# 7th Workshop on Current Trends in Cryptology (CTCrypt 2018)

May 28-30, 2018, «Pushkarskaya sloboda», Suzdal, Russia.

## Pre-proceedings

**In cooperation**

**General partner**

**Official partner**

**Partners**

**Expo partner**

**Support**

**Media partners**

**CTCrypt 2018 is organized by**

– Technical Committee for Standardization «Cryptography and security mechanisms» (TC 26), Russian Federation

– Steklov Mathematical Institute of Russian Academy of Science

– Academy of Cryptography of the Russian Federation

# Steering Committee

## Co-chairs

| | | |
|---|---|---|
| Aleksandr Shoitov | – | Academy of Cryptography, Russia |
| Vladimir Sachkov | – | Academy of Cryptography, Russia |
| Igor Kachalin | – | TC 26, Russia |

## Steering Committee Members

| | | |
|---|---|---|
| Mikhail Glukhov | – | Academy of Cryptography, Russia |
| Dmitry Matyukhin | – | TC 26, Russia |
| Andrei Zubkov | – | Steklov Mathematical Institute of Russian Academy of Science, Russia |
| Andrei Pichkur | – | Educational and Methodical Association of Higher Educational Institutions of Russia on Education in Information Security, Russia |

# Program Committee

## Co-chairs

| | | |
|---|---|---|
| Alexander Lapshin | – | Academy of Cryptography, Russia |
| Dmitry Matyukhin | – | TC 26, Russia |
| Andrei Zubkov | – | Steklov Mathematical Institute of Russian Academy of Science, Russia |

## Program Committee Members

| | | |
|---|---|---|
| Igor Kruglov | – | Academy of Cryptography, Russia |
| Ivan Chizhov | – | Lomonosov Moscow State University, Russia |
| Eduard Primenko | – | Lomonosov Moscow State University, Russia |
| Andrei Zyazin | – | Moscow Technological University, Russia |
| Sergey Checheta | – | Educational and Methodical Association of Higher Educational Institutions of Russia on Education in Information Security, Russia |
| Alexey Tarasov | – | Educational and Methodical Association of Higher Educational Institutions of Russia on Education in Information Security, Russia |
| Alexey Alexandrov | – | Vladimir State University, Russia |
| Grigory Marshalko | – | TC 26, Russia |
| Vasily Shishkin | – | TC 26, Russia |
| Stanislav Smyshlyaev | – | TC 26, Russia |
| Sergey Agievich | – | Research Institute for Applied Problems of Mathematics and Informatics, Belarus |
| Yury Kharin | – | Research Institute for Applied Problems of Mathematics and Informatics, Belarus |
| Markku-Juhani Olavi Saarinen | – | ARM, Great Britain |
| Atul Luykx | – | Visa, USA |
| Igor Semaev | – | Bergen University, Norway |
| Amr Youssef | – | Concordia University, Canada |
| Andrey Bogdanov | – | Technical University of Denmark, Denmark |

## External Reviewers

Liliya Akhmetzianova, Grigory Sedov, Vasily Nikolaev,
Maxim Nikolaev, Alexey Sarantsev, Andrey Trishin,
Evgeny Alekseev, Denis Fomin, Nikolay Varnovsky,
Michael Koypish

**Dear colleagues!**

For seven consecutive years the «Current Trends in Cryptology» workshop gathers together leading national and foreign cryptography specialists. The year 2018 has seen 23 papers provided by authors from 5 countries. 16 papers have been included into the workshop program after a thorough review by members of a program committee, which is traditionally international. I would like to point out that this year an issue of including of some articles into the program provoked a lot of debates among the program committee members. At the same time, since the workshop is a place to share new ideas, views, and conceptions it was unanimously agreed to give authors of the disputable articles an opportunity to represent the results of their researches and to talk them over with the workshop's audience.

More than 100 delegates from 9 countries all over the world have registered for participation in the $7^{\text{th}}$ workshop. The number approximately corresponds to that of the previous years.

Subject of the included to the workshop program articles is vast and cover issues concerning synthesis and analysis of specific cryptographic mechanisms as well as fundamental problems of cryptography. An actively discussed post-quantum cryptography will be also mentioned.

In accordance with an established practice, scientific part of the Workshop will be enlarged by panel discussions of the most actual and urgent cryptography issues. Traditionally, the discussions are attended by representatives of cryptographic devices developers and producers, scientific community and regulatory authorities. Two panel discussions with an extremely pressing subjects, in our opinion, will be organized this year. The first one will be dedicated to the role and place of cryptography in digitalization of the society and, particularly, within the current «Digital Economy of the Russian Federation» program.

The second one will be devoted to the issues concerning peculiarities and distinctive aspects of blockchain and distributed ledger technology-based systems. Similar issues were raised within a last year's workshop panel discussion that appeared to be successful both in the entry list and discussed topics. This year we are to discuss, among others, changes in views on these technologies and on approaches of its implementation have undergone over the past year as well as issues of realization and non-realization of announced projects.

The workshop «Current Trends in Cryptology» has proved itself to be the place where everyone can take a look at state-of-the-art results of the researches conducted by foreign cryptography specialists and, moreover, ask

them questions in person. Lightweight cryptography – one of the most popular direction of modern cryptography – is the subject of the report to be delivered by Thomas Peyrin. Phillip Rogaway, one of the «provable security» founders, will speak on authenticated encryption, another equally discussed issue. Aleksandr A. Nechayev's disciple Oleg Kozlitin, invited speaker of the Academy of Cryptography of the Russian Federation – one of the organizers of the Workshop – will share his research results in mathematical problems in cryptography.

The range of the discussed questions, high level professionalism participants, and impartial selection of papers allows to consider the workshop «Current Trends in Cryptology» to be a leading Russian scientific forum on cryptography.

*Dear colleagues, we are facing three days of difficult but interesting work the results of which, I hope, will allow to develop the existing approaches and lay the groundwork for the new ones to solve the tasks which the modern society comes across increasingly in course of digitalization. Thereon I would like to declare the $7^{th}$ workshop «Current Trends in Cryptography» open.*

President of the Academy of Cryptography of the Russian Federation

Aleksandr Shoitov

# Invited Talks

# The Rise of Authenticated Encryption

Phillip Rogaway

Department of Computer Science University of California, Davis, USA

**Abstract**

To many theory-oriented cryptographers, symmetric encryption is among our most passé of problems. Yet from the point of view of providing a useful theory and desirable schemes, the area is very much alive. For this talk I'll explore the long dialectic that has taken us from semantic security to robust authenticated-encryption. I'll trace the history of AE, explaining why it emerged, how it has evolved, and what some modern AE schemes now look like.

# Pseudorandom Generators Based on Shift Registers Over Finite Commutative Rings

Oleg Kozlitin

Academy of Cryptography of the Russian Federation

**Abstract**

One of the most popular ways of pseudo-random sequence constructing is to use the shift registers over finite commutative rings. The shift register with linear feedback function (LFSR) and non-linear output function is a classical representative of such kind generators. There are a lot of cryptographic parameters of LFSR which have to be studied. Among them are the periodical properties, the statistical properties and the linear complexity (rank) of output sequence. Consideration of these parameters is a main content of many papers. In that connection, mention must be made of the articles by A.A. Nechaev, V.L. Kurakin, A.S. Kuzmin, A.V. Mikhalev, V.N. Tsypyshev, O.V. Kamlovskiy, D.N. Bylkov and other authors.

The aim of this report is to describe two methods of generalization of the classical LFSR. The first method is to use polynomial feedback function. The pseudo-random generator with polynomial feedback function is called polynomial generator. The polynomial shift register is a special case of the polynomial generator. The polynomial generator over residue ring was investigated in the articles by V.S. Anashin and M.V. Larin. Some results about periodical properties of polynomial generator over arbitrary finite commutative ring with identity were received by V.E. Viktorenkov. The cycle structure of polynomial generator over Galois ring was described by D.M. Ermilov.

In this report the periodical properties of polynomial generator over finite chain ring (finite commutative local ring of main ideals) and the cycle structure of multidimensional polynomial generator over Galois ring will be discussed.

The second method of generalization is to use several linear feedback functions instead of one linear feedback function. The use of multidimensional linear shift register (k-LFSR) is one of the possible ways to solve this problem. Originally, k-LFSR was proposed by Japanese mathematicians T. Nomura and A. Fukuda as a decoder of two-dimensional cyclic code. Later this automation was studied by A.A. Nechaev, A.V. Mikhalev, V.L. Kurakin and A.S. Kuzmin as a pseudo-random generator. Since 2003, the so-called self-controlled k-LFSR has been investigated. Some cryptographic properties of output sequence of self-controlled 2-LFSR over Galois ring will be described in this report. Also we will discuss the ways of optimal choice of the automation's parameters (an output function and a control function).

# Lightweight Symmetric-Key Cryptography

Thomas Peyrin

Nanyang Technological University, Singapore

**Abstract**

In this talk, we will review the current state of the art of lightweight cryptography, a recent trend in symmetric-key cryptography design that aims at providing secure algorithms for very constrained devices such as RFID tags. After a short introduction to cryptographic primitives design, we will first discuss the problems faced by the research community to come up with such specific lightweight algorithms and why previous solutions are often not fit for such use cases. Then, we will explain the current solutions that have emerged and identify some questions that remain open.

# Cryptography for Society. Security Issues from Common Users' Point of View

Dmitry Malinkin

OJSC «Rostelecom», Search engine «Sputnik», Moscow, Russia

**Abstract**

Cryptography initially served the state and military interests, it was considered important only for domain experts. But since the World Wide Web has entangled the globe, cryptographic methods were brought out to common people. Almost everyone is now using encryption: politicians, law enforcers, developers and large IT companies, hackers, fighters for data privacy and freedom of information and common people who want to keep their data private. Moreover, Russian users often do not even know that they use a secure channel in an instant messenger or a browser with secure encryption. But more and more people start asking questions: *Who created the encryption we're using?* If it is the foreign company, then will our personal information be available to Western intelligence services? In exchange, Russian IT companies are considering other questions: *What will happen if Western cryptography methods are replaced by Russian ones? How will common users react to this? What are the risks? How not to damage the fragile electronic Government-Citizen interaction through the transition to Russian cryptography?* The very interaction in which a lot of effort and resources is already invested, and thanks to which the digital dialogue between the state and its citizens has become simple, plain and fast.

# From «Hype» to Practice

Maxim Shevchenko

JSC «InfoTeCS», Moscow, Russia

**Abstract**

We saw a lot of pilot projects started last year and aimed to implement blockchain-based solutions into the real economic sectors. The developers faced quite a lot of issues to be solved to meet the requirements of legislation and regulations, as well as a number of practical problems, e.g. of ensuring the security of users private data.

These problems are now addressed in national and international standardization workflows. ISO/TC307 started 8 projects focused on the essentials problems of the technology. A TC26 Working Group prepared the draft of the first Russian guideline for the blockchain terminology. These terminology guidelines might be a basement for a conceptual framework of the technology.

During my speech I'll highlight these issues to start a discussion during the panel discussion.

# Securely Scaling Distributed Ledger Systems

Philipp Jovanovic

École Polytechnique Fédérale de Lausanne, Switzerland

**Abstract**

Designing a secure permissionless distributed ledger that performs on par with centralized payment processors such as Visa is challenging. Most existing distributed ledgers are unable to «scale-out» – growing total processing capacity with number of participants – and those that do compromise security or decentralization. This work presents OmniLedger, the first scale-out distributed ledger that can preserve long-term security under permissionless operation. OmniLedger ensures strong correctness and security by using a bias-resistant public randomness protocol to choose large statistically representative shards to process transactions, and by introducing an efficient cross-shard commit protocol to handle transactions affecting multiple shards atomically. In addition, OmniLedger optimizes performance via scalable intra-shard parallel transaction processing, ledger pruning via collectively-signed state blocks, and optional low-latency «trust-but-verify» validation of low-value transactions. Evaluation of our working experimental prototype shows that OmniLedger's throughput scales linearly in the number of validators available, supporting Visa-level workloads and beyond, while confirming typical transactions in under two seconds.

# An Authentication Language for Blockchain Based on Σ-Protocols Enhanced by Boolean Predicates

Alexander Chepurnoy

IOHK Research, Sestroretsk, Russia
alex.chepurnoy@iohk.io

**Abstract**

Every coin in Bitcoin is protected by a program in the stack-based Script language. An interpreter for the language is evaluating the program against a redeeming program (in the same language) as well as a context (few variables containing information about a spending transaction and the blockchain), producing a single boolean value as a result. While Bitcoin Script allows for some contracts to be programmed, its abilities are limited while many instructions were removed after denial-of-service attacks or security issues discovered. Also, to add new cryptographic primitives, for example, ring signatures, a hard-fork is required.

Generalizing the Bitcoin Script, we introduce a notion of an *authentication language* where a verifier is running an interpreter which three inputs are a *proposition* defined in terms of the language, a *context* and also a *proof* generated by a prover for the proposition against the same context. The interpreter is deterministically producing a boolean value and must finish evaluation for any possible inputs within concrete constant time.

We propose an alternative authentication language, named Σ-State. It defines guarding proposition for a coin as a logic formula which combines predicates over a context and cryptographic statements provable via Σ-protocols with and, or, k-out-of-n connectives. A prover willing to spend the coin first reduces the compound proposition to a (possibly complex) cryptographic statement by evaluating predicates over known shared context (state of the blockchain system and a spending transaction). Then the prover is turning a corresponding Σ-protocol into a signature with the help of a Fiat-Shamir transformation. A verifier (a full-node in a blockchain setting) checks the proposition against the context and the signature. Language expressiveness is defined by a set of predicates over context and a set of cryptographic statements. We show how the latter could be updated with a soft-fork by using a language like ZKPDL [1], and how the former could be updated with a soft-fork by using versioning conventions. We propose a set of context predicates for a Bitcoin-like cryptocurrency with a guarantee of constant upper-bound verification time. We provide several examples: ring and threshold signatures, pre-issued mining rewards, crowdfunding, and demurrage currency.

**References**

[1] Sarah Meiklejohn et al. Zkpdl: A language-based system for efficient zero-knowledge proofs and electronic cash. In *USENIX Security Symposium*, volume 10, pages 193–206, 2010.

# Contents

## Symmetric cryptography



## Algebraic aspects

## Public key cryptography



## Probabilistic aspects and Applications

# Symmetric cryptography

# Near Birthday Attack On «8 bits» AEAD Mode

Liliya Ahmetzyanova, Grigory Karpunin, and Grigory Sedov

Crypto-Pro LLC, Moscow, Russia
{lah, karpunin, sedovgk}@cryptopro.ru

**Abstract**

This work describes an attack on the «8 bits» authentication encryption with associated data (AEAD) mode proposed during the AEAD standardization process of the Russian Technical Committee for Standardization TC 26. The «8 bits» mode is similar to the CCM mode [9] except for several design features. We show that these distinctive features allow to construct a near birthday attack on «8 bits» mode. We also propose countermeasures to resist suggested attack.

**Keywords:** «8 bits» mode, birthday attack, AEAD forgery.

## 1 Introduction

Authenticated encryption schemes, which aim at providing both confidentiality and integrity of data, have gained renewed attention in the light of the recently commenced CAESAR competition [2].

The AEAD modes are the most widely spread subset of authenticated encryption schemes which allow to additionally process associated data that needs to be authenticated but not encrypted. The importance of the AEAD schemes development is explained by the exploitation simplicity thereof they are much easier to implement properly than MAC and encryption schemes separately under random and independent keys. Also when using the AEAD scheme we can reduce the key size, state size, and improve the data processing speed. Another advantage of such schemes is their transparent embedding into high-level schemes and protocols because there is no need of using additional diversifications for enough key material generation. For example, the use of such schemes is supposed to be mandatory for the Record protocol in TLS 1.3 [6].

The AEAD scheme named «8 bits» was proposed during the AEAD standardization process of the Russian Technical Committee for Standardization TC 26 and was presented at the seminar «Mathematical methods of cryptanalysis» in MSU. This scheme is based on the standardized blockcipher

modes of operation CTR and OMAC. The prototype of «8 bits» is the CCM mode [9] which is standardized in IEEE 802.11i [3]. The crucial difference between CCM and «8 bits» is the absence of additional tag encryption that causes a near birthday attack.

Although the birthday bound is sufficient for practical applications there are some AEAD modes with no applicable attacks on the authentication with near birthday complexity [1, 9, 7]. Thus we claim that the near birthday complexity attack should be considered as a flaw in the construction of AEAD scheme.

The current paper contains the description of the above-mentioned attack and is organized as follows. In Section 2 we provide basic definitions and remind the reader of some notions, in Section 3 the definition of «8 bits» is provided, and in Section 4 we describe the above-mentioned attack.

## 2 Preliminaries and Basic Definitions

By $V_n$ we denote the set of $n$-component bit strings. Also we consider $V_n$ as a vector space over field $\mathbb{F}_2 = \{0, 1\}$. Let $V^*$ be the set of all bit strings of finite length. For nonnegative integers $l$ and $i$ let $\mathrm{str}_l(i)$ be a $l$-bit representation of $i$ with the least significant bit on the right. For a nonnegative integer $l$ and a bit string $M \in V_l$ let $\mathrm{int}(M)$ be an integer $i$ such as $\mathrm{str}_l(i) = M$.

For a bit string $M$ and a positive integer $l \leqslant |M|$ let $\mathrm{msb}_l(M)$ ($\mathrm{lsb}_l(M)$) be the string, consisting of the leftmost (rightmost) $l$ bits of $M$.

For bit strings $A$ and $B$ by $A\|B$ we denote their concatenation. For the bit string $A$ by $A^n$ we denote the string $A$ concatenated $n$ times. Let $|M|$ be the bit length of the string $M$.

For any set $A$, define $Perm(A)$ as the set of all bijective mappings from $A$ to $A$ (permutations on $A$).

A block cipher is a mapping $E\colon V_k \times V_n \to V_n$ such that for all $K \in V_k$ mapping $E(K, \cdot)$ is a permutation on $V_n$. By $n$ and $k$ we denote the block size and the key size respectively. By $E_K(\cdot)$ we denote the mapping $E(K, \cdot)$.

We model an adversary using an interactive probabilistic algorithm that has access to one or more oracles. The resources of $A$ are measured in the terms of time and query complexities. For a fixed model of computation and a method of encoding the time complexity includes the description size of $A$. The query complexity usually includes the number of queries and the maximal length of queries or the total legth of queries.

