hand, in Figs. 11a and 11b we see that almost all customers can be uniquely identified, see Table 4.

Comparing the distribution gaps between the frequencies of $\mathcal{D}^{Min}_{Name/Address/City}$ and $\mathcal{D}^{Max}_{Name/Address/City}$, it is natural to expect that a realistic data set is much closer to $\mathcal{D}^{Max}_{Name/Address/City}$ due to the nature of the attributes. Here we mean that the Full name attribute is more expected to be independent of City and Address

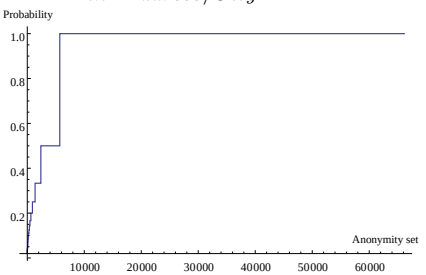
Table 4: The number and sizes of anonymity sets in $\mathcal{D}^{Max}_{Name/Address/City}$

Custom data set	Sets in total	Sets of size 1	Probability of unique identification
$\mathcal{D}^{Max}_{Name/Address/City}$	77736	75919	94.8%

Figure 12: Distributions for $\mathcal{D}^{Max-Name}_{Min-Address/City}$



(a) The frequency distribution



(b) The probability distribution

attributes, while the Address list may contain a lot of distinguishability itself for every particular customer. In terms of this expectation, in Figs. 13a and 13b we provide the distribution of the set $\mathcal{D}^{Max-Name}_{Min-Address/City}$ where we consider the attribute pair Name/Address being minimized with **Alg-Min** and then randomly merged with the Full name attribute list.

For the data set $\mathcal{D}^{Max-Name}_{Min-Address/City}$, the total number and sizes of anonymity sets of size 1 are given in Table 5.

Another interesting observation (given by Table 6) is that there exists many anonymity sets which are of size ≤ 6, and thus it provides a high probability of guessing a customer's record. Notably, the data set $\mathcal{D}^{Max-Name}_{Min-Address/City}$ is obtained by minimizing the probability of guessing the attributes Address/City, for which we realistically expect some possible correlation. This in turn gives the conclusion that the realistic data set $\mathcal{D}$, given with these three attributes, with high probability contains much smaller anonymity sets than $\mathcal{D}^{Max-Name}_{Min-Address/City}$ (small anonymity sets impose high guessing probabilities).

3.3 Customized data sets with 4 attributes

In order to consider all 4 attributes, we have to reduce the lists to the size 55382, which is the size of the E-mail attribute list. Still this makes the analysis quite interesting due to the fact that we have removed many small anonymity sets from the attribute lists Full name, Address and City. First, we consider the data sets $\mathcal{D}^{Max}_4$ and $\mathcal{D}^{Min}_4$, due to their extremal frequency distributions. Here, the index 4

Table 5: The number and sizes of anonymity sets in $\mathcal{D}^{Max-Name}_{Min-Address/City}$

Custom data set	Sets in total	Sets of size 1	Probability of unique identification
$\mathcal{D}^{Max-Name}_{Min-Address/City}$	66154	60447	75.5%

Table 6: The number and sizes of anonymity sets in $\mathcal{D}^{Max-Name}_{Min-Address/City}$

The number of anonymity sets	3325	1012	465	270	166
The size of the anonymity set	2	3	4	5	6

means that all four attributes from $\mathcal{D}$ are merged in the corresponding way, that is by **Alg-Max** or **Alg-Min** respectively. Clearly, these two data sets represent the upper and lower bounds for other custom sets that we construct.

By Table 7 we show the number and sizes of anonymity sets for the sets $\mathcal{D}^{Max}_4$ and $\mathcal{D}^{Min}_4$. Regarding the set $\mathcal{D}^{Min}_4$, Table 8 shows the number of anonymity sets whose size is quite small, which implies somewhat larger guessing probabilities. In addition, in Figs. 15a and 15b we see the frequency and probability distributions.

Remark 3.3. Note that by Table 7 we have that almost all anonymity sets in $\mathcal{D}^{Max}_4$ are of size 1, which implies the flat distribution of frequency and probability distribution.

In what follows, we analyze the custom data sets for which there exists a realistic expectation of independency between certain attributes. For instance, one may assume that the attributes Full name and E-mail are not correlated with City and Address. In this context, we may assume that Full name and E-mail are correlated, as well as City and Address. Hence, by $\mathcal{D}^{Min-City/Address}_{Min-Name/E-mail}$ we denote the set which is obtained by randomly merging the sets $\mathcal{D}^{Min}_{City/Address}$ and $\mathcal{D}^{Min}_{Name/E-mail}$. By considering their distributions in Figs. 17a and 17b and Tables 9 and 10, we see that the set $\mathcal{D}^{Min-City/Address}_{Min-Name/E-mail}$ has a high portion of customers who can be identified uniquely (approximately 54.3%). Another interesting information is that it contains many anonymity sets whose sizes are very small, which also gives high guessing

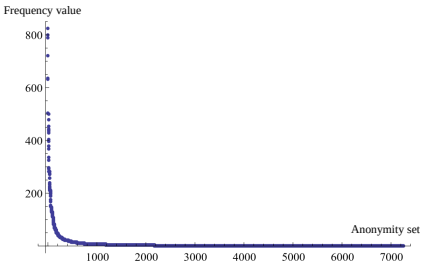
Table 7: The number and sizes of anonymity sets in $\mathcal{D}^{Max}_4$ and $\mathcal{D}^{Min}_4$.