## 3 «8 bits» AEAD Mode

The «8 bits» mode is defined for the block size $n = 128$ bit. The additional parameter of the mode is a tag size $s \leqslant n$. This parameter should be fixed and known both by the sender and the receiver.

Both the sender and the receiver know the secret key $K$ used for computing a tag and a ciphertext. By $P \in V^*$ and $A \in V^*$ we denote a plaintext and the associated data respectively. The length of both $P$ and $A$ must be less than $2^{64}$ bits. $P$ also must have non-zero length. The pair of a plaintext and the associated data we will call «a message». This mode uses also an initialization vector $IV \in V_{56}$. This vector must be unique for each new message.

### 3.1 OMAC Algorithm

The integrity and authenticity in «8 bits» is achieved using the OMAC algorithm. Let us remind the reader of the main idea of the OMAC computation. The detailed description of OMAC can be found in [4].

The algorithm starts with a derivation of MAC keys. The derivation is made as follows.

$$R = E_K(0^{128}),$$
$$K_1 = \begin{cases} R \ll 1, & \text{if } \mathrm{msb}_1(R) = 0; \\ (R \ll 1) \oplus B_{128}, & \text{otherwise;} \end{cases} \quad (1)$$
$$K_2 = \begin{cases} K_1 \ll 1, & \text{if } \mathrm{msb}_1(R) = 0; \\ (K_1 \ll 1) \oplus B_{128}, & \text{otherwise,} \end{cases}$$

where $B_{128} = 0^{120}||10000111$.

Then the message $M \in V^*$ is divided into $t = \left\lceil \frac{|M|}{n} \right\rceil$ blocks $M_1, \ldots, M_{t-1} \in V_n$ and $M_t \in V_r$, $r \leqslant n$, that $M = M_1 \| \ldots \| M_{t-1} \| M_t$. The OMAC algorithm then can be described as follows (Pseudocode 1).

$\underline{OMAC^{(s)}(M = M_1\| \ldots \|M_t, K)}$

1: $C_0 = 0^n$
2: $C_i = E_K(C_{i-1} \oplus M_i), i = 1, \ldots, t-1$
3: **if** $|M_t| = n$ **then**
4: $\quad K^* = K_1,\ M^* = M_t$
5: **else**
6: $\quad K^* = K_2,\ M^* = M_t\|1\|0^{n-|M_t|-1}$
7: $T = \mathrm{msb}_s(E_K(C_{t-1} \oplus M^* \oplus K^*))$
8: **return** $T$

Pseudocode 1: The OMAC algorithm

The mode structure is illustrated by Figure 2. By $\mathsf{MAC}_K^{(s)}(M)$ in the «8 bits» description the computation of the tag of size $s$ under the key $K$ according to the OMAC algorithm is denoted.

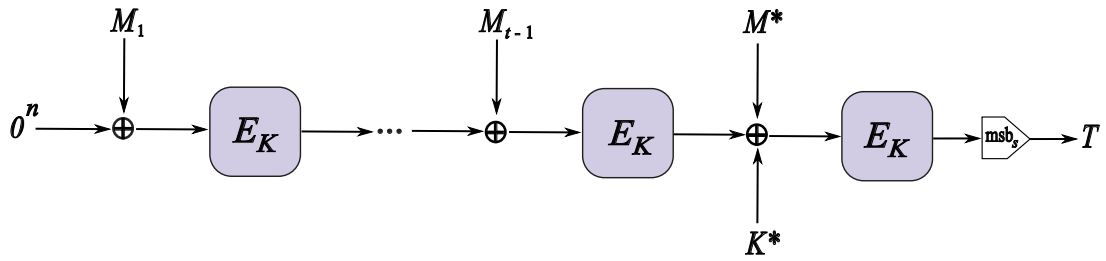



Figure 2: The OMAC algorithm which takes the message $M = M_1\| \ldots \|M_t \in V^*$, $t = \lceil |M|/n \rceil$, and the key $K \in V_k$ as inputs and outputs the tag $T \in V_s$.

## 3.2 The «8 bits» authenticated encryption and decryption algorithms

Let $t$ be the length of a plaintext in blocks, i.e. $t = \left\lceil \frac{|P|}{128} \right\rceil$, $s \leqslant n$ be the tag size and $d = \left\lceil \frac{|P|+|A|+72}{128} \right\rceil$. The «8 bits» authenticated encryption and decryption algorithms can be described as follows (see Pseudocode 3, Pseudocode 4). The probabilistic key generation algorithm 8bits.$\mathcal{K}()$ outputs the key $K \xleftarrow{\mathcal{U}} V_k$.

$\underline{8\text{bits}^{(s)}.\mathcal{E}(K, IV, P, A)}$

$S_i = 0^8 \| IV \| \text{str}_{64}(i), i = 1, \ldots, t$
$\Gamma = E_K(S_1) \| E_K(S_2) \| \ldots \| E_K(S_t)$
$C = P \oplus \text{msb}_{|P|}(\Gamma)$
**if** $|A| = 0$ **then**
    $F = 1^7 \| 0$
**else**
    $F = 1^7 \| 1$
$B = F \| \text{str}_8(s) \| IV \| A \| C \| 0^{128d-|C|-|A|-72} \| \text{str}_{64}(|A|) \| \text{str}_{64}(|C|)$
$T = \mathsf{MAC}_K^{(s)}(B)$
**return** $C \| T$

Pseudocode 3: Authenticated Encryption Algorithm of the «8 bits»mode

$\underline{8\text{bits}^{(s)}.\mathcal{D}(K, IV, C \| T, A, s)}$

**if** $|A| = 0$ **then**
    $F = 1^7 \| 0$
**else**
    $F = 1^7 \| 1$
$B = F \| \text{str}_8(s) \| IV \| A \| C \| 0^{128d-|C|-|A|-72} \| \text{str}_{64}(|A|) \| \text{str}_{64}(|C|)$
$T' = \mathsf{MAC}_K^{(s)}(B)$
**if** $T \neq T'$ **then**
    **return** $\perp$
$S_i = 0^8 \| IV \| \text{str}_{64}(i), i = 1, \ldots, t$
$\Gamma = E_K(S_1) \| E_K(S_2) \| \ldots \| E_K(S_t)$
$P = C \oplus \text{msb}_{|C|}(\Gamma)$
**return** $P$

Pseudocode 4: Authenticated Decryption Algorithm of the «8 bits» mode

# 4 Attack

## 4.1 Adversary model

The standard model relevant for analyzing the AEAD security is the IND-CCA3 model [8] which allows to investigate the security of proposed scheme from the point of view of both integrity and confidentiality. The work [8] states that the scheme is IND-CCA3-secure iff it is IND-CPA- (confidentiality) and Auth-secure (integrity). In the current work we consider the Auth model in detail and describe the attack which capabilities are covered by this model.

**Definition 1.** *The advantage of the adversary $\mathcal{A}$ in the Auth model for the* AEAD *mode is defined as follows:*

$$\mathsf{Adv}^{Auth}_{\mathsf{AEAD}}(\mathcal{A}) = \Pr\left[\mathbf{Exp}^{Auth}_{\mathsf{AEAD}}(\mathcal{A}) = 1\right],$$

*where* $\mathbf{Exp}^{Auth}_{\mathsf{AEAD}}(\mathcal{A})$ *is described in the following way:*

$\underline{\mathbf{Exp}^{\mathrm{Auth}}_{\mathsf{AEAD}}(\mathcal{A})}$
$K \xleftarrow{\$} \mathsf{AEAD}.\mathcal{K}()$
$sent \leftarrow \varnothing$
$(IV', C'\|T', A') \leftarrow \mathcal{A}^{\mathsf{Encrypt}}$
$P \leftarrow \mathsf{AEAD}.\mathcal{D}(K, IV', C'\|T', A')$
**if** $(C'\|T', A') \notin sent$ **and** $P \neq \perp$
**then**
  $win \leftarrow 1$
**else**
  $win \leftarrow 0$
**end if**
**return** $win$

$\underline{\mathsf{Oracle\ Encrypt}(IV, P, A)}$
$C\|T \xleftarrow{\$} \mathsf{AEAD}.\mathcal{E}(K, IV, P, A)$
$sent \leftarrow sent \cup \{(A, C\|T)\}$
**return** $C\|T$

The Auth model allows the adversary to choose adaptively messages for encryption and receive their ciphertexts and tags. The adversary's goal is to make the receiver to accept a «non-authentic» pair of ciphertext $C$ and the associated data $A$. In the Auth model the «non-authentic» message means it was never transmitted by the sender (satisfied the condition $\notin send$).

Now consider the «8bits» scheme with $s = 128$. Suppose that $IV$ is generated with the use of a counter, i.e. for each new message the initialization vector $IV'$ takes value $\mathrm{str}_{56}(\mathrm{int}_{56}(IV)+1)$ if the previous initialization vector was $IV$. Let the first initialization vector be $0^{56}$.

## 4.2 Attack details

The main idea behind the proposed attack is to exploit the information received from obtained ciphertexts to «break» integrity. In the model relevant for schemes providing integrity the adversary can observe only tags, while in the AEAD schemes case the adversary can get additional information from received ciphertexts. The proposed attack uses the possibility of getting enciphered counter values to find collision with the OMAC values. It results in the possibility of modifying undetectively the associated data length stored in the last block of the OMAC-processed string.

**The first stage.** Let us introduce a parameter $l$ such that $6 \leqslant l < 55$. Then we make $2^l$ queries $P_1, P_2, \ldots, P_{2^l}$, $|P_i| = 2^{64} - 128$, with empty associated data to the Encrypt oracle. The Encrypt oracle returns the corresponding ciphertexts $C_1, C_2, \ldots, C_{2^l}$. Note that for the message $P_i$ the corresponding initialization vectors are $IV_i = \mathrm{str}_{64}(i-1)$. By $\mathcal{IV}' = \{IV_i |\ i = 1, \ldots, 2^l\}$ we denote the set of all initialization vectors for the messages $P_1, P_2, \ldots, P_{2^l}$.

Note that the messages length is the greatest possible length multiple of the block size. The number of 128-bit blocks in any of these messages is equal to $2^{57} - 1$.

Let $P_i[j]$ be the $j$-th 128-bit block of the message $P_i$ and $S_i[j]$ be the string $0^8 \| IV \| \mathrm{str}_{64}(j)$ which is used for ciphering the block $P_i[j]$: $C_i[j] = P_i[j] \oplus E_K(S_i[j])$. Denote by $\mathcal{S} = \{S_i[j] |\ i = 1, \ldots, 2^l,\ j = 1, \ldots, 2^{57} - 1\}$ the set of all strings $S_i[j]$.

Given the plaintext blocks $P_i[j]$ and the ciphertext blocks $C_i[j]$ at the end of this stage we get the keystream blocks $\Gamma_i[j] = E_K(S_i[j])$ for all strings $S_i[j]$ from $\mathcal{S}$.

**The second stage.** At the second stage we compute $2^{56} - 2^l$ values of ciphertexts and OMAC tags for all remained values of initialization vectors $\mathcal{IV}'' = V_{56} \setminus \mathcal{IV}'$. More accurately, we make $2^{56} - 2^l$ queries $(IV, P, A)$ to the $\mathsf{Encrypt}$ oracle, where $IV \in \mathcal{IV}''$, $P = 0^1 \in V_1$, $A = 0^1 \in V_1$. For query $(IV, P, A)$ the $\mathsf{Encrypt}$ oracle returns a pair $C \| T$, where $C$ is the ciphertext and $T$ is a tag. The tag $T$ is computed by the $\mathsf{MAC}_K^{(128)}$ algorithm with an input $B$ formatted as follows

$$B = \underbrace{F \| \mathrm{str}_8(128) \| IV \| A \| C \| 0^{54}}_{B_0} \| \underbrace{\mathrm{str}_{64}(1) \| \mathrm{str}_{64}(1)}_{B_1}.$$

Since $B$ consists of only two blocks, the tag $T$ is equal to $E_K(E_K(B_0) \oplus K_1 \oplus B_1)$.

Note that for any new $IV$ the string $B_0$ is new and $B_1$ is constant and equal to $\mathrm{str}_{64}(1) \| \mathrm{str}_{64}(1)$. By $\mathcal{B} = \{(F \| \mathrm{str}_8(128) \| IV \| A \| C \| 0^{54}) |\ IV \in \mathcal{IV}''\}$ we denote the set of all such blocks $B_0$.

Thus at the end of this stage we have the OMAC tags $E_K(E_K(B_0) \oplus K_1 \oplus B_1)$ for all $2^{56} - 2^l$ blocks $B_0 \in \mathcal{B}$.

**The third stage.** In this paragraph we estimate the probability $p$ of getting collision between OMAC tags from the second stage and the one of keystream blocks $\Gamma_i[j]$ from the first stage.

More formally, we estimate the probability

$$p = \Pr_K \left[ \{E_K(S)_{S \in \mathcal{S}}\} \cap \{E_K(E_K(B_0) \oplus K_1 \oplus B_1)\}_{B_0 \in \mathcal{B}} \neq \varnothing \right]$$

under the following conditions : $\mathcal{S} \cap \mathcal{B} = \varnothing$, $0^{128} \notin \mathcal{S}$, $0^{128} \notin \mathcal{B}$, $|\mathcal{S}| = 2^l(2^{57} - 1)$, $|\mathcal{B}| = 2^{56} - 2^l$, the key $K_1 = K_1(E_K(0^{128}))$ is a function which depends only on the value $E_K(0^{128})$.

We estimate this probability in the ideal cipher model, where $E_K$ is supposed to be a random permutation on $V_{128}$.

Then, by the technical Lemma 1 from the Appendix section, we obtain

$$p \geqslant 1 - e^{-\frac{(2^{56}-2^l)(2^l(2^{57}-1)-1)}{2^{128}}} = 1 - e^{-2^{l-15}\left(1-\frac{1}{2^{56-l}}\right)\left(1-\frac{1}{2^{57}}-\frac{1}{2^{57+l}}\right)}.$$

This estimation increases monotonically on $6 \leqslant l < 55$, and if $l = 15$ then we have $p > 0.63 \approx 1 - e^{-1}$.

Therefore if we encrypt $2^{15}$ messages on the first stage then one of OMAC values collides with one of the keystream blocks with the probability $p > 0.63$.

**Forging tag.** Suppose that we have collision and $\mathsf{MAC}_K^{(128)}(B) = \Gamma_i[j] = E_K(0^8\|IV_i\|\mathrm{str}_{64}(j))$, where

$$B = \underbrace{F\|\mathrm{str}_8(128)\|IV\|A\|C\|0^{54}}_{B_0} \,\|\, \underbrace{\mathrm{str}_{64}(1)\|\mathrm{str}_{64}(1)}_{B_1}.$$

Consider pairs $(C', A')$ of the ciphertexts $C' = 0^1 \in V_1$ and the associated data $A' = 0\|C\|0^u$ with $u = 0, \ldots, 53$. Note that the OMAC input for such pairs is equal to

$$B' = \underbrace{F\|\mathrm{str}_8(128)\|IV\|A'\|C'\|0^{55-(u+2)}}_{B'_0} \,\|\, \underbrace{\mathrm{str}_{64}(u+2)\|\mathrm{str}_{64}(1)}_{B'_1}.$$

Note that $B'_0 = B_0$ and thus OMAC value $\mathsf{MAC}_K^{(128)}(B')$ is equal to $E_K(E_K(B'_0) \oplus K_1 \oplus B'_1) = E_K(E_K(B_0) \oplus K_1 \oplus B'_1)$

Let us consider the set of strings $\hat{\mathcal{B}} = \{\hat{B} \in V_{128} | \hat{B} = \mathrm{str}_{64}(r)\|\mathrm{str}_{64}(1), r = 2, \ldots, 55$. This set describes all possible values of $B'_1$ for pairs $(C', A')$. Note that for all $\hat{B} \in \hat{\mathcal{B}}$ holds

$$\begin{aligned} E_K(B_0) \oplus K_1 \oplus \hat{B} &= E_K(B_0) \oplus K_1 \oplus B_1 \oplus (B_1 \oplus \hat{B}) = \\ &= 0^8\|IV_i\|\mathrm{str}_{64}(j) \oplus ((\mathrm{str}_{64}(1)\|\mathrm{str}_{64}(1)) \oplus (\mathrm{str}_{64}(r)\|\mathrm{str}_{64}(1))) = \\ &= 0^8\|IV_i\|\mathrm{str}_{64}(j) \oplus ((\mathrm{str}_{64}(1) \oplus \mathrm{str}_{64}(r))\|\mathrm{str}_{64}(0)) = 0^8\|IV_t\|\mathrm{str}_{64}(j), \end{aligned}$$

where $IV_t$ can differ from $IV_i$ only in the 6 least significant bits. Hence $0^8\|IV_t\|\mathrm{str}_{64}(j) \in \mathcal{S}$, and we know the corresponding ciphertext $E_K(0^8\|IV_t\|\mathrm{str}_{64}(j))$ from the first stage.

Therefore we can forge the tag value for the pair $(C', A')$ that corresponds to $\hat{B}$ as follows

$$E_K(E_K(B_0) \oplus K_1 \oplus \hat{B}) = E_K(0^8\|IV_t\|\mathrm{str}_{64}(i)),$$

where $\hat{B} = \mathrm{str}_{64}(r) \| \mathrm{str}_{64}(1)$. Therefore we get 54 forged tags with the probability $p > 0.63$.

### 4.3 Attack complexity

In the current section we estimate the complexity of the described attack.

At the first stage the adversary should form $2^{15}$ queries with length of $2^{57} - 1$ blocks and, hence, store near $2^{72}$ blocks in a sorted list. So the complexity of the first stage is near $72 \cdot 2^{72} \approx 2^{79}$.

The second stage needs processing just one block for $(2^{56} - 2^{15})$ remained messages and find the collision with the values from the stored sorted list. So the complexity of the second stage can be bound by $72 \cdot 2^{56} \approx 2^{63}$.

As it is proven in the previous section the success probability $p$ is greater than 0.63.

Summarizing this section we claim that with the total (time and query) complexity of near $2^{79}$ and probability of $p > 0.63$ the adversary can forge tag for 54 «non-authentic» messages.

### 4.4 Provable security

This attack can be interpreted using provable security ideas. Consider «8 bits» in the Auth model supposing that the used block cipher is a family of all permutations. In this case the adversary complexity can be measured only in terms of the total length of queries, since the resulting «8 bits» mode becomes the information theoretic object which security does not depend on adversary's time complexity (only on the query complexity).