Custom data set	Number of anonymity sets in total	Number of anonymity sets of size 1	Probability of unique identification
$\mathcal{D}^{Max}_4$	55348	55314	99.8%
$\mathcal{D}^{Min}_4$	7252	3903	7%

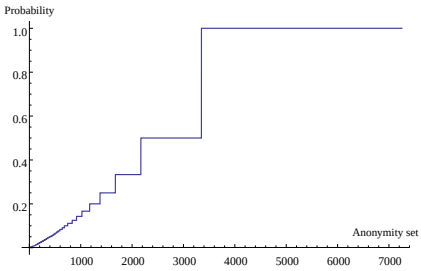
Table 8: The number and sizes of anonymity sets in $\mathcal{D}_4^{Min}$

The number of anonymity sets	1179	496	299	202	150
The size of the anonymity set	2	3	4	5	6

Figure 14: Distributions for $\mathcal{D}_4^{Min}$



(a) The frequency distribution



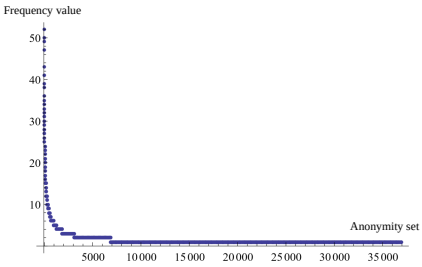
(b) The probability distribution

probabilities.

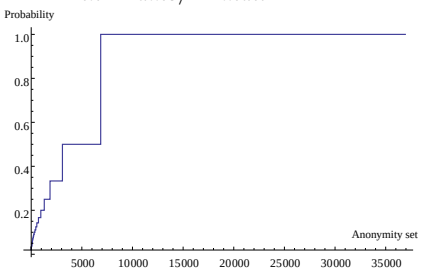
The main conclusion in this part is that the realistic data set $\mathcal{D}$, when given with all 4 attributes for each customer, is highly expected to have higher probabilities than the set $\mathcal{D}_{Min-Name/E-mail}^{Min-City/Address}$, which represents the lower bound in terms of the probability distribution. This conclusion is based on the fact that the set $\mathcal{D}_{Min-Name/E-mail}^{Min-City/Address}$ has the worst possible probability distributions for the attribute pairs City/Address and Name/E-mail (minimized by **Alg-Min**), and the fact that these two pairs are realistically expected to be independent.

Another interesting case is when we assume independence between the Full name, E-mail, and Address/City. In this case we apply **Alg-Min** only to Address/City and

Figure 16: Distributions for $\mathcal{D}_{Min-Name/E-mail}^{Min-City/Address}$



(a) The frequency distribution



(b) The probability distribution

Table 9: The number and sizes of anonymity sets in $\mathcal{D}^{Min-City/Address}_{Min-Name/E-mail}$

Custom data set	Sets in total	Sets of size 1	Probability of unique identification
$\mathcal{D}^{Min-City/Address}_{Min-Name/E-mail}$	36939	30077	54.3%

Table 10: The number and sizes of anonymity sets in $\mathcal{D}^{Min-City/Address}_{Min-Name/E-mail}$.

The number of anonymity sets	3778	1225	568	342	231
The size of the anonymity set	2	3	4	5	6

merge it independently to Full name and E-mail. Denoting this custom data set by $\mathcal{D}^{Max-Name-E-mail}_{Min-City/Address}$, in Tables 11 and 12 we see that 98.7% customer's records can be guessed uniquely. More importantly, Table 12 shows the sizes of ALL distinct anonymity sets in $\mathcal{D}^{Max-Name-E-mail}_{Min-City/Address}$, and we see that their low values imply high guessing probabilities.

4 Conclusions

The analysis presented in Sections 3.1 to 3.3 (obtained by applying algorithms **Alg-Min** and **Alg-Max**) indicates that with high probability we are able to estimate closely sizes of anonymity sets (and thus corresponding probabilities) of real data sets given with respect to more attributes, even if their attribute lists may be given separately. Consequently, the analysis shows that masked real data sets with preserved formats, statistically (in terms of re-identification attacks) *leak a lot of information.*

Table 11: The number and sizes of anonymity sets in $\mathcal{D}^{Max-Name-E-mail}_{Min-City/Address}$

Custom data set	Sets in total	Sets of size 1	Probability of unique identification
$\mathcal{D}^{Max-Name-E-mail}_{Min-City/Address}$	55020	54684	98.7%

Table 12: The number and sizes of ALL anonymity sets in $\mathcal{D}^{Max-Name-E-mail}_{Min-City/Address}$

The number of anonymity sets	54684	316	17	2	1
The size of the anonymity set	1	2	3	4	7

References

[1] Olusola Olajide Ajayi and Temidayo Olarewaju Adebiyi. "Application of Data Masking in Achieving Information Privacy". In: *IOSR Journal of Engineering* 4.2 (2014), pp. 13–21.