The adversary constructed in the proposed attack makes $2^{56}$ encryption queries of total length of near $2^{72}$ blocks and then one decryption query of length no more than 1 block. So we can estimate the advantage in the Auth model as follows: $\mathsf{Adv}^{\mathrm{Auth}}_{8bits}(\mathcal{A}) > 0.63$, where $\mathcal{A}$ makes the above-mentioned amount of queries of the certain lengths.

The total length of all queries is slightly bigger than the birthday bound, but significantly lower than the trivial random guessing forgery attack complexity. For the original CCM mode the proven bound [5] is near the birthday attack complexity (for the total length of processed data). But there is no known attacks of such complexity on the CCM mode, hence the complexity of real attack has to be at least be equal to the birthday attack complexity .

## 5 Conclusion

In the current paper the near birthday attack for the «8 bits» mode was proposed. The described attack is based on the adversary's possibility to obtain clear tag values and then to compare them with keystream blocks used for encryption. Note that the considered mode can be easily modified to resist this attack with a minor loss in performance. The only modification is to additionally encrypt the tag value (as in the original CCM mode). Thus, under secure blockcipher the adversary will not obtain information about tag values.

## 6 Acknowledgments

We thank Evgeny K. Alekseev for useful discussions and comments during this work.

## References

[1] Bellare M., Rogaway P., Wagner D. The EAX mode of operation // International Workshop on Fast Software Encryption / Springer. — 2004. — P. 389–407.

[2] Competition for Authenticated Encryption: Security, Applicability, and Robustness. — CAESAR. — 2014. — May. — Access mode: `http://competitions.cr.yp.to/caesar.html`.

[3] IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications // IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012). — 2016. — Dec. — P. 1–3534.

[4] «Information technology. Cryptographic protection of information. Block cipher modes of operation» (in Russian) : National standard of the Russian Federation : GOST R 34.13-2015 / STANDARTINFORM : 01.01.2016.

[5] Jonsson, Jakob. On the Security of CTR + CBC-MAC // Selected Areas in Cryptography / Ed. by Nyberg, Kaisa and Heys, Howard. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2003. — P. 76–93.

[6] The Transport Layer Security (TLS) Protocol Version 1.3 : Internet-Draft : draft-ietf-tls-tls13-23 / Internet Engineering Task Force ;

Executor: Eric Rescorla : 2018. — Jan. — 154 p. Access mode: https://datatracker.ietf.org/doc/html/draft-ietf-tls-tls13-23. — Work in Progress.

[7] Rogaway P., Shrimpton T. A provable-security treatment of the key-wrap problem // Annual International Conference on the Theory and Applications of Cryptographic Techniques / Springer. — 2006. — P. 373–390.

[8] Shrimpton T. A characterization of authenticated-encryption as a form of chosen-ciphertext security. — Cryptology ePrint Archive, Report 2004/272. — 2004. — https://eprint.iacr.org/2004/272.

[9] Whiting D., Housley R., Ferguson N. Counter with CBC-MAC (CCM). — RFC 3610. — 2003. — Sep. — Access mode: https://rfc-editor.org/rfc/rfc3610.txt.

## A Appendix

**Lemma 1.** *Let $x_1, \ldots, x_s$ and $y_1, \ldots, y_t$ be different non-zero vectors from $V_n$; let $\beta$ be an arbitrary vector from $V_n$; let the set of all permutations $Perm(V_n)$ on $V_n$. Consider a uniform distribution over $Perm(V_n)$. Then the following estimation holds*

$$\Pr_{P\in Perm(V_n)}\left[\{P(x_m)\}_{m=1}^{s}\cap\{P(P(y_k)\oplus K_1(P(0^n))\oplus\beta))\}_{k=1}^{t}\neq\varnothing\right]\geqslant 1-e^{-\frac{t(s-1)}{2^n}}.$$

*Proof.* Let us estimate the probability of the opposite event

$$\begin{aligned}\Pr_{P\in Perm(V_n)}\left[\{P(x_m)\}_{m=1}^{s}\cap\{P(P(y_k)\oplus K_1(P(0^n))\oplus\beta))\}_{k=1}^{t}=\varnothing\right]=\\ =\Pr_{P\in Perm(V_n)}\left[\{x_m\}_{m=1}^{s}\cap\{P(y_k)\oplus K_1(P(0^n))\oplus\beta)\}_{k=1}^{t}=\varnothing\right]=\\ =\Pr_{P\in Perm(V_n)}\left[\{x_m\oplus K_1(P(0^n))\oplus\beta\}_{m=1}^{s}\cap\{P(y_k)\}_{k=1}^{t}=\varnothing\right].\end{aligned}$$

Since the set $Perm(V_n)$ have a uniform distribution, we can calculate this probability by combinatorial methods:

$$\begin{aligned}\Pr_{P\in Perm(V_n)}\left[\{x_m\oplus K_1(P(0^n))\oplus\beta\}_{m=1}^{s}\cap\{P(y_k)\}_{k=1}^{t}=\varnothing\right]=\\ =\frac{\#\{P\in Perm(V_n)|\ \{x_m\oplus K_1(P(0^n))\oplus\beta\}_{m=1}^{s}\cap\{P(y_k)\}_{k=1}^{t}=\varnothing\}}{|Perm(V_n)|}=\\ =\frac{1}{2^n!}\sum_{\alpha\in V_n}\#\{P\in Perm(V_n)|\ P(0^n)=\alpha\\ \text{and }\{x_m\oplus K_1(\alpha)\oplus\beta\}_{m=1}^{s}\cap\{P(y_k)\}_{k=1}^{t}=\varnothing\}.\quad(2)\end{aligned}$$

Note that given a fixed $\alpha$ the permutations to be enumerated under the sum sign have the following properties: 1) the value of a permutation on zero vector $0^n$ is equal to $\alpha$; 2) the values of a permutation on the vectors $y_1,\ldots,y_t$ may be arbitrary not including in the $s$-element set $S_\alpha = \{x_m \oplus K_1(\alpha) \oplus \beta\}_{m=1}^{s}$. There are two variants $\alpha \in S_\alpha$ and $\alpha \notin S_\alpha$. Depending on the variant the number of these permutations may be slightly different:

$$\#\{P \in Perm(V_n) |\ P(0^n) = \alpha \text{ and } S_\alpha \cap \{P(y_k)\}_{k=1}^{t} = \varnothing\} = \\ = \begin{cases} (2^n - s) \cdot (2^n - s - 1) \cdot \ldots \cdot (2^n - s - (t-1)) \cdot (2^n - (t+1))!, & \text{if } \alpha \in S_\alpha; \\ (2^n - s - 1) \cdot (2^n - s - 2) \cdot \ldots \cdot (2^n - s - t) \cdot (2^n - (t+1))!, & \text{if } \alpha \notin S_\alpha. \end{cases} \quad (3)$$

We need an upper bound of the opposite event probability. So, taking into account formulas (2) and (3), we have the following inequalities:

$$\Pr_{P \in Perm(V_n)} \left[\{x_m \oplus K_1(P(0^n)) \oplus \beta\}_{m=1}^{s} \cap \{P(y_k)\}_{k=1}^{t} = \varnothing\right] \leqslant \\ \leqslant 2^n \cdot \frac{(2^n - s) \cdot (2^n - s - 1) \cdot \ldots \cdot (2^n - s - (t-1)) \cdot (2^n - (t+1))!}{2^n!} = \\ = \frac{2^n - s}{2^n - 1} \cdot \frac{2^n - s - 1}{2^n - 2} \cdot \ldots \cdot \frac{2^n - s - (t-1)}{2^n - t} \leqslant \left(\frac{2^n - (s-1)}{2^n}\right)^t = \\ = \left(1 - \frac{s-1}{2^n}\right)^t = e^{t \ln\left(1 - \frac{s-1}{2^n}\right)} \leqslant e^{-\frac{t(s-1)}{2^n}}.$$

This estimation of the opposite event probability proves the lemma. $\square$

# Within a Friend Zone: How Far Can We Proceed with Data Encryption not Getting Out

Ivan Lavrikov and Vasily Shishkin

Technical Committee for Standardization
«Cryptography and security mechanisms» (TC 26), Moscow, Russia
{lavrikov_iv, shishkin_va}@tc26.ru

**Abstract**

Usage of block cipher modes of operation is the main way to achieve different important properties for the present security techniques in the secret-key cryptography. The amount of data to be processed with some mode of operation without change of key is the crucial characteristic of many informational systems and protocols. In this paper we investigate two different approaches of estimation of the stated characteristic – direct cryptographic analysis and provable security – and provide grounded margins that could be used during the synthesis of different systems and providing the terms of use for such systems.

**Keywords:** block cipher, mode of operation, provable security.

## 1 Introduction

Custom approach in design and analysis of block cipher modes of operation is studying the properties of both mechanisms independently. The core of this approach is «any *secure* block cipher in any *secure* mode of operation results in *secure* cryptosystem». Thus, during the analysis of mode of operation structural properties of a block cipher are out of scope and it is assumed that block cipher behaves as a random permutation.

In this note we investigate important characteristic of block cipher mode of operation – maximum amount of data that could be processed without key change.

First we fix the maximum acceptable value for success probability of obtaining additional information about unknown part of plaintext. The upper bound for the stated amount of data can be estimated in two different ways. First approach is based on construction of particular attacks on a mode of operation targeting additional information about unknown part of the plaintext.

The maximum amount of data to be processed without key change should be upper bounded in such a way that all known attacks targeting information about unknown plaintexts have probability of success less than the previously fixed value.

The second approach is so called «provable security». In this note we show that bounds obtained by both approaches are very close. For the most widespread modes of operation we obtain formulas determining exact amount of data which can be processed without key change. Values obtained by these formulas were used in recommendations [2] of Russian National Standardization Organization GOST R.

## 2 Terms and definitions, known results

Let $E_K : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$ be the block cipher transformation of the plaintext block $P \in \{0,1\}^n$ with the use of the key $K \in \{0,1\}^k$ to the ciphertext block $C \in \{0,1\}^n$: $E(P,K) = C$. We will also use the short variant for the stated notation for trasformation $E$, plaintext block $P$, ciphertext block $C$ and key $K$ as follows: $E_K(P) = E(P,K) = C$. Let $D(C,K) = E^{-1}(C,K)$ and $D_K(C) = E_K^{-1}(C)$ for all $C \in \{0,1\}^n$, $K \in \{0,1\}^k$.

In the following we assume that the key for the block cipher is chosen uniformly at random and the transformation $E_K$ with a uniform random key is indistinguishable from a random permutation.

There are a lot of papers devoted to the study of block cipher modes of operation properties (see, for example, [6, 7, 8, 9, 10, 11, 12, 14, 15, 16]). Almost all of the stated articles are devoted to the «provable security» approach. Bounds for the amount of data to be processed based on the provable security results could be considered only as lower values for the upper bound of the acceptable amount of data to be processed. Upper values for the stated bound could be found only by the means of the combinatorial-algorithmic approach by the presentation of concrete cryptanalytic attacks.

Note that in case when stated lower values are close to stated upper values, the lower values become a reasonable estimations for the maximum acceptable amount of data to be processed by block cipher mode of operation without key change.

## 2.1 Descriptions of modes of operation

### 2.1.1 Electronic Code Book mode of operation, ECB

A plaintext $P \in V^*$, $|P| = n \cdot t$, is divided into blocks of length $n$: $P = P_1 \| P_2 \| \ldots \| P_t$. Ciphertext blocks are calculates with the following equation:

$$C_i = E_K(P_i),\ i = \overline{1,t}.$$

The length of data to be encrypted in ECB mode must be a multiple of block cipher's block length $n$, so in some cases original data must be padded with the use of some padding procedure.

Illustration of encryption process with ECB mode is given in fig. 1.

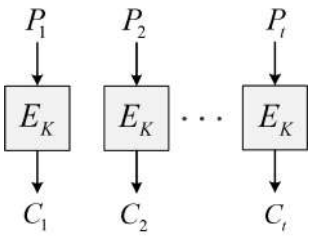



Figure 1: Encryption in ECB mode

### 2.1.2 Counter mode of operation, CTR

The parameter of CTR mode is the value $0 < s \leq n$. For encryption (decryption) of every plaintext with the use of one key the unique initialization value $IV \in \{0,1\}^l$ for some fixed $l \in \mathbb{N}, l \geq n$ is used. Encryption in CTR mode is based on XOR operation of plaintext with the sequence that is output of encryption of the counter sequence $S_i \in \{0,1\}^n$, $i = 1, 2, \ldots$ by the block cipher in ECB mode with truncation of every output block to the length $s$. The initial value of counter is achieved from the initialization value: $S_1 = \mathcal{I}_n(IV)$, where $\mathcal{I}_n$ is some fixed function $\mathcal{I}_n : \{0,1\}^l \to \{0,1\}^n$. Subsequent counter values are generated by the means of some function $A : \{0,1\}^n \to \{0,1\}^n$ in the following way $S_{i+1} = A(S_i)$, $i = 2, 3, \ldots$.

A plaintext $P \in V^*$ is divided into blocks of length $s$ (with the exception of the last one), $P = P_1 \| P_2 \| \ldots \| P_t$, $P_i \in \{0,1\}^s$, $i = \overline{1, t-1}$, $P_t \in \{0,1\}^r$, $r \leq s$. Ciphertext blocks are calculated as

$$\begin{cases} C_i = P_i \oplus \mathcal{T}_s(E_K(S_i)), & i = \overline{1, t-1}. \\ C_t = P_t \oplus \mathcal{T}_r(E_K(S_t)), \end{cases}$$

Illustration of encryption process in CTR mode is given in fig. 2.

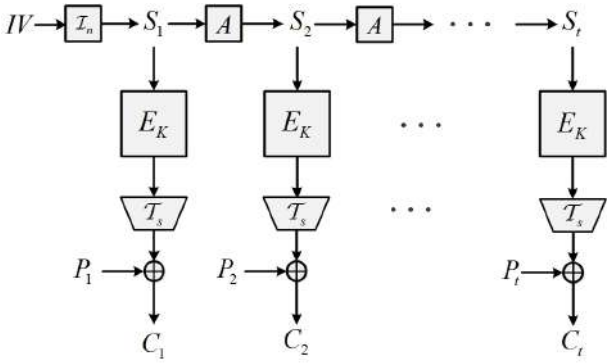



Figure 2: Encryption in CTR mode

Note that function $A$ and initialization values must be chosen in such a way that all counter values $S_i$, $i = 1, 2, \ldots$ used in encryption process without key change are pairwise distinct.

### 2.1.3 Output Feedback Mode of operation

OFB mode of operation has following parameters: $s$, $m$ and $q$ such that $0 < s \leq n$, $m = n \cdot q$, $q \geq 1$, where $q$ is integer.

During the encryption process without key change for each separate plaintext unique or pseudorandom value of initialization value $IV \in \{0,1\}^l$ for some fixed $l \in \mathbb{N}, l \geq n$ and binary feedback shift register $R$ of length $m$ are used. The initial value of $R$ is the value $\mathcal{I}_m(IV)$, where $\mathcal{I}_m$ is some fixed function $\mathcal{I}_m : \{0,1\}^l \to \{0,1\}^m$. Encryption in OFB mode is based on XOR operation of plaintext blocks with sequence that is obtained by concatenation of blocks of length $s$. To get a regular block of the sequence most significant $n$ bits of register $R$ are encrypted with the use of a block cipher in ECB mode and the result is truncated to the length $s$. Then the register $R$ is shifted to $n$ positions in the direction of the most significant bits and vacated cells are filled with previously obtained result of block cipher application.

A plaintext $P \in V^*$ is divided into blocks of length $s$ (with the exception of the last one), $P = P_1 \| P_2 \| \ldots \| P_t$, $P_i \in \{0,1\}^s$, $i = \overline{1, t-1}$, $P_t \in \{0,1\}^r$, $r \leq s$. Ciphertext blocks are calculates as follows

$$\begin{cases} R_1 = IV, & \\ Y_i = E_K(\mathrm{msb}n(R_i)), & i = \overline{1,t}, \\ R_i = \mathrm{lsb}m - n(R_{i-1}) \| Y_i, & i = \overline{2,t}. \\ C_i = P_i \oplus \mathcal{T}_s(Y_i), & i = \overline{1,t-1}, \\ C_t = P_t \oplus \mathcal{T}_r(Y_t). & \end{cases}$$

Illustration of encryption process in OFB mode is given in fig. 3.

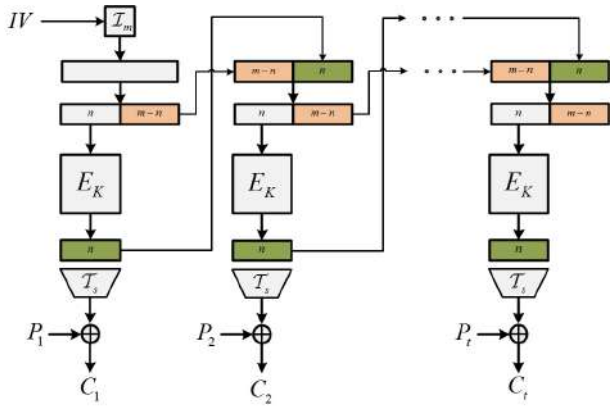



Figure 3: Encryption in OFB mode

### 2.1.4 Cipher Block Chaining mode of operation

CBC mode of operation has parameter $m$, $m = n \cdot q$, for some integer $q \geq 1$. During the encryption process binary shift register $R$ of length $m$ is used. For encryption of each particular plaintext with the use of one key pseudorandom initialisation value $IV \in \{0,1\}^l$ for some fixed $l \in \mathbb{N}, l \geq n$ is used. In CBC mode regular block of ciphertext is calculated by encryption in ECB mode of XOR of regular plaintext block and $n$ most significant bits of register $R$. After that register $R$ is shifted by $n$ positions in the direction of the most significant bits and the vacated cells are filled with obtained ciphertext value. The initial value of register $R$ is obtained as follows: $\mathcal{I}_m(IV)$, where $\mathcal{I}_m$ is some fixed function, $\mathcal{I}_m : \{0,1\}^l \to \{0,1\}^m$.

A plaintext $P \in V^*$ is divided into blocks of length $n$: $P = P_1 \| P_2 \| \ldots \| P_t$, $P_i \in \{0,1\}^n$ (in some cases original data must be padded with the use of some padding procedure). Ciphertext blocks are calculated as follows

$$\begin{cases} R_1 = \mathcal{I}_m(IV), \\ C_i = E_K(P_i \oplus \mathrm{msb}n(R_i)), & i = \overline{1,t}, \\ R_i = \mathrm{lsb}m - n(R_{i-1}) \| C_{i-1}, & i = \overline{2,t}. \end{cases}$$

Illustration of encryption process in CBC mode is given in fig. 4.

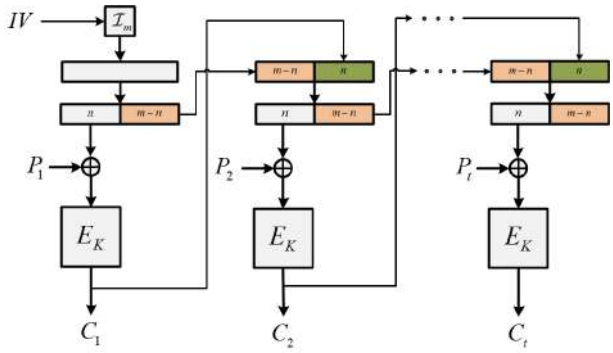



Figure 4: Encryption in CBC mode

### 2.1.5 Cipher Feedback mode of operation

CFB mode of operation has parameters $s$ and $m$, $0 < s \leq n$, $n \leq m$. During the encryption process of each plaintext with the use of one key binary shift register $R$ of length $m$ and pseudorandom initialisation value $IV \in \{0,1\}^l$ for some fixed $l \in \mathbb{N}, l \geq n$ are used. The initial value of register $R$ is obtained with use of a function $\mathcal{I}_m : \{0,1\}^l \to \{0,1\}^m$ and is set to $\mathcal{I}_m(IV)$.

Encryption in CFB mode is based on XOR operation of plaintext blocks with cipher sequence that is obtained by truncation of the result of encryption in ECB mode of $n$ most significant bits of register $R$ to $s$ bits . Then register $R$ is shifted by $s$ positions in the direction of the most significant bits and vacated cells are filled with obtained ciphertext block.

A plaintext $P \in V^*$ is divided into blocks of length $s$ (with the exception of the last one), $P = P_1 \| P_2 \| \ldots \| P_t$, $P_i \in \{0,1\}^s$, $i = \overline{1, t-1}$, $P_t \in \{0,1\}^r$, $r \leq s$. Ciphertext blocks are calculated as follows

$$\begin{cases} R_1 = \mathcal{I}_m(IV), \\ C_i = P_i \oplus \mathcal{T}_s(E_K(\text{msb}n(R_i))), & i = \overline{1, t-1}, \\ R_i = \text{lsb}m - s(R_{i-1}) \| C_{i-1}, & i = \overline{2, t}, \\ C_t = P_t \oplus \mathcal{T}_r(E_K(\text{msb}n(R_t))). \end{cases}$$

Illustration of encryption process in CFB mode is given in fig. 5.

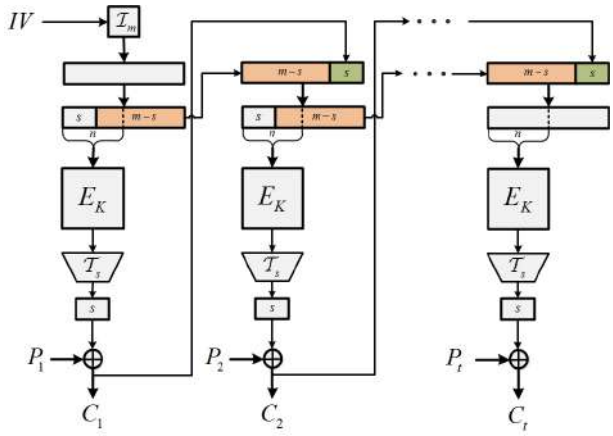



Figure 5: Encryption in CFB mode

## 2.2 Approaches to the problem solving

Let's assume that $N = u + v$ plaintext blocks were processed without key change. Denote these plaintext blocks $P_1, P_2, \ldots, P_u, P'_1, P'_2, \ldots, P'_v$, $P_i, P'_j \in \{0,1\}^n$, $i = 1, 2, \ldots, u$, $j = 1, 2, \ldots, v$, and corresponding ciphertext blocks – $C_1, C_2, \ldots, C_u, C'_1, C'_2, \ldots, C'_v$. In the following we will consider blocks $P_1, P_2, \ldots, P_u$ to be known and blocks $P'_1, P'_2, \ldots, P'_v$ to be unknown.

Let $\pi$ be the fixed maximum acceptable value for probability of obtaining additional information about unknown part of the plaintext. The maximum acceptable amount of data to be processed we will calculate in blocks and will denote this value as $N_{max}$. The main task is to find functional dependency between $N_{max}$ and $\pi$.

From the «provable security» point of view cryptographic properties of many block cipher modes of operation are already defined. Let's state some of the known results in this area. Following commonly used notations, let $\mathcal{A}$ be some distinguisher for some mode of operation. Assume that stated distinguisher makes at most $q$ queries of the total length $\mu$ to the some oracle. Advantage of the distinguisher $\mathcal{A}$ in solving some decisional task $\mathcal{T}$ for some mode of operation $\mathcal{E}$ we will denote by $\Delta^{\mathcal{T}}_{\mathcal{E}}(\mathcal{A})$ and the maximum of this value for all distinguishers with given amount of calculations (time of work) $t$, number of queries $q$ of total length $\mu$ to some oracle we will denote $\Delta^{\mathcal{T}}_{\mathcal{E}}(t, q, \mu)$. Formal definitions for the terms used throughout the article can be found, for example, in [15]

In [7] some properties of chaining modes of operation are proven. Margins for the CBC mode of operation in the «left-or-right» notation can be deduced from results of this article.

It's worth mentioning that cryptanalytic techniques based on the *birthday paradox* are still usable for the block cipher modes of operation analysis even

in case when instead of block cipher arbitrary random permutation is used. In particular, the next proposition holds.

**Proposition 1.** *[7, Proposition 15] There is left-or-right distinguisher $\mathcal{A}$ for CBC mode with arbitrary random function, that makes at most $q$ queries of the total length at most $\mu$ bits ($\mu \leq l \cdot 2^{\frac{n}{2}}$), such that*

$$\Delta^{lr}_{CBC}(\mathcal{A}) \geq 0,316 \cdot \left(1 - 2 \cdot 2^{-\frac{n}{2}}\right) \cdot \left(\frac{\mu^2}{n^2} - \frac{\mu}{n}\right) \cdot \frac{1}{2^n}.$$

The next lemma indicates that in the stated case techniques based on the birthday paradox are the best possible ones.

**Lemma 1.** *[7, Lemma 16]*

*Let $\mathcal{A}$ be some left-or-right distinguisher that distinguish CBC mode with arbitrary random function from arbitrary random transformation (with the same domain and range) that makes at most $q$ queries of the total length at most $\mu$ bits. Then*

$$\Delta^{lr}_{CBC}(\mathcal{A}) \leq \left(\frac{\mu^2}{n^2} - \frac{\mu}{n}\right) \cdot \frac{1}{2^n}.$$

Stated proposition and lemma could be used to prove the next

**Theorem 1.** *[7, Theorem 17]*

*There is a constant $c > 0$ such that if $\mathcal{F}$ is the family of functions from $\{0,1\}^n$ to $\{0,1\}^L$ with $\Delta^{prf}_{\mathcal{F}}(q', t', \mu') \leq \varepsilon'$. Then for every $q$*

$$\Delta^{lr}_{CBC_{\mathcal{F}}}(t, q, \mu) \leq \varepsilon,$$

*where $\mu = q'n$, $t = t' - c\mu$, $\varepsilon = 2\varepsilon' + \left(\frac{\mu^2}{n^2} - \frac{\mu}{n}\right) \cdot \frac{1}{2^n}$.*

**Remark 1.** *In [15] it is stated that similar technique could be used for studying properties of OFB mode.*

Let's state some properties of CFB mode of operation based on [5]. The best margin in this case is again based on birthday paradox. First we provide the result of [5] stating that there doesn't exist any distinguisher better than one based on birthday paradox.

**Lemma 2.** *[5, Lemma 1]*

*For any left-or-right distinguisher $\mathcal{A}$ for CFB mode with arbitrary random function that makes $q$ queries of total length $qL$ bits*

$$\Delta^{lr}_{CFB}(\mathcal{A}) \leq \frac{q(q-1)}{2^{n+1}}.$$

**Theorem 2.** *[5, Theorem 1]*

*If $\mathcal{F}$ – a family of functions from $\{0,1\}^n$ to $\{0,1\}^L$ such that $\Delta^{prf}_{\mathcal{F}}(q',t',\mu') \leq \varepsilon'$, then $\Delta^{lr}_{CFB_{\mathcal{F}}}(t,q,\mu) \leq \varepsilon$, where $q = q'$, $\mu = q'L$, $t = t' - q - c$ (where $c$ – some small-valued constant), $\varepsilon = 2\varepsilon' + \frac{q(q-1)}{2^{n+1}}$.*

Let's denote CTR mode with random choice of initialization value as CTR\$ and nonce-based CTR mode – as CTRC.

**Lemma 3.** *[7, Lemma 12]*

*Let $\mathcal{F}$ be a family of functions from $\{0,1\}^n$ to $\{0,1\}^L$. Then for any non-negative integers $q, \mu$ and $t$, where binary order of $t$ is comparable with $q$,*

$$\Delta^{lr}_{CTR\$_{\mathcal{F}}}(t,q,\mu) \leq \frac{\mu(q-1)}{L \cdot 2^n}$$

*and for $\mu > L \cdot 2^n$*

$$\Delta^{lr}_{CTRC_{\mathcal{F}}}(t,q,\mu) = 0.$$

**Theorem 3.** *[7, Theorem 13]*

*Let $\mathcal{F}$ be a family of functions from $\{0,1\}^n$ to $\{0,1\}^L$.*

- *There is a constant $c > 0$ such that if $\Delta^{prf}_{\mathcal{F}}(t',q',\mu') < \varepsilon'$, then for every non-negative integer $q$*

$$\Delta^{lr}_{CTR\$_{\mathcal{F}}}(t,q,\mu) < 2 \cdot \varepsilon' + \frac{\mu(q-1)}{L \cdot 2^n},$$

  *where $\mu = q'L$, $t = t' - c \cdot \frac{\mu}{L}(n+L)$.*

- *There is a constant $c > 0$ such that if $\Delta^{prf}_{\mathcal{F}}(t',q',\mu') < \varepsilon'$, then for every non-negative integer $q$*

$$\Delta^{lr}_{CTRC_{\mathcal{F}}}(t,q,\mu) < 2 \cdot \varepsilon',$$

  *where $\mu = \min\{q'L, L2^n\}$, $t = t' - c \cdot \frac{\mu}{L}(n+L)$.*

Let's note that all stated margins for block cipher modes of operation are based on properties of pseudorandom functions, but pseudorandom permutations are more suitable for studying block cipher's properties. In [7] the following result is proven

**Proposition 2.** *[7, Proposition 8]*

*Let $\mathcal{F} = \{E_K : \{0,1\}^n \to \{0,1\}^n | K \in \mathcal{K}\}$ be some family of permutations. Then $|\Delta^{prf}_{\mathcal{F}}(t,q,\mu) - \Delta^{prp-cpa}_{\mathcal{F}}(t,q,\mu)| < \frac{q^2}{2^{n+1}}$.*

**Remark 2.** *In particular, from this proposition it follows that* $\Delta^{prf}_{\mathcal{F}}(t,q,\mu) < \Delta^{prp-cpa}_{\mathcal{F}}(t,q,\mu) + \frac{q^2}{2^{n+1}}$.

Taking into account Remark 2, Theorem 1, Theorem 2 and Theorem 3 the next proposition could be stated

**Proposition 3.** *Let* $\mathcal{F} = \{E_K : \{0,1\}^n \to \{0,1\}^n | K \in \mathcal{K}\}$ *be some family of permutations such that* $\Delta^{prp-cpa}_{\mathcal{F}}(t',q',\mu') \le \varepsilon'$. *Then*

– *there is a constant* $c > 0$ *such that for any* $q$

$$\Delta^{lr}_{CBC_{\mathcal{F}}}(t,q,\mu) \le 2\varepsilon' + \frac{q^2}{2^n} + \left(\frac{\mu^2}{n^2} - \frac{\mu}{n}\right)\cdot\frac{1}{2^n},$$

*where* $\mu = q'n$, $t = t' - c\mu$;

– *there is a constant* $c > 0$ *such that*

$$\Delta^{lr}_{CFB_{\mathcal{F}}}(t,q,\mu) \le 2\varepsilon' + \frac{q^2}{2^n} + \frac{q(q-1)}{2^{n+1}},$$

*where* $q = q'$, $\mu = q'n$, $t = t' - q - c$;

– *there is a constant* $c > 0$ *such that for any* $q$

$$\Delta^{lr}_{CTR\$_{\mathcal{F}}}(t,q,\mu) \le 2\varepsilon' + \frac{q^2}{2^n} + \frac{\mu(q-1)}{L\cdot 2^n},$$

*where* $\mu = q'n$, $t = t' - c\cdot 2\mu$;

– *there is a constant* $c > 0$ *such that for any* $q$

$$\Delta^{lr}_{CTRC_{\mathcal{F}}}(t,q,\mu) \le 2\varepsilon' + \frac{q^2}{2^n},$$

*where* $\mu = \min\{q'n, n2^n\}$, $t = t' - c\cdot 2\mu$.

Modern block ciphers are developed to achieve broad variety of special properties. Indistinguishability of permutations family (corresponding to the block cipher) from the family of arbitrary random permutations seems to be very important property. In the opposite case existence of effective methods for distinguishing stated families of permutations could witness the existence of structural weaknesses in block cipher. Thus in the next sections we will assume that modes of operations are used with a secure block cipher, i.e. $\varepsilon' = 0$.

## 3 Some properties of the modes of operations

### 3.1 ECB

For any $P, P' \in \{0,1\}^n$ $E_K(P) = E_K(P')$ iff $P = P'$. Thus, given two ciphertext blocks $C$ and $C'$ we get some information about unknown plaintext blocks $P$ and $P'$. Then for ECB mode of operation $N_{max} = 1$ independently of $\pi$.

In some articles and even national standards ECB mode is offered to encryption of cipher keys. Let's assume that stated key has a length of $r > 1$ blocks. Then after the encrypting of the key in ECB mode we will get $r$ ciphertext blocks $C_1, C_2, \ldots, C_r$. In case $r$ is small, all the stated blocks will be different with high probability (about 1). But in this case we can get to the conclusion that all blocks of the initial key was different too. Thus, complexity of brutforce attack will decrease from the value of order $2^{rn}$ to the value of order $2^n(2^n - 1)\ldots(2^n - r + 1)$.

### 3.2 CTR

Let's assume that $s = n$ and unknown plaintext consists of same block $P'$ repetition.

Since bijection of block cipher, all values $P_i \oplus C_i$ and $P' \oplus C'_j$, $i \in \overline{1,u}$, $j \in \overline{1,v}$ are pairwise distinct. Then unknown plaintext block can't belong to the set $M = \{P_i \oplus C_i \oplus C'_j \mid i \in \overline{1,u},\ j \in \overline{1,v}\}$. The average cardinality of the set $M$ can be estimated as follows (for more information, see, for example, [1])

$$|M| = 2^n - 2^n\left(1 - \frac{1}{2^n}\right)^{uv} \cong 2^n\left(1 - e^{-\frac{uv}{2^n}}\right),$$

consequently, the value of $P'$ could be chosen from $2^n e^{-\frac{uv}{2^n}}$ values. Thus, $\pi \geq \frac{1}{2^n} e^{\frac{uv}{2^n}}$, from which it follows that $uv \leq 2^n \ln(\pi 2^n)$.

Since $u + v = N$, $uv$ is maximum iff $u = v = \frac{N}{2}$. Thus

$$N_{max} \leq 2^{\frac{n}{2}+1}\sqrt{\ln(\pi 2^n)}.$$

To get the margins for $N_{max}$ with different values of $s$ additional research is required.

From the provable security point of view, for the CTR$ mode there is a

constant $c > 0$ such that for any $q$

$$\Delta^{lr}_{\mathrm{CTR\$}_{\mathcal{F}}}(t, q, \mu) \leq \frac{q^2}{2^n} + \frac{\mu(q-1)}{L \cdot 2^n},$$

where $\mu = q'n$, $t = t' - c \cdot 2\mu$.

Let $q = q'$, then if $\Delta^{lr}_{\mathrm{CTR\$}_{\mathcal{F}}}(t, q, \mu) = \varepsilon$, we get $\varepsilon \leq \frac{q^2}{2^n} + \frac{\mu(q-1)}{n \cdot 2^n}$. The last inequality holds if $q \geq \frac{1+\sqrt{1+4 \cdot 2 \cdot 2^n \cdot \varepsilon}}{4} \sim 2^{\frac{n-1}{2}} \cdot \sqrt{\varepsilon}$.

Thus, if we encrypt no more than $2^{\frac{n-1}{2}}$ plaintext blocks, the advantage of any distinguisher can be bounded by value $\varepsilon$.

For CTRC mode there is a constant $c > 0$ such that for any $q$

$$\Delta^{lr}_{\mathrm{CTRC}_{\mathcal{F}}}(t, q, \mu) \leq \frac{q^2}{2^n},$$

where $\mu = \min\{q'n, n2^n\}$, $t = t' - c \cdot 2\mu$.

Then the inequality $\Delta^{lr}_{\mathrm{CTRC}_{\mathcal{F}}}(t, q, \mu) = \varepsilon$ holds if $q \geq 2^{\frac{n}{2}} \cdot \sqrt{\varepsilon}$.

Thus, if we encrypt no more than $2^{\frac{n}{2}}$ plaintext blocks, the advantage of any distinguisher can be bounded by value $\varepsilon$.