[2] Zaruhi Aslanyan and Martin S. Boesgaard. "Privacy Analysis of Format-Preserving Data-Masking Techniques". In: *2019 12th CMI Conference on Cybersecurity and Privacy (CMI)*. 2019, pp. 1–6. DOI: 10.1109/CMI48017.2019.8962143.

[3] InfoWatch Analytics Center. *Data Breach Report: A Study on Global Data Leaks in H1 2018*. https://infowatch.com/report2018_half. 2018.

[4] Baojiang Cui, BaiHui Zhang, and Kaiyue Wang. "A Data Masking Scheme for Sensitive Big Data Based on Format-Preserving Encryption". In: *2017 IEEE International Conference on Computational Science and Engineering, CSE 2017, and IEEE International Conference on Embedded and Ubiquitous Computing, EUC 2017, Guangzhou, China, July 21-24, 2017, Volume 1*. IEEE Computer Society, 2017, pp. 518–524. DOI: 10.1109/CSE-EUC.2017.97. URL: https://doi.org/10.1109/CSE-EUC.2017.97.

[5] Josep Domingo-Ferrer, Oriol Farràs, Jordi Ribes-González, and David Sánchez. "Privacy-preserving cloud computing on sensitive data: A survey of methods, products and challenges". In: *Comput. Commun.* 140-141 (2019), pp. 38–60. DOI: 10.1016/j.comcom.2019.04.011. URL: https://doi.org/10.1016/j.comcom.2019.04.011.

[6] Ruby Jain and Manimala Puri. "Protection and privacy of sensitive data: Challenges and Opportunities". In: *IICMR International Research Journal* I^4 11.1 (2016), pp. 12–16.

[7] Net 2000 Ltd. *Data Masking: What You Need to Know*. http://www.datamasker.com/DataMasking_WhatYouNeedToKnow.pdf. 2016.

[8] Net 2000 Ltd. *Data Sanitization Techniques*. http://www.datamasker.com/datasanitization_whitepaper.pdf. 2010.

[9] Mariano Di Martino, Pieter Robyns, Winnie Weyts, Peter Quax, Wim Lamotte, and Ken Andries. "Personal Information Leakage by Abusing the GDPR 'Right of Access'". In: *Fifteenth Symposium on Usable Privacy and Security, SOUPS 2019, Santa Clara, CA, USA, August 11-13, 2019*. Ed. by Heather

Richter Lipford. USENIX Association, 2019. URL: https://www.usenix.org/conference/soups2019/presentation/dimartino.

[10] Ron Ben Natan. *Implementing database security and auditing*. Elsevier, 2005. ISBN: 978-1-55558-334-7.

[11] NIST. *Special Publication 800-38G, Recommendation for Block Cipher Modes of Operation: Methods for Format-Preserving Encryption*. http://dx.doi.org/10.6028/NIST.SP.800-38G [accessed 10/04/2018].

[12] Lucila Ohno-Machado, Paulo Sérgio Panse Silveira, and Staal A. Vinterbo. "Protecting patient privacy by quantifiable control of disclosures in disseminated databases". In: *Int. J. Medical Informatics* 73.7-8 (2004), pp. 599–606. DOI: 10.1016/j.ijmedinf.2004.05.002. URL: https://doi.org/10.1016/j.ijmedinf.2004.05.002.

[13] GK Ravikumar, TN Manjunath, Ravindra S Hegadi, and IM Umesh. "A survey on recent trends, process and development in data masking for testing". In: *International Journal of Computer Science Issues (IJCSI)* 8.2 (2011), p. 535.

[14] Ricardo Jorge Santos, Jorge Bernardino, and Marco Vieira. "Balancing Security and Performance for Enhancing Data Privacy in Data Warehouses". In: *IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2011, Changsha, China, 16-18 November, 2011*. IEEE Computer Society, 2011, pp. 242–249. DOI: 10.1109/TrustCom.2011.33. URL: https://doi.org/10.1109/TrustCom.2011.33.

[15] Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati. "Over-encryption: Management of Access Control Evolution on Outsourced Data". In: *Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23-27, 2007*. Ed. by Christoph Koch, Johannes Gehrke, Minos N. Garofalakis, Divesh Srivastava, Karl Aberer, Anand Deshpande, Daniela Florescu, Chee Yong Chan, Venkatesh Ganti, Carl-Christian Kanne, Wolfgang Klas, and Erich J. Neuhold. ACM, 2007, pp. 123–134. URL: http://www.vldb.org/conf/2007/papers/research/p123-decapitani.pdf.