### 3.3 OFB

Let's assume that $s = n$ and $q = 1$. In this case

$$P_i = E^i_K(a) \oplus C_i,$$
$$P'_j = E^{j+u}_K(a) \oplus C'_j,$$

where $a \in \{0,1\}^n$ is some fixed binary string. If $a$ belongs to the cycle of permutation $E_K$ of length less or equal to $N$, then there are $i \in \overline{1,u}$, $j \in \overline{1,v}$ such that $E^i_K(a) = E^{j+u}_K(a)$ and

$$P'_j = P_i \oplus C_i \oplus C'_j.$$

Let's estimate the probability $p$ of the event that for the random permutation given on the set $\{0,1\}^n$ a randomly chosen value from $\{0,1\}^n$ belongs to the cycle of length less or equal to $N$. Note that

$$p = 1 - \frac{2^n - 1}{2^n} \cdot \frac{2^n - 2}{2^n} \cdot \ldots \cdot \frac{2^n - N + 1}{2^n} = 1 - \frac{(2^n)!}{2^{nN}(2^n - N)!}.$$

By Stirling's formula, taking into account $\left(1 + \frac{1}{x}\right)^x \cong e$ we get

$$p \cong 1 - e^{-\frac{N^2}{2^n} + \frac{N}{2^{n+1}}} \cong 1 - e^{-\frac{N^2}{2^n}}.$$

Since $p \leq \pi$, for the OFB mode we get

$$N_{max} \leq 2^{\frac{n}{2}} \sqrt{\ln\left(\frac{1}{1-\pi}\right)}.$$

To get the margins for $N_{max}$ with different values of $s$ and $q$ additional research is required.

From the provable security point of view for the OFB mode of operations security margins are similar to the CBC mode's margins.

### 3.4 CBC

Let's assume $q = 1$. Let $A$ denotes the event that there are $i \in \overline{1,u}$ and $j \in \overline{1,v}$ such that corresponding ciphertext blocks are equal, $C_i = C'_j$. Since $P_i = D_K(C_i) \oplus C_{i-1}$, $P'_j = D_K(C'_j) \oplus C'_{j-1}$ and due to bijectiveness of $D_K$, we get

$$P'_j = P_i \oplus C_{i-1} \oplus C'_{j-1},$$

that let us identify unknown plaintext block $P'_j$. Thus, the probability of $A$ must satisfy the following inequality

$$\mathcal{P}\{A\} \leq \pi. \tag{1}$$

Let's estimate the probability of $A$ (for more information, see, for example, [13]).

$$\mathcal{P}\{A\} = 1 - \left(1 - \frac{1}{2^n}\right)^{uv} \cong 1 - e^{-\frac{uv}{2^n}}. \tag{2}$$

From (1) and (2) we get inequality $uv \leq 2^n \ln\left(\frac{1}{1-\pi}\right)$.

Since $u + v = N$ and $uv$ is maximum iff $u = v = \frac{N}{2}$,

$$N_{max} \leq 2^{\frac{n}{2}+1} \sqrt{\ln\left(\frac{1}{1-\pi}\right)}.$$

To get the margins for $N_{max}$ with different values of $q$ additional research is required.

Let's state CBC mode properties from the provable security point of view. Taking into account Proposition 3 and assumption about blockcipher's properties, we get that there is a constant $c > 0$ such that for any $q$

$$\Delta^{lr}_{\mathrm{CBC}_{\mathcal{F}}}(t, q, \mu) \leq \frac{q^2}{2^n} + \left(\frac{\mu^2}{n^2} + \frac{\mu}{n}\right) \cdot \frac{1}{2^n},$$

where $\mu = q'n$, $t = t' - c\mu$. Let $q = q'$ then with $\mu = qn$ we get

$$\Delta^{lr}_{\mathrm{CBC}_{\mathcal{F}}}(t, q, \mu) \leq \frac{q^2}{2^n} + \left(q^2 + q\right) \cdot \frac{1}{2^n},$$

consequently $\Delta^{lr}_{\mathrm{CBC}_{\mathcal{F}}}(t, q, \mu) \leq \frac{1}{2^{n-1}} \cdot q^2 + \frac{1}{2^n} \cdot q$.

Let's denote $\Delta^{lr}_{\mathrm{CBC}_{\mathcal{F}}}(t, q, \mu) = \varepsilon$. Then $\varepsilon \leq \frac{1}{2^{n-1}} \cdot q^2 + \frac{1}{2^n} \cdot q$. The last inequality holds in case

$$q \geq \frac{\sqrt{1 + 2^{n+3} \cdot \varepsilon} - 1}{4} \sim \sqrt{\varepsilon} \cdot 2^{\frac{n-1}{2}}.$$

Thus, if we encrypt no more than $2^{\frac{n-1}{2}}$ plaintext blocks, advantage of any distinguisher could be bounded by value $\varepsilon$.

### 3.5 CFB

Let's assume $s = m = n$. Let $B$ denotes the event that there are some $i \in \overline{2, u+1}$ and $j \in \overline{2, v+1}$ such that ciphertext blocks $C_{i-1}$ and $C'_{j-1}$ are equal. Since $P_i = E_K(C_{i-1}) \oplus C_i$, $P'_j = E_K(C'_{j-1}) \oplus C'_j$ and due to bijectiveness of $E_K$ we get

$$P'_j = P_i \oplus C_i \oplus C'_j,$$

that let us find the value of unknown plaintext block $P'_j$. Thus the probability of $B$ must satisfy the inequality $\mathcal{P}\{B\} \leq \pi$. By the same reasoning as in section 3.4 we get $N_{max} \leq 2^{\frac{n}{2}+1}\sqrt{\ln\left(\frac{1}{1-\pi}\right)}$.

To get the margins for $N_{max}$ with different values of $s$ and $m$ additional research is required.

Let's state CFB mode properties from the provable security point of view. For this mode of operation there is a constant $c > 0$ such that

$$\Delta^{lr}_{\mathrm{CFB}_{\mathcal{F}}}(t, q, \mu) \leq \frac{q^2}{2^n} + \frac{q(q-1)}{2^{n+1}},$$

where $q = q'$, $\mu = q'n$, $t = t' - q - c$.

Given $\mu = qn$, denoting $\Delta^{lr}_{\mathrm{CFB}_{\mathcal{F}}}(t, q, \mu) = \varepsilon$, we get $\varepsilon \leq \frac{q^2}{2^n} + \frac{q(q-1)}{2^{n+1}}$.

The last inequality holds in case

$$q \geq \frac{1 + \sqrt{1 + 3 \cdot 2^{n+3} \cdot \varepsilon}}{6} \sim \sqrt{\varepsilon} \cdot \frac{1}{\sqrt{3}} 2^{\frac{n+1}{2}}.$$

Thus, if we encrypt no more than $\frac{1}{\sqrt{3}} 2^{\frac{n+1}{2}}$ plaintext blocks, the advantage

of any distinguisher could be confined with value $\varepsilon$.

## 4 Conclusion

For the most commonly used in practice block cipher modes of operation upper bounds for the value of maximum acceptable amount of blocks to be processed without key change are given. Summary is given in Table 4. The value in column «Margin 1» corresponds to the value given by direct cryptanalysis (for OFB and CBC modes of operations amount of blocks in one message to be processed is given). The value in column «Margin 2» corresponds to the value given by provable security approach (given margin states that if we encrypt no more than stated number of plaintext blocks, the advantage of any distinguisher could be confined with value $\varepsilon$).

| Mode of operation | Margin 1 | Margin 2 |
|---|---|---|
| ECB | 1 | — |
| CTR | $2^{\frac{n}{2}+1}\sqrt{\ln(\pi 2^n)}$ | $2^{\frac{n}{2}}$ |
| OFB | $2^{\frac{n}{2}}\sqrt{\ln\left(\frac{1}{1-\pi}\right)}$ | $2^{\frac{n-1}{2}}$ |
| CBC | $2^{\frac{n}{2}+1}\sqrt{\ln\left(\frac{1}{1-\pi}\right)}$ | $2^{\frac{n-1}{2}}$ |
| CFB | $2^{\frac{n}{2}+1}\sqrt{\ln\left(\frac{1}{1-\pi}\right)}$ | $\frac{1}{\sqrt{3}}2^{\frac{n+1}{2}}$ |

Table 1: Upper bounds for acceptable amount of data to be processed.

The boundaries presented in the second and in the third columns of table 4 are very close. So, the amount of data to be processed without key change can be set equal to the lowest ones which correspond to the numbers in the third column.

## References


[1] V. F. Kolchin, B. A. Sevastianov, and V. P. Chistyakov, Random allocations, Nauka, Moscow, 1976.

[2] R 1323565.1.005-2017. Information technology. Cryptographic techniques. Acceptable amount of data to be processed without key change for particular block cipher modes of operation (in Russian). GOST, 2017.

[3] ISO/IEC 10116. Information technology – Security techniques – Modes of operation for an n-bit block cipher. ISO/IEC, 2001.

[4] NIST. Recommendation for Block Cipher Modes of Operation: Methods and Techniques. NIST Special Publication 800-38A, December 2001.

[5] A. Alkassar, A. Geraldy, B. Pfitzmann, A.-R. Sadeghi. Optimized self-synchronizing mode of operation. In Mitsuru Matsui, editor, Fast Software Encryption, 8th International Workshop, FSE 2001 Yokohama, Japan, April 2-4, 2001, Revised Papers, volume 2355 of Lecture Notes in Computer Science, pages 78–91. Springer, 2001.

[6] Bard V.G. Modes of Encryption Secure against Blockwise-Adaptive Chosen-Plaintext Attack. Cryptology ePrint Archive, Report 2006/251.

[7] Bellare, M., Desai, A., Jokipii, E., and Rogaway, P. A concrete security treatment of symmetric encryption. In 38th Annual Symposium on Foundations of Computer Science (Miami Beach, Florida, Oct. 19-22, 1997), IEEE Computer Society Press, pp. 394-403.

[8] Bellare, M., Kilian, J., and Rogaway, P. The security of cipher block chaining. In Advances in Cryptology - CRYPTO'94 (Santa Barbara, CA, USA, Aug. 21-25, 1994), Y. Desmedt, Ed., vol. 839 of Lecture Notes in Computer Science, Springer, Berlin, Germany, pp. 341-358.

[9] P. Fouque, A. Joux, and G. Poupard. Blockwise Adversarial Model for On-line Ciphers and Symmetric Encryption Schemes. Selected Areas of Cryptography. In Selected Areas of Cryptography, 2004. pp. 212-226.

[10] Goldwasser, S., and Micali, S. Probabilistic encryption. Journal of Computer and System Sciences 28, 2 (1984), 270-299.

[11] A. Joux, G. Martinet, and F. Valette Blockwise-Adaptive Attackers Revisiting the (in)security of some provably secure Encryption Modes: CBC, GEM, IACBC. In CRYPTO'02. pp. 17-31.

[12] McGrew, D. Impossible plaintext cryptanalysis and probable-plaintext collision attacks of 64-bit block cipher modes. Cryptology ePrint Archive, Report 2012/623.

[13] K. Nishimura, M. Sibuya, Probability To Meet in the Middle, J. Cryptology, Vol. 2, no. 1, 1990, pp. 13-22.

[14] Razali E., Phan C.-W. R., Joye M. On the Notions of PRP-RKA, KR and KR-RKA for Block Ciphers. In Provable Security 2007, LNCS 4784, pp. 188-197.

[15] Rogaway, P. Evaluation of Some Blockcipher Modes of Operation. CRYPTREC, 2011.

[16] Rogaway, P. Nonce-based symmetric encryption. In Fast Software Encryption - FSE 2004 (New Delhi, India, Feb. 5-7, 2004), B. Roy and W. Meier, Eds., vol. 3017 of Lecture Notes in Computer Science, Springer, Berlin, Germany, pp. 348-359.

# XS-circuits in Block Ciphers

Sergey Agievich

Research Institute for Applied Problems of Mathematics and Informatics,
Belarusian State University, Minsk, Belarus
agievich@bsu.by

**Abstract**

XS-circuits describe block ciphers that utilize 2 operations: X) bitwise modulo 2 addition of binary words and S) substitution of words using key-dependent $S$-boxes with possibly complicated internal structure. We propose a matrix model of XS-circuits which, despite simplicity, covers rather wide range of block ciphers. We obtain results on invertibility, transitivity and 2-transitivity of mappings induced by XS-circuits that satisfy our model. We introduce the similarity relation between circuits and provide canonical representatives of classes of similar circuits.

**Keywords:** block cipher, round permutation, $S$-box, circuit, diffusion, transitivity, 2-transitivity.

## 1 Introduction

A circuit is a directed acyclic graph that describes some algorithm. Leaves of the circuit are inputs of the algorithm, non-leaves are either intermediate results or outputs. Non-leaves are labeled by symbols of operations. The configuration when a vertex $v$ with a label O receives arcs from vertices $u_1, u_2, \ldots$ means that $v = \mathsf{O}(u_1, u_2, \ldots)$.

Many symmetric cryptographic algorithms can be described by circuits in which vertices are binary words of particular length $m$ and operations belong to the following set:

R) cyclic shift (rotation);

X) bitwise modulo 2 addition;

A) addition of words as integers modulo $2^m$;

L) bitwise logical AND and OR;

M) multiplication of words as elements of the field of order $2^m$;

S) substitution of words with preservation of their length $m$.

Here R and S are unary operations, they are parametrized by a shift value and a substitution rule respectively. All other operations are binary. For small $m$, S is usually implemented using so-called table $S$-boxes through table lookup.

Different combinations of operations give different types of circuits. Some of them, for example, circuits of types ARX and LRX, have gained much attention in the last decade. In the circuits mentioned, the operation S is intentionally not used to avoid table lookup. That is because in modern processors the time of lookup can depend on the sequence of lookup queries and this dependence forms the basis for mounting timing attacks. But S should not only mean table $S$-boxes. The operation S can represent a complex cryptographic transformation, possibly built using another circuit with a smaller length of processing words. The internal circuit of S can be of type ARX or LRX and, therefore, be protected against timing attacks.

The simplest nontrivial circuits with the operation S are the circuits of type XS. They describe, for example, Feistel ciphers or encryption modes like CBC. In the first case, S is instantiated by round functions with (in general) different round keys. In the second case, S is itself a block cipher with some fixed key.

The examples above are typical. In the examples, S describes a key-dependent and therefore a priori secret transformation. Following the cryptographic tradition, we say that S is instantiated by an *oracle* $S$: its response $v = S(u)$ to a query $u$ can be determined only by querying.

In most cases below we require that $S$ is bijective. Responses of such an oracle are weakly connected with each other: $S$ returns different $v$ when processing different $u$. That is the only a priori information on responses. For bijective $S$, we allow access to the inverse oracle $S^{-1}$ which on a query $v$ gives a response $u$.

If a circuit contains several operations S, then they can be instantiated by independent oracles $S_1, S_2, \ldots$ or by multiple identical instances of a single oracle $S$. Feistel ciphers are described by circuits of the first type (call them *inhomogeneous*), encryptions modes by circuits of the second type (*homogeneous*).

A circuit of a block cipher usually contains multiple identical parts connected consecutively. These parts represent round permutations of the cipher. Further we consider the simplest round circuits which contain only one S-operation. In Section 2 we provide a matrix model of such circuits. Similar models were proposed in [1, 3, 4, 13]. Our model is stricter, and due to this fact we obtain more targeting and precise results. In Section 4 we introduce

cascades, that is, inhomogenious compositions of round circuits. Sections 5, 6 deal with diffusion characteristics of cascades. These characteristics are related to cryptographic strength of corresponding block ciphers. Section 7 is devoted to the similarity relation between round circuits. Cascades of related circuits have the same diffusion characteristics. We provide canonical representatives of classes of similar circuits.

As a final remark, there is only one step from XS- to XMS-circuits. A circuit of the latter type is used, for example, in the AES block cipher. Usually XMS-circuits are classified as XLS where L stands not for logical operations but for linear transformations over tuples of underlying words. Actually, these transformations are described by circuits of type XM, so the overall XLS-circuit indeed has type XMS in our notations. It is interesting that XMS-circuits are also extensively used in message authentication algorithms like GCM [14]. In these algorithms, S is a block cipher with a fixed key (the homogenious case).

## 2 Preliminaries

Consider a circuit with the same number $n$ of inputs and outputs. Call $n$ its *dimension*. Let $x_1, \ldots, x_n$ be inputs and $y_1, \ldots, y_n$ be outputs. They are binary words of length $m$ which we interpret as elements of the field $\mathbb{F}_{2^m}$. In most cases, the specific value of $m$ does not matter, so we usually write $F$ instead of $\mathbb{F}_{2^m}$. Arrange inputs and outputs into the vectors $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$.

Elements of $F$ can be added together using the operation X and substituted separately using the operation S. To simplify notations for sums, we multiply each potential summand by zero or unity of the field $F$ and sum all the resulting products. Multiplication by 1 means inclusion into the sum, multiplication by 0 means exclusion.

We call the number of S operations used in the circuit its *S-complexity*. Further we concentrate mostly on the circuits of S-complexity 1. From these simplest circuits a circuit of arbitrary complexity can be built.

For each instantiation of its S operations, the circuit induces a mapping $F^n \to F^n \colon x \mapsto y$. We are mainly interested in such a circuit that all these mappings are invertible. Two circuits of the same S-complexity are equivalent if their mappings are necessarily identical under identical instantiations. Among all pairwise equivalent circuits, find one which contains the minimum number of X operations. Call this number the *X-complexity* and assign it to all circuits of the equivalence class.

A circuit of dimension $n$ and $\mathsf{S}$-complexity 1 can be described by three parameters: a column vector $a = (a_1, \ldots, a_n)^T$, a matrix $B = (b_{ij})$, $i, j = 1, \ldots, n$, and a row vector $c = (c_1, \ldots, c_n)$. Coordinates of the vectors and elements of the matrix belong to the set $\{0, 1\} \subset F$. Despite binarity, $a$, $B$ and $c$ can be used in operations with arbitrary vectors and matrices over $F$.

The parameters $(a, B, c)$ and an oracle $S$, some instantiation of $\mathsf{S}$, describe the following mapping $x \mapsto y$:

1) $u \leftarrow a_1x_1 + a_2x_2 + \ldots + a_nx_n$;

2) $v \leftarrow S(u)$;

3) for $i = 1, \ldots, n$: $y_i \leftarrow b_{1i}x_1 + b_{2i}x_2 + \ldots + b_{ni}x_n + c_iv$.

Denote this mapping by $(a, B, c)[S]$. It can be written in the matrix form:

$$(a, B, c)[S](x) = xB + S(xa)c.$$

Let us exclude from consideration zero vectors $a$ and $c$, because with them the $\mathsf{S}$-complexity actually reduces to 0. Indeed, if $a = 0$ then $S$ gets only one (zero) query, and if $c = 0$ then $S$ is not queried at all.

It is convenient to encode the parameters $(a, B, c)$ by the matrix

$$\begin{pmatrix} B & a \\ c & 0 \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12} & \ldots & b_{1n} & a_1 \\ b_{21} & b_{22} & \ldots & b_{2n} & a_2 \\ & \ldots\ldots\ldots & & & \\ b_{n1} & b_{n2} & \ldots & b_{nn} & a_n \\ c_1 & c_2 & \ldots & c_n & 0 \end{pmatrix}.$$

Call it the *extended matrix* of the circuit. In Table 1 we provide extended matrices of some well-known circuits.

In each column of an extended matrix there is at least one unity (otherwise, either the corresponding circuit zeroizes some output coordinate or $a = 0$). Under direct implementation of the extended matrix, each next unity in the column requires an extra $\mathsf{X}$ addition. From here we obtain the upper bound on the $\mathsf{X}$-complexity of $(a, B, c)$: the number of unities in its extended matrix minus the number of columns.

Returning to Table 1, the Feistel, GFN1, Matsui and Skipjack circuits all have $\mathsf{X}$-complexity 1. The $\mathsf{X}$-complexity of LaiMassey, SMS4 and MARS3 is equal to 3.

| Circuit | Extended matrix | Comments |
|---|---|---|
| Feistel | $\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$ | Used in the Lucifer and DES block ciphers which were developed under the direction of H. Feistel [9]. |
| LaiMassey | $\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}$ | Used by Lai X. and J. Massey in the IDEA block cipher [11]. |
| Matsui | $\begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ | Used by M. Matsui in the MISTY2 block cipher [17]. |
| SkipjackA | $\begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$ | Used in the Skipjack block cipher [15]. Describes its first and third 8-round cascades. |
| SkipjackB | $\begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$ | Describes the second and forth 8-round cascades of Skipjack. |
| MARS3 | $\begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{pmatrix}$ | Used in the MARS block cipher [5]. In the specification of MARS the circuit is called the type-3 Feistel network. We modify the original circuit by replacing two operations A by X. |
| SMS4 | $\begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$ | Used in the SMS4 block cipher [8]. |
| GFN1 | $\begin{pmatrix} 0 & 0 & \dots & 0 & 1 & 1 \\ 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 & 0 \\ & & \cdots\cdots\cdots & & & \\ 0 & 0 & \dots & 1 & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 & 0 \end{pmatrix}$ | The generalization of Feistel to an arbitrary dimension. Introduced in [21] under the name Generalized Feistel Network of type 1. |
| SkipjackG | $\begin{pmatrix} 0 & 0 & \dots & 0 & 0 & 1 \\ 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 & 0 \\ & & \cdots\cdots\cdots & & & \\ 0 & 0 & \dots & 1 & 0 & 0 \\ 1 & 0 & \dots & 0 & 1 & 0 \end{pmatrix}$ | The generalization of the Skipjack circuits to an arbitrary dimension. Proposed in [19]. |

Table 1: Extended matrices of XS-circuits of S-complexity 1

## 3 Invertibility

We have agreed to concentrate on circuits that induce invertible mappings. Let us give a formal definition.

**Definition 1.** *A circuit $(a, B, c)$ with nonzero $a$ and $c$ is invertible if the corresponding mapping $(a, B, c)[S]$ is invertible for any bijective oracle $S$ over any field $F = \mathbb{F}_{2^m}$.*

**Theorem 1.** *A circuit $(a, B, c)$ of dimension $n$ is invertible if and only if one of the following cases holds:*

1. *The matrix $B$ is invertible and $cB^{-1}a = 0$.*
2. *The matrices $B$, $(B\ a)$ and $\binom{B}{c}$ have ranks $n-1$, $n$ and $n$ respectively.*

*In the second case, the extended matrix of the circuit is invertible.*

*Proof.* Let us consider 2 cases: $B$ is invertible or not.

**1**. Let $B$ be invertible. Then $yB^{-1} = x + S(xa)cB^{-1}$ and $yB^{-1}a = xa + S(xa)cB^{-1}a$.

**1.1**. If $cB^{-1}a \neq 0$ then

$$xa + S(xa) = yB^{-1}a.$$

For the circuit to be invertible it is necessary that for any $v = yB^{-1}a$ there exists a solution $u = xa$ of the equation $u + S(u) = v$. But the mapping $u \mapsto u + S(u)$ can be non-bijective (that is, $S$ may not be a complete mapping), and the target equation may not have solutions for a certain $v$.

**1.2**. If $cB^{-1}a = 0$ then $xa = yB^{-1}a$ and inversion is defined by the equation

$$x = yB^{-1} + S(yB^{-1}a)cB^{-1}.$$

**2**. Let $B$ be non-invertible. To determine $x$ from $y = xB + S(xa)c$ it is necessary to get the response $S(xa)$ of $S$. This response can be obtained either directly from $y$ or indirectly by determining $xa$ from $y$ and then using the query $xa$ to $S$.

**2.1**. To determine $xa$ from $y$ there must exist a row vector $\alpha \in F^n$ such that $B\alpha = a$, $c\alpha = 0$ and consequently $xa = y\alpha$. After determining $u = xa$ we can find $v = S(u)$ and obtain the equation $x(B\ a) = (y + vc, u)$ in $x$. This equation can have more than one soultions since the matrix $(B\ a)$ does not have full rank. Indeed, $B$ is not invertible and $a = B\alpha$ is a linear combination of columns of $B$.

**2.2**. Suppose that $S(xa)$ can be determined by $y$. Then $\alpha$ has to satisfy the equations $B\alpha = 0$ and $c\alpha = 1$ which can be used to calculate $v = S(xa) = y\alpha$ and $u = xa = S^{-1}(v)$. After determining $u$, we again obtain the equation $x(B\ a) = (y + vc, u)$. In order that this equation has a unique solution, the matrix $(B\ a)$ has to have full rank. If $\operatorname{rank}(B\ a) = n$ then $\operatorname{rank} B = n - 1$. Therefore, all nonzero row vectors $\beta \in F^n$ such that $B\beta = 0$ are collinear to $\alpha$. Since $c\alpha = 1$ and consequently $c\beta \neq 0$, $\operatorname{rank}\binom{B}{c} = n$.

**3**. If $\operatorname{rank} B = n - 1$ and $\operatorname{rank}(B\ a) = \operatorname{rank}\binom{B}{c} = n$ then the extended matrix $\left(\begin{smallmatrix} B & a \\ c & 0 \end{smallmatrix}\right)$ has full rank. Indeed, none of the rows of $(B\ a)$ can be expressed as a linear combination of other rows. In case the row $(c\ 0)$ is a linear combination of rows of $(B\ a)$, the vector $c$ is a linear combination of rows of $B$. But it contradicts the fact that $\binom{B}{c}$ has full rank. □

We refer the circuits which correspond to the different cases of Theorem 1 as circuits of type I and type II respectively. From the proof above it follows that a type I circuit is invertible even if its oracle $S$ is not bijective. In Table 1 only the Skipjack and Matsui circuits are of type II.

**Theorem 2.** *For an invertible circuit $(a, B, c)$ with an oracle $S$ the inverse mapping $(a, B, c)[S]^{-1}$ is again determined by an $\mathsf{XS}$-circuit of $\mathsf{S}$-complexity* 1. *This inverse circuit is defined as follows:*

1. *In the first case of Theorem 1 the inverse circuit uses the same oracle $S$ and its description is $(B^{-1}a, B^{-1}, cB^{-1})$.*
2. *In the second case of Theorem 1 the inverse circuit uses the inverse oracle $S^{-1}$ and its extended matrix is inverse of the extended matrix of $(a, B, c)$.*

*Proof.* Let us continue the previous proof. The inverse circuit of the first case was already described in clause 1.2. Consider the second case.

The left bottom element of the inverted extended matrix must be 0:

$$\begin{pmatrix} B & a \\ c & 0 \end{pmatrix}^{-1} = \begin{pmatrix} D & \alpha \\ \gamma & 0 \end{pmatrix}.$$

Indeed, otherwise $B\alpha = a$ which contradicts the fact that $(B\ a)$ has full rank.

Return to the equation $x(B\ a) = (y + vc, u)$ of clause 2.2. Multiplying both parts of this equation by the matrix $\binom{D}{\gamma}$ we obtain

$$x = yD + vcD + u\gamma.$$

The required result follows from the fact that $cD = 0$ and $u = S^{-1}(v) = S^{-1}(y\alpha)$. $\square$

Further we denote the inverse of a circuit $(a, B, c)$ by $(a, B, c)^{-1}$.

**Example 1.** *The matrix $B$ of the GFN1 circuit is a special circulant: multiplication of a row (column) vector on the right (left) by $B$ causes left (right) cyclic shift of the vector. The matrix $B^{-1}$ induces cyclic shifts in the reverse direction. Therefore, the extended matrix $\left(\begin{smallmatrix} B^{-1} & B^{-1}a \\ cB^{-1} & 0 \end{smallmatrix}\right)$ of GFN1$^{-1}$ has the form*

$$\begin{pmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ & & \dots\dots\dots & & & \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 1 & 0 & 0 & \dots & 0 & 1 \\ 0 & 1 & 0 & \dots & 0 & 0 \end{pmatrix}.$$

*The extended matrix of SkipjackG$^{-1}$:*

$$\begin{pmatrix} 0 & 0 & \dots & 0 & 0 & 1 \\ 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 & 0 \\ & & \dots\dots\dots & & & \\ 0 & 0 & \dots & 1 & 0 & 0 \\ 1 & 0 & \dots & 0 & 1 & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ & & \dots\dots\dots & & & \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 1 & 0 & \dots & 0 & 1 \\ 1 & 0 & 0 & \dots & 0 & 0 \end{pmatrix}.$$

## 4 Regularity

Combine several instances of a circuit $(a, B, c)$ by connecting one instance's output to the next instance's input. Call the resulting XS-circuit a *cascade*. Its dimension is the dimension of the underlying circuit $(a, B, c)$. The cascade is invertible if $(a, B, c)$ is invertible.

In cryptography, instances, parts of a cascade, are usually called *rounds*. We suppose that rounds use independent oracles $S_1, S_2, \dots$ or, in other words, cascades are inhomogeneous XS-circuits. Suppose also that round oracles are bijective.

Let $(a, B, c)^t$ be the $t$-round cascade. Its S-complexity equals $t$ and X-complexity does not exceed the the total X-complexity of the rounds. If $(a, B, c)^t$ is invertible then its inverse is the $t$-round cascade $(a, B, c)^{-t}$ which contains $t$ rounds of $(a, B, c)^{-1}$ and also has S-complexity $t$.

Setting some $y(0) \in F^n$ as the cascade input, we obtain $y(1) \in F^n$ after the first round, $y(2) \in F^n$ after the second one and so on.

Let $(a, B, c)^t[S_1, \ldots, S_t]$ be the mapping $y(0) \mapsto y(t)$ induced by the cascade $(a, B, c)^t$ with oracles $S_1, \ldots, S_t$.

Round outputs satisfy the following equations:

$$y(t) = y(0)B^t + \sum_{\tau=1}^{t} S_\tau(y(\tau-1)a)cB^{t-\tau}, \quad t = 1, 2, \ldots.$$

They can be rewritten as follows:

$$\begin{aligned} y(t) &= y(0)B^t + \sum_{\tau=1}^{t} v(\tau)cB^{t-\tau}, \\ v(t) &= S_t(u(t)), \\ u(t) &= y(0)B^{t-1}a + \sum_{\tau=1}^{t-1} v(\tau)cB^{t-1-\tau}a, \quad t = 1, 2, \ldots. \end{aligned}$$

Here $u(1), \ldots, u(t)$ is the *trace of queries* and $v(1), \ldots, v(t)$ is the *trace of responses*. More precisely, we deal with *t-traces*. Since round oracles are independent, there exist $|F|^t$ different $t$-traces of each type.

In the cryptographic context, each cascade's round has to establish complex dependencies between certain coordinates of input and output vectors and simultaneously has to shuffle all the coordinates. Using related terms of Shannon, rounds are responsible for confusion and diffusion. In our case confusion is managed by round oracles, diffusion is maintained by the round circuit $(a, B, c)$ itself.

Further we introduce several characteristics of diffusion. In particular, we will analyze how a cascade processes not one but two vectors: $y(0)$ and $y'(0)$. The additional vector $y'(0)$ produces an additional sequence $y'(1), y'(2), \ldots$ of round outputs. This sequence induces an additional trace of queries and is induced by an additional trace of responses. A query $u'(t)$ and, consequently, a corresponding response $v'(t)$ can differ from $u(t)$ and $v(t)$. Traces are *compatible*, that is, each oracle returns the same responses to the same queries and different responses to different queries.

We are interested in the dynamics of the *differences*

$$\Delta y(t) = y(t) + y'(t), \quad \Delta u(t) = u(t) + u'(t), \quad \Delta v(t) = v(t) + v'(t)$$

during the rounds. In the equations above $+$ can be replaced with $-$ (because $F$ is a field of characteristic 2), that is why the term "difference" is relevant.

The difference $\Delta u(t)$ is the input difference of $S_t$, $\Delta v(t)$ is the output

one. The relation between these differences can be written as follows:

$$\Delta v(t) = \Delta S_t(\Delta u(t)).$$

The compatibility of traces means that $\Delta v(t) = 0$ if and only if $\Delta u(t) = 0$.

In cryptography, the event $\Delta u(t) \neq 0$ is called the *activation* of $S_t$. In case of the activation, the output difference $\Delta v(t)$ is hard to predict during cryptanalysis. The more activations a cascade guarantees while processing different $y(0)$ and $y'(0)$, the higher quality of diffusion.

Relations between differences are derived from the previous equations by inserting the symbol $\Delta$ before the expressions $y(t)$, $y(0)$, $v(\tau)$, $v(t)$, $S_t$, $u(t)$, etc. For example,

$$\Delta u(t) = \Delta y(0)B^{t-1}a + \sum_{\tau=1}^{t-1} \Delta v(\tau)cB^{t-1-\tau}a.$$

**Definition 2.** *The lag of a circuit $(a, B, c)$ is the minimum positive integer $l$ such that $cB^{l-1}a = 1$.*

The lag $l$ characterizes the relationship between a query $u(t)$ and previous responses $v(1), \ldots, v(t-1)$: For a suffuciently large $t$ the query $u(t)$ depends on $v(t-l)$ but not on $v(t-l+1), \ldots, v(t-1)$. The smaller the lag, the higher quality of diffusion because unpredictable oracle's responses are used faster to create new queries. The lag also characterizes the relationship between $\Delta u(t)$ and $\Delta v(1), \ldots, \Delta v(t-1)$.

Circuits that provide reasonable (rational) diffusion are described by the following definitions. Further we justify the relevance of the requirements of these definitions.

**Definition 3.** *An invertible circuit $(a, B, c)$ of dimension $n$ is regular if the following conditions hold:*

*1) the matrix* $C = \begin{pmatrix} cB^{n-1} \\ \ldots \\ cB \\ c \end{pmatrix}$ *is invertible;*

*2) the matrix* $A = \begin{pmatrix} a & Ba & \ldots & B^{n-1}a \end{pmatrix}$ *is invertible.*

**Definition 4.** *A circuit $(a, B, c)$ of dimension $n$ is strongly regular if it is regular and additionally*

| Circuit | Lag | Inverse lag | Sum of lags |
|---|---|---|---|
| Feistel | 1 | 1 | 2 |
| Matsui | 2 | 1 | 3 |
| SkipjackA | 1 | 4 | 5 |
| SkipjackB | 4 | 1 | 5 |
| MARS3 | 1 | 1 | 2 |
| SMS4 | 1 | 1 | 2 |
| GFN1 | 1 | $n-1$ | $n$ |
| SkipjackG | 1 | $n$ | $n+1$ |

Table 2: Lags of the regular standard circuits

*3) the matrix* $C_l = \begin{pmatrix} cB^{(n-1)l} \\ \cdots \\ cB^l \\ c \end{pmatrix}$ *is invertible. Here* $l$ *is the lag of* $(a, B, c)$.

The forthcoming Corollary 3 shows that the lag of a regular circuit doesn't exceed its dimension. Therefore, in the last definition, $l$ is finite and the third condition is correct.

Trivially, if a regular circuit has lag 1 then this circuit is strongly regular. Further we prove more complicated facts, for example, the fact that mutually inverse circuits are both regular or both non-regular (Corollaries 1 and 2). Despite this fact, the following example shows that mutually inverse circuits are not necessarily strongly regular simultaneously.

**Example 2.** *FourCell is a circuit proposed in [6]. Its extended matrix has the form*

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

*FourCell has lag* 4. *The circuit is regular but not strongly regular. The inverse circuit has lag* 1 *and therefore is strongly regular.*

All circuits of Table 1 except LaiMassey are strongly regular. In Table 2 we report their lags as well as *inverse lags*, that is, lags of inverse circuits. The lags of $\mathrm{GFN1}^{-1}$ and $\mathrm{SkipjackG}^{-1}$ are easily calculated using Example 1.

## 5 Transitivity

**Definition 5.** *A cascade $(a, B, c)^t$ of dimension $n$ is transitive if for any $\alpha, \beta \in F^n$ there exist round oracles $S_1, \ldots, S_t$ such that*

$$(a, B, c)^t[S_1, \ldots, S_t](\alpha) = \beta.$$

*A circuit $(a, B, c)$ is transitive if $(a, B, c)^t$ is transitive for some $t$. The minimal such $t$ is the index of transititvity of $(a, B, c)$.*

Transitivity indeed characterizes diffusion in the sense that for a sufficiently large $t$ any $y(t)$ is reachable from any $y(0)$. The smaller the index of transitivity, the faster diffusion.

**Example 3.** *The LaiMassey circuit maps $x = (x_1, x_2)$ to*

$$y = (y_1 + S(x_1 + x_2), y_2 + S(x_1 + x_2)).$$

*This mapping saves the sum of coordinates: $y_1 + y_2 = x_1 + x_2$. The sum is not being changed during all further rounds and therefore the circuit is not transitive.*

**Theorem 3.** *The index of transitivity of a circuit $(a, B, c)$ does not exceed its dimension $n$. The index equals $n$ if and only if the first condition of regularity (invertibility of $C$) holds.*

*Proof.* Let $\alpha, \beta$ be arbitrary elements of $F^n$. The number of vectors $y(t)$ reachable from $y(0) = \alpha$ does not exceed the number of $t$-traces of responses. With $t < n$ this number is less than $|F^n|$ and therefore there exists unreachable $y(t)$. Consequently, the index of transitivity cannot be less than $n$.

Let $C$ be invertible. Then there exists a unique vector $v = (v(1), \ldots, v(n)) \in F^n$ such that

$$vC = \alpha B^n + \beta.$$

The responses $v(1), \ldots, v(n)$ transfer $y(0) = \alpha$ to $y(n) = \beta$:

$$y(n) = y(0)B^n + \sum_{\tau=1}^{n} v(\tau)cB^{n-\tau} = \alpha B^n + vC = \beta.$$

Therefore, $(a, B, c)^n$ is transitive.

Let $(a, B, c)^n$ be transitive. Suppose by contradiction that $C$ is not invertible. Then there exist $\alpha, \beta \in F^n$ such that the equation $vC = \alpha B^n + \beta$ does not have solutions in $v$. It means that no trace of responses transfers $y(0) = \alpha$ to $y(n) = \beta$, a contradiction. □

Note that transitivity does not require invertibility. For example, a circuit of dimension 1 which maps $x_1$ to $x_1 + S(x_1)$ is transitive but not invertible. However, below we need invertibility.

**Corollary 1.** *The first condition of regularity holds for an invertible circuit $(a, B, c)$ if and only if it holds for the inverse circuit $(a, B, c)^{-1}$.*

*Proof.* By definition, mutually inverse circuits are transitive simultaneously, their indices of transitivity coincide. The required result follows from the second part of Theorem 3. □

Call a binary operation *Latin* if its table is a Latin square or, in other words, if the operation induces a quasigroup on the underlying set. Latin operations are often used in cryptography, for example, to instantiate round oracles using round keys, as in the following theorem. The theorem means that a circuit which dimension $n$ is equal to its index of transitivity can be used to extend a Latin operation on $F$ to a Latin operation on $F^n$.

**Theorem 4.** *Let a cascade $(a, B, c)^n$ of dimension $n$ be transitive and use the oracles*

$$S_t^k(u) = S(u * k_t), \quad u \in F, \quad t = 1, \ldots, n,$$

*where $k = (k_1, \ldots, k_n) \in F^n$, $S$ is a fixed permutation on $F$, and $*$ is a Latin operation on $F$. Then the operation*

$$O\colon F^n \times F^n \to F^n,\ (\alpha, k) \mapsto (a, B, c)^n[S_1^k, \ldots, S_n^k](\alpha)$$

*is Latin too.*

*Proof.* Firstly, due to transitivity of $(a, B, c)^n$ there exists a unique $n$-trace of responses which transfers any given $\alpha$ to any given $\beta$. Since $*$ is Latin, this trace unambiguously determines $k$, that is, the equation $O(\alpha, k) = \beta$ has a unique solution in $k$. Secondly, due to invertibility any given $\beta$ and $k$ unambiguously determine $\alpha$, that is, the equation $O(\alpha, k) = \beta$ has a unique solution in $\alpha$. In result, $O$ induces a quasigroup on $F^n$. □

## 6 2-transitivity

**Definition 6.** *A cascade $(a, B, c)^t$ of dimension $n$ is 2-transitive if for any distinct $\alpha, \alpha' \in F^n$ and any distinct $\beta, \beta' \in F^n$ there exist round oracles $S_1, \ldots, S_t$ such that*

$$(a, B, c)^t[S_1, \ldots, S_t](\alpha) = \beta, \quad (a, B, c)^t[S_1, \ldots, S_t](\alpha') = \beta'.$$

2-transitivity is an important diffusion property of cascades. In particular, 2-transitivity of $(a, B, c)^t$ implies absence of so-called *impossible differentials*, that is, unrealizable transitions from some difference $\Delta y(0) = \Delta\alpha$ to some difference $\Delta y(t) = \Delta\beta$. Such transitions can be used to mount impossible differential attacks.

In addition, 2-transitivity helps to determine the permutation group generated by the mappings $(a, B, c)[S]$, where $S$ runs over all bijections over $F$. Usually, 2-transitivity is a serious evidence that this group is the alternating group. It is the largest achievable group (for $n \geq 2$), its appearance demonstrates the welcomed diversity of the mappings $(a, B, c)[S]$.

Unfortunately, 2-transitivity is a rather complicated property which cannot be supported by such a simple criterion as in the case of transitivity (Theorem 3). Let us introduce a weakened version of 2-transitivity.

**Definition 7.** *A cascade $(a, B, c)^t$ of dimension $n$ is weakly 2-transitive if there do not exist nonzero $\Delta\alpha, \Delta\beta \in F^n$ such that $(a, B, c)^t[S_1, \ldots, S_t]$ necessarily, independently of the choice of the round oracles, tranfers the difference $\Delta y(0) = \Delta\alpha$ to the difference $\Delta y(t) = \Delta\beta$.*

As before, a circuit $(a, B, c)$ is *(weakly) 2-transitive* if $(a, B, c)^t$ is (weakly) 2-transitive for some $t$. The minimal such $t$ is the *index of (weak) 2-transitivity.*

**Example 4.** *Let us continue Example 3. The LaiMassey circuit is not weakly 2-transitive. Indeed, for any nonzero $\Delta\gamma \in F$ the difference $\Delta x = (\Delta\gamma, \Delta\gamma)$ goes to the difference $\Delta y = (\Delta\gamma, \Delta\gamma)$. This difference is being saved during all further rounds.*

Note that (weak) 2-transitivity, as well as transitivity, does not require invertibility.

**Theorem 5.** *The index of weak 2-transitivity of a circuit $(a, B, c)$ does not exceed its dimension $n$. The index equals $n$ if and only if the second condition of regularity (invertibility of $A$) holds.*

*Proof.* A cascade $(a, B, c)^t$ is not weakly 2-transitive if and only if there exists a nonzero input difference $\Delta y(0)$ which induces the zero vector $\Delta u = (\Delta u(1), \ldots, \Delta u(t))$ of internal differences between queries to the round oracles. The vector $\Delta u$ must exist because once two queries to some oracle $S_\tau$ are distinct, the corresponding responses are distinct too and the output difference $\Delta y(t)$ depends on the difference $\Delta v(\tau)$ between these responses. Note

that if $\Delta u = 0$, then the differences $\Delta v(1), \ldots, \Delta v(t)$ are zero too. This fact can be written as

$$\Delta y(0) \begin{pmatrix} a & Ba & \ldots & B^{t-1}a \end{pmatrix} = 0.$$

If $t < n$, then the last equation has a nonzero solution in $\Delta y(0)$. This solution induces zero $\Delta u$ and $(a, B, c)^t$ is not weakly 2-transitive. This proves the first part of the theorem.

If $A = \begin{pmatrix} a & Ba & \ldots & B^{n-1}a \end{pmatrix}$ is invertible then $(a, B, c)^n$ is weakly 2-transitive. Indeed, otherwise $\Delta y(0)A = 0$ for some nonzero $\Delta y(0)$, which is impossible.

Conversly, if $(a, B, c)^n$ is weakly 2-transitive then $A$ is invertible. Indeed, otherwise there exists a nonzero $\Delta y(0)$ which induces $\Delta u = 0$. □

**Corollary 2.** *The second condition of regularity holds for an invertible circuit $(a, B, c)$ if and only if it holds for the inverse circuit $(a, B, c)^{-1}$.*

*Proof.* By defintion, mutually inverse circuits are weakly 2-transitive simultaneously, their indices of weak 2-transitivity coincide. The required result follows from the second part of Theorem 5. □

**Theorem 6.** *Let circuits $(a, B, c)$ and $(a, B, c)^{-1}$ of dimension $n$ be strongly regular and*

$$\left(1 - \frac{2}{|F|}\right)^{n-1} \left(1 - \frac{1}{|F|}\right) > \frac{1}{2}.$$

*Then the circuits are 2-transitive and their indices of 2-transitivity do not exceed*

$$2n + (n-1)(l + l'),$$

*where $l$ is the lag of $(a, B, c)$ and $l'$ is the lag of $(a, B, c)^{-1}$.*

*Proof.* From the proof of Theorem 5 it follows that for any nonzero input difference $\Delta y(0)$ there exists $r \leq n$ such that $\Delta u(r)$, the difference between queries to $\mathsf{S}_r$, is nonzero. The corresponding difference $\Delta v(r) = \Delta \mathsf{S}_r(\Delta u(r))$ between responses is nonzero too.

Let $r$ be the first round when $\Delta u(r) \neq 0$. By definition of lag

$$\begin{aligned} \Delta u(r + l) &= \Delta v(r) + \Delta y(0) B^{r+l-1} a + \sum_{\tau=1}^{r-1} \Delta v(\tau) c B^{r+l-1-\tau} a \\ &= \Delta v(r) + \Delta y(0) B^{r+l-1} a. \end{aligned}$$

Manipulating responses of $\mathsf{S}_r$ (round oracles are free to choose which response to give), we obtain different $\Delta v(r)$. At least $|F| - 2$ of them provide $\Delta u(r + l) \neq 0$.

Having a nonzero $\Delta u(r+l)$, we tune a nonzero $\Delta v(r+l)$ to achieve a nonzero $\Delta u(r+2l)$. Continue in such a manner until the round number $r+(n-1)l$. In this round, we do not require that $\Delta u(r+nl) \neq 0$ and have $|F|-1$ ways to choose $\Delta v(r+(n-1)l)$.

Thus, there exist at least $(|F|-2)^{n-1}(|F|-1)$ vectors

$$\Delta v = (\Delta v(r), \Delta v(r+l), \ldots, \Delta v(r+(n-1)l))$$

with nonzero coordinates.

Let the oracles $\mathsf{S}_t$, $t \neq r+il$, implement the identity mapping, that is, they output input queries. Then

$$\Delta y(r+(n-1)l) = \Delta y(0)M + \Delta v C_l,$$

where $M$ is some matrix of order $n$, $C_l$ is the matrix of the definition of strong regularity. Due to the invertibility of $C_l$, different $\Delta v$ induce different $\Delta y(r+(n-1)l)$. Therefore, the difference $\Delta y(r+(n-1)l)$ can take at least $(|F|-2)^{n-1}(|F|-1)$ distinct values.

To provide the required difference $\Delta v(t)$, $t = r+il$, the oracle $\mathsf{S}_t$ first returns an arbitrary $\mathsf{S}_t(u(t))$ and then the specific $\mathsf{S}_t(u'(t)) = \mathsf{S}_t(u(t)) + \Delta v(t)$. By choosing a vector $v = (v(r), v(r+l), \ldots, v(r+(n-1)l))$ of the first responses, achieve that the vector

$$y(r+(n-1)l) = y(0)M + vC_l$$

takes a fixed value $\gamma \in F^n$.

In sum, applying the circuit $(a,B,c)^{r+(n-1)l}$ to a given pair $(\alpha, \alpha')$, $\alpha \neq \alpha'$, and running over all possible round oracles, we obtain at least $(|F|-2)^{n-1}(|F|-1)$ different pairs $(\gamma, z)$, $z \in F^n$.

The same holds for the inverse circuit $(a,B,c)^{-1}$: The circuit $(a,B,c)^{-r'-(n-1)l'}$, $r' \leq n$, with various round oracles transfers a given pair $(\beta, \beta')$, $\beta \neq \beta'$, to at least $(|F|-2)^{n-1}(|F|-1)$ different pairs $(\gamma, z')$, $z' \in F^n$.

Under conditions of the theorem,

$$2(|F|-2)^{n-1}(|F|-1) > |F|^n$$

and there must exist a collision $z = z'$. This collision means that the pair $(\alpha, \alpha')$ can be transferred to the pair $(\beta, \beta')$ by the circuit $(a,B,c)^{r+r'+(n-1)(l+l')}$. This implies the required result. $\square$

The additional condition of Theorem 6 is not burdensome. It holds for

| Circuit | Upper bound (Theorem 6) | Lower bound |
|---|---|---|
| Feistel | 6 | 6 [10] |
| Matsui | 7 | |
| SkipjackA | 22 | 17 [3] |
| SkipjackB | 22 | 17 [3] |
| MARS3 | 14 | 12 [16] |
| SMS4 | 14 | 12 [16] |
| GFN1 ($n = 4$) | 20 | 20 [7] |
| SkipjackG ($n = 4$) | 22 | 17 [16] |

Table 3: Bounds on the indices of 2-transitivity

example if $|F| = 2^m > 4n$. In practice, $m \geq 16$, $n \leq 8$ and the condition indeed satisifies.

In Table 3 we present bounds on the indices of 2-transitivity of the standard circuits. Upper bounds are built using Theorem 6 and Table 2. Lower bounds are the quantities $d + 1$, where $d$ is the maximum known number of rounds such that an impossible differential for $(a, B, c)^d$ exists.

Note that the upper bounds for SkipjackA and SkipjackB presented in Table 3 should not be transferred on the Skipjack cipher itself. In this cipher 8 SkipjackA rounds are followed by 8 SkipjackB rounds and otherwise.

The proof of Theorem 6 can be easily extended to the case when the last rounds of a cascade differ from the first ones. In particular, using the fact that SkipjackA and $\text{SkipjackB}^{-1}$ both have lag 1, the cascade of first 7 SkipjackA and then 7 SkipjackB rounds is 2-transitive.

It is interesting that although the 14-round cascade $\text{SkipjackA}^7\text{SkipjackB}^7$, as well as 22-round cascades $\text{SkipjackA}^{22}$ and $\text{SkipjackB}^{22}$ are 2-transitive (we multiply round circuits from left to right), the 24-round cascade

$$\text{SkipjackA}^4\text{SkipjackB}^8\text{SkipjackA}^8\text{SkipjackB}^4$$

is not (see [2] for details).

## 7 Similarity

**Definition 8.** *Circuits $(a, B, c)$ and $(a', B', c')$ of dimension $n$ are similar if there exists an invertible $(0, 1)$-matrix $P$ of order $n$ such that $a' = P^{-1}a$, $B' = P^{-1}BP$, $c' = cP$.*

Similarity means that if $y = (a, B, c)[S](x)$ and $x' = xP$, $y' = yP$

then $y' = (a, B, c)[S](x')$. Indeed, from $y = xB + S(xa)c$ it follows that

$$yP = xPP^{-1}BP + S(xPP^{-1}a)cP$$

or

$$y' = x'B' + S(x'a')c'.$$

The conclusion above is easily extended to several rounds: If $(a, B, c)^t[S_1, S_2, \ldots, S_t]$ transfers $y(0)$ to $y(t)$ then $(a', B', c')^t[S_1, S_2, \ldots, S_t]$ transfers $y'(0) = y(0)P$ to $y'(t) = y(t)P$. It means that similar circuits have the same cryptographic quality. In particular, they have the same type, lag, indices of transitivity and (weak) 2-transitivity, they are (strongly) regular simultaneously. At the same time, mutually similar circuits can have different X-complexity. To reduce the number of X operations, a circuit can be replaced by a similar one.

Similarity is an equivalence relation. It is natural to pose the problem of determining canonical representatives of equivalence classes as well as other classification problems.

Manipulating $P$ and replacing $B$ by $P^{-1}BP$, we can bring $B$ to a convenient matrix canonical form. Let us use the Frobenius normal form:

$$B = \operatorname{diag}(B_1, B_2, \ldots, B_k).$$

Here $B_i$ are Frobenius cells, that is, companion matrices of polynomials $f_{B_i}(\lambda) \in \mathbb{F}_2[\lambda]$. The polynomials divide each other: $f_{B_1}(\lambda) \div f_{B_2}(\lambda) \div \ldots \div f_{B_k}(\lambda)$.

The condition $k = 1$ is necessary for regularity of a circuit. Indeed, by the Cayley–Hamilton theorem the matrix $B$ is a root of $f_{B_k}$. If $k > 1$, then $\deg f_{B_k} < n$ and some nonzero linear combination of the matrix powers $B^0, B^1, \ldots, B^{n-1}$ drops to zero. But it means that the matrices $C$ and $A$ of the definition of regularity do not have full rank, that is, regularity does not hold.

Further we consider only single-cell canonical matrices $B$. Such a matrix has the form:

$$\begin{pmatrix} 0 & 0 & \ldots & 0 & 0 & b_1 \\ 1 & 0 & \ldots & 0 & 0 & b_2 \\ 0 & 1 & \ldots & 0 & 0 & b_3 \\ & & \ldots\ldots\ldots & & & \\ 0 & 0 & \ldots & 1 & 0 & b_{n-1} \\ 0 & 0 & \ldots & 0 & 1 & b_n \end{pmatrix}.$$

Its characteristic polynomial $f_B(\lambda) = \lambda^n + b_n\lambda^{n-1} + \ldots + b_1$. The coefficient $b_1$

equals 1 for circuits of type I and 0 for circuits of type II.

**Theorem 7.** *Let $(a, B, c)$ be a circuit of dimension $n$ in which $B$ is a Frobenius cell with a characteristic polynomial $\lambda^n + b_n\lambda^{n-1} + \ldots + b_1$. The circuit is invertible if and only if one of the following cases holds:*

*1)* $b_1 = 1$ *and* $a_1(b_2c_1+b_3c_2+\ldots+b_nc_{n-1}+c_n)+a_2c_1+a_3c_2+\ldots+a_nc_{n-1} = 0$*;*

*2)* $b_1 = 0$*,* $a_1 = 1$ *and* $b_2c_1 + b_3c_2 + \ldots + b_nc_{n-1} + c_n = 1$*.*

*There exist* $2^{2n-1} - 3 \cdot 2^{n-1} + 1$ *suitable pairs* $(a, c)$ *in the first case and* $2^{2n-2}$ *in the second.*

*Proof.* Let us apply Theorem 1. If $b_1 = 1$ then the invertibility requires that $cB^{-1}a = 0$. The stated result follows from the fact that

$$B^{-1} = \begin{pmatrix} b_2 & 1 & 0 & \ldots & 0 \\ b_3 & 0 & 1 & \ldots & 0 \\ & \ldots & \ldots & & \\ b_n & 0 & 0 & \ldots & 1 \\ b_1 & 0 & 0 & \ldots & 0 \end{pmatrix}.$$

If $b_1 = 0$ then the matrices $(B\ a)$ and $\binom{B}{c}$ must have full rank $n$. For the first matrix, it is true if and only if $a_1 = 1$. The second matrix have full rank if and only if $c$ cannot be expressed linearly through the last $n-1$ rows of $B$. It is equivalent to the inequality $b_2c_1 + b_3c_2 + \ldots + b_nc_{n-1} \neq c_n$ written in the statement of the theorem in a slightly different form.

The second case of the last part of the theorem is obvious. To treat the first case, we have to determine the number of pairs $(a, c)$ which make the quadratic form $g(a, c) = cB^{-1}a$ equal to 0. The form $g$ is linearly equivalent to $a_1c_n + a_2c_1 + \ldots + a_nc_{n-1}$ and the required number of pairs is $2^{2n-1} + 2^{n-1}$ (see, for example, [12, Theorem 6.32]). From this number we have to subtract $2^{n+1} - 1$, the number of pairs $(a, c)$ such that $a = 0$ or $c = 0$. □

Let $P$ be a $(0, 1)$-normalizer of $B$, that is, an invertible matrix such that $P^{-1}BP = B$. Using $P$, we can bring $(a, B, c)$ to the form $(P^{-1}a, B, cP)$ in which, generally, the vectors $a$ and $c$ are changed but the matrix $B$ is not. In other words, we can refine the canonical form not only the matrix but also the vectors of circuit's description.

Let $p_1, p_2, \ldots, p_n$ be consecutive rows of $P$. From the equality $PB = BP$

it follows that

$$\begin{aligned} p_n B &= p_{n-1} + b_n p_n, \\ p_{n-1} B &= p_{n-2} + b_{n-1} p_n, \\ &\dots \\ p_2 B &= p_1 + b_2 p_n. \end{aligned}$$

These equiations mean that all rows of $P$ can be expressed through $p_n$:

$$P = P(p_n) = \begin{pmatrix} p_n M_1 \\ p_n M_2 \\ .. \\ p_n M_n \end{pmatrix}.$$

Here $M_n = E$ and $M_i = BM_{i+1} + b_{i+1}E = B^{n-i} + b_n B^{n-i-1} + \ldots + b_{i+1}E$, $i = n-1, \ldots, 2, 1$, where $E$ is the identity matrix.

Multiplying $P(p_n)$ on the left by an invertible matrix, we can bring it to the form

$$\begin{pmatrix} p_n B^{n-1} \\ p_n B^{n-2} \\ \dots \\ p_n E \end{pmatrix}.$$

With $p_n = c$ this is the matrix $C$ of the definition of regularity. Thus, $P(p_n)$ is invertible if and only if the first condition of regularity holds with $c = p_n$. Moreover, there exists a bijective correspondence between acceptable vectors $c$ of regular circuits $(a, B, c)$ and normalizers $P$ of the matrix $B$: $c \leftrightarrow P(c)$.

The vector $c = (0, 0, \ldots, 0, 1)$ is acceptable because in the corresponding matrix $C$ the main diagonal contains only unities, all elements above the diagonal are zero and, therefore, $C$ is invertible. A normalizer $P(c')$ transfers a regular circuit $(a, B, c)$ to the similar circuit $(a', B, c')$. Indeed, $cP(c')$ is the last row of $P(c')$ which is $c'$. Moreover, only one $P(c')$ transfers $c$ to $c'$ and $a' = P(c')^{-1}a$ is uniquely determined.

This reasoning can be inverted: We can bring a regular circuit $(a', B, c')$ to the form $(a, B, c)$ in which $c = (0, 0, \ldots, 0, 1)$ and $a$ is uniquely determined. Simultaneously, acting in same manner, we can bring $(a', B, c')$ to the form $(a, B, c)$ in which $a = (1, 0, \ldots, 0, 0)^T$ and $c$ is uniquely determined. The chosen $a$ is acceptable because the corresponding matrix $A$ of the definition of regularity equals $E$.

Gathering all, we obtain the following result.

**Theorem 8.** *A regular circuit is similar to each of the following circuits:*

*1)* $((1,0,0,\ldots,0)^T, B, c)$*;*

*2)* $(a, B, (0,0,\ldots,0,1))$*.*

*Here* $B$ *is a uniquely determined Frobenius cell. The vectors* $a$ *and* $c$ *are also uniquely determined.*

Theorem 8 provides two canonical forms of regular circuits. These forms are represented schematically in Fugires 1 and 2.

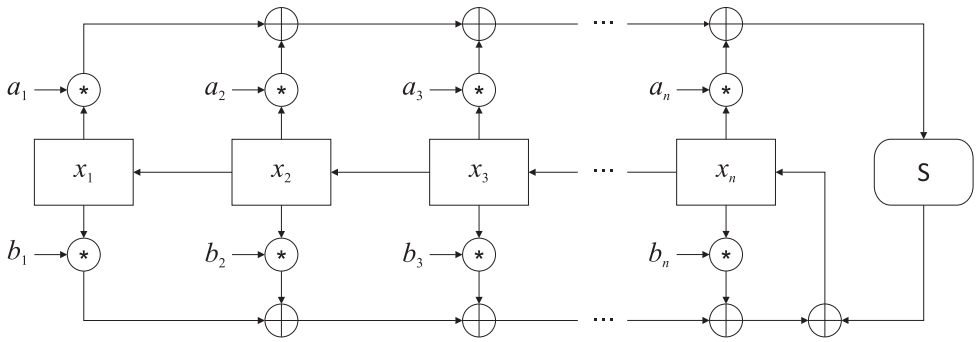



Figure 1: The first canonical form

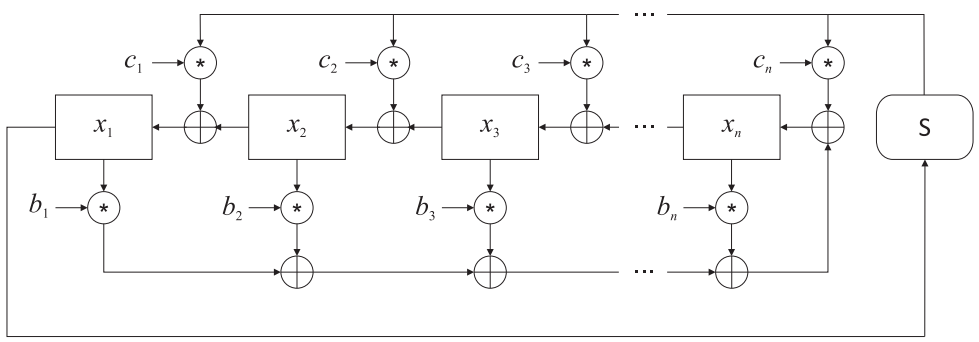



Figure 2: The second canonical form

**Corollary 3.** *The lag of a regular circuit does not exceed its dimension.*

*Proof.* Since similar circuits have the same lag, it is sufficient to consider a circuit of the first canonical form. If its lag is greater than its dimension $n$ then the first coordinates of $c, cB, \ldots, cB^{n-1}$ are zero. Therefore, $C$ is not invertible which contradicts regularity. □

Let us refine the vectors $a$ and $c$ which can appear in Theorem 8. The number of acceptable vectors $a$ (vectors $c$) is the number of equivalence classes of regular circuits with similar matrices $B$.

Identify vectors (row and column) with polynomials: both a vector $w = (w_1, w_2, \ldots, w_n)$ and its transpose are associated with the polynomial $w(\lambda) = w_1 + w_2\lambda + \ldots + w_n\lambda^{n-1}$. In particular, $f_B(\lambda) = \lambda^n + b(\lambda)$, where $b$ is the last column of $B$.

**Theorem 9.** *Let $(a, B, c)$ be an invertible circuit of dimension $n$ in which $B$ is a Frobenius cell. The circuit is regular if and only if both the polynomials $a(\lambda)$ and $(cP)(\lambda)$ are coprime with $f_B(\lambda)$. Here $P = (p_{ij})$, $1 \le i, j \le n$, where*

$$p_{ij} = \begin{cases} b_{i+j}, & i + j \le n, \\ 1, & i + j = n + 1, \\ 0, & i + j > n + 1. \end{cases}$$

*Proof.* Columns of $A$ are described by the polynomials

$$(B^i a)(\lambda) = \lambda^i a(\lambda) \bmod f_B(\lambda), \quad i = 0, 1, \ldots, n - 1.$$

The matrix $A$ is invertible if and only if any nonzero linear combination of its columns is nonzero. In other words, if and only if

$$g(\lambda)a(\lambda) \not\equiv 0 \pmod{f_B(\lambda)}$$

for any nonzero $g(\lambda) \in \mathbb{F}_2[\lambda]$, $\deg g \le n$. It is equivalent to coprimeness of $a(\lambda)$ and $f_B(\lambda)$.

In [18] it was proved that $B = PB^TP^{-1}$. From this fact, taking into account the symmetry of $P$, it follows that columns of $PC^T$ have the form

$$P(cB^i)^T = P(cP(B^T)^iP^{-1})^T = B^i(cP)^T.$$

Processing $PC^T$ in the same way as $A$, we conclude the proof. □

**Example 5.** *Let $n \ge 2$ and $f_B(\lambda)$ be irreducible. Then there are $2^{n-1} - 1$ equivalence classes of regular circuits which matrices are similar to $B$. Indeed, let such a circuit has the first canonical form. By Theorem 7, the circuit is invertible if and only if $c_n = b_2c_1 + b_3c_2 + \ldots + b_nc_{n-1}$. There are $2^{n-1} - 1$ acceptable nonzero $c$ and they all satisfy the conditions of Theorem 9.*

*It is interesting that if $f_B(\lambda)$ is irreducible then the set of normalizers of $B$ augmented with the zero matrix forms the field of order $2^n$. In particular, the sum of distinct normalizers is again a normalizer.*

**Example 6.** *Let $f_B(\lambda) = \lambda^n + 1$. This is the case of the Feistel, SMS4, MARS3 and GFN1 circuits. Normalizers of $B$ are all invertible circulants. In the most interesting case $n = 2^k$, acceptable $a$, $c$ are those that contain an*

*odd number of unities and both the number of normalizers and equivalence classes is* $2^{n-1}$.

*The SMS4 circuit has the first canonical form, MARS3 has the second one. These circuits are similar: MARS3 can be converted to SMS4 using the normalizer*

$$P = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}.$$

**Example 7.** *Let* $(a, B, c)$ *be a circuit of type II. Bring it to the second canonical form. For invertibility, it is sufficient and necessary that* $a(0) = 1$. *For regularity,* $a(\lambda)$ *must additionally be coprime to* $f_B(\lambda)$. *If* $f_B(\lambda) = \lambda^n$ *(SkipjackA, SkipjackG) then the last condition holds for every* $(a_2, \ldots, a_n)$ *and consequently there are* $2^{n-1}$ *equivalence classes.*

*It is interesting that SkipjackB does not belong to the equivalence class of SkipjackA because its polynomial* $f_B(\lambda) = \lambda^4 + \lambda \neq \lambda^4$.

## References


[1] Berger T.P., Minier M., Thomas G. Extended Generalized Feistel Networks Using Matrix Representation. In: Lange, T., Lauter, K., Lisoněk, P. (eds) Selected Areas in Cryptography – SAC 2013. SAC 2013. Lecture Notes in Computer Science, vol 8282. Springer, Berlin, Heidelberg (2013).

[2] Biham E., Biryukov A., Shamir A. Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. J. Cryptology, 18, 291-311 (2005).

[3] Blondeau C., Bogdanov A., Wang M. On the (In)Equivalence of Impossible Differential and Zero-Correlation Distinguishers for Feistel- and Skipjack-type Ciphers. In: Boureanu, I., Owesarski, P., Vaudenay, S. (eds) Applied Cryptography and Network Security. ACNS 2014. Lecture Notes in Computer Science, vol 8479. Springer, Cham (2014).

[4] Blondeau C., Wang M. Analysis of Impossible, Integral and Zero-Correlation Attacks on Type-II Generalized Feistel Networks using the Matrix Method. In: Leander, G. (eds) Fast Software Encryption. FSE 2015. Lecture Notes in Computer Science, vol 9054. Springer, Berlin, Heidelberg (2015).

[5] Burwick C., Coppersmith D., D'Avignon E., Gennaro R., Halevi S., Jutla C., Matyas Jr. S.M., O'Connor L., Peyravian M., Safford D. Zunic N. MARS: A Candidate Cipher for AES. In: AES – The First Advanced Encryption Standard Candidate Conference, Conference Proceedings (1998).

[6] Choy J., Chew G., Khoo K., Yap. H. Cryptographic properties and application of a Generalized Unbalanced Feistel Network structure. Cryptogr. Commun. (2011) 3: 141. https://doi.org/10.1007/s12095-011-0042-6.

[7] Choy J., Yap H. Impossible Boomerang Attack for Block Cipher Structures. In: Takagi, T., Mambo, M. (eds) Advances in Information and Computer Security. IWSEC 2009. Lecture Notes in Computer Science, vol 5824. Springer, Berlin, Heidelberg (2009).

[8] Diffie W., Ledin G. SMS4 Encryption Algorithm for Wireless Networks. Cryptology ePrint Archive, Report 2008/329. https://eprint.iacr.org/2008/329 (2008). Accessed 16 February 2018.

[9] Feistel H., Notz W.A., Smith J.L. Some cryptographic techniques for machine-to-machine data communications. Proceedings of IEEE 63, 1545-1554 (1975).

[10] Knudsen L.R. DEAL – A 128-bit Block Cipher. Technical report, NIST AES Proposal. urlhttp://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.7982 (1998). Accessed 16 February 2018.

[11] Lai X., Massey J.L. A proposal for a new block encryption standard. In: Damgärd I.B. (eds) Advances in Cryptology – EUROCRYPT'90. EUROCRYPT 1990. Lecture Notes in Computer Science, vol 473. Springer, Berlin, Heidelberg (1991).

[12] Lidl R., Niederraiter H. Finite Fields. Cambridge University Press (1997).

[13] Malyshev F.M. The duality of differential and linear methods in cryptography. Mat. Vopr. Kriptogr., Volume 5, Issue 3, 35–47 (2014).

[14] McGrew D.A., Viega J. The security and performance of the Galois / Counter Mode (GCM) of operation. In: Canteaut, A., Viswanathan, K. (eds) Progress in Cryptology – INDOCRYPT 2004.

INDOCRYPT 2004. Lecture Notes in Computer Science, vol 3348. Springer, Berlin, Heidelberg (2004).

[15] National Institute of Standards and Technology (NIST). Skipjack and KEA Algorithm Specifications. Version 2.0. https://csrc.nist.gov/CSRC/media//Projects/Cryptographic-Algorithm-Validation-Program/documents/skipjack/skipjack.pdf (1998). Accessed 16 February 2018.

[16] Luo Y., Wu Z., Lai X., Gong G. A unified method for finding impossible differentials of block cipher structures. Inform. Sci. 263, 211–220 (2014).

[17] Matsui M. New block encryption algorithm MISTY. In: Biham, E. (eds) Fast Software Encryption. FSE 1997. Lecture Notes in Computer Science, vol 1267. Springer, Berlin, Heidelberg (1997).

[18] Solomon L. Similarity of the companion matrix and its transpose, Linear Algebra and Its Appl. 302-303, 555-561 (1999).

[19] Sung J., Lee S., Lim J., Hong S., Park S. Provable Security for the Skipjack-like Structure against Differential Cryptanalysis and Linear Cryptanalysis. In: Okamoto, T. (eds) Advances in Cryptology – ASIACRYPT 2000. ASIACRYPT 2000. Lecture Notes in Computer Science, vol 1976. Springer, Berlin, Heidelberg (2000).

[20] Yap H. Impossible Differential Characteristics of Extended Feistel Networks with Provable Security against Differential Cryptanalysis. In: Kim, H., Kim, T., Kiumi, A. (eds) Advances in Security Technology. SecTech 2008. Communications in Computer and Information Science, vol 29. Springer, Berlin, Heidelberg (2009).

[21] Zheng Y., Matsumoto T., Imai H. On the Construction of Block Ciphers Provably Secure and Not Relying on Any Unproved Hypotheses. In: Brassard, G. (eds) Advances in Cryptology – CRYPTO'89. CRYPTO 1989. Lecture Notes in Computer Science, vol 435. Springer, New York, NY (1990).

# On Some Properties of an XSL-network

Alexey Kurochkin

Technical Committee for Standardization
«Cryptography and security mechanisms» (TC 26), Moscow, Russia
playfootball77@mail.ru

**Abstract**

A new type of distinguishing property, named the zero-sum property has been recently presented by Aumasson and Meier [1]. It has been applied to the inner permutation of the hash function Keccak and it has led to a distinguishing property for the Keccak-f permutation up to 16 rounds (the number of rounds initially was 18). Later the number of rounds was increased to 24, due to the results of the work [1]. In this paper, we presented a modified algorithm for finding zero-sums for hash-functions having an XSL structure, with some restrictions on the linear transformation. As an example, we apply the algorithm to a representative of a family of *Photon* hash functions [2].

**Keywords:** zero-sum, algebraic degree, Photon, hash-function.

## 1 Zero-sum

In this section, we introduce the notion of zero-sums, demonstrate the algorithm for finding it, and demonstrate the modification of this algorithm. As an example, we apply it to the hash function $Photon - 196$.

### 1.1 Intoduction

**Definition 1.** *[1] Let $F$ be a function from $F_2^n$ into $F_2^m$. A zero-sum for $F$ of size (dimension) $K$ is a subset $\{x_1, \ldots, x_K\} \subset F_2^n$ of elements which sum to zero and for which the corresponding images by $F$ also sum to zero, i.e.*

$$\bigoplus_{i=1}^{K} x_i = \bigoplus_{i=1}^{K} F(x_i) = 0.$$

*where the sum is defined by the addition in $F_2^n$ (and in $F_2^m$), i.e., the bitwise exclusive-or. Since it is expected that a randomly chosen function does not have many zero-sums, the existence of several such sets of inputs can be seen as a distinguishing property of $F$.*

**Definition 2.** *Let $K \subset F_2^n$ be a subspace. Denote $\underline{K} = \{K \oplus a | a \in F_2^n$. That is $\underline{K}$ the set of all coset classes by $K$.*

**Statement 1.** *[1] Let $F$ be a function from $F_2^n$ into $F_2^m$ and $V$ the subspace of dimension $(degF) + 1$. Then, for every $V^{'} \in \underline{V}$ is true:*

$$\bigoplus_{v \in V'} F(x \oplus v) = 0.$$

The fact that the permutation used in a hash function does not depend on any secret parameter allows to exploit the previous property starting from the middle, i.e., from an intermediate internal state. This property was used by Aumasson and Meier [1] and also by Knudsen and Rijmen in the case of a known-key property of a block cipher [3]. The only information needed for finding such zero-sums on the iterated permutation is an upper bound on the algebraic degrees of both the round transformation and its inverse. The upper bound for the algebraic degree of nonlinearity of the iterative mapping it is calculated, as a rule, as in [4]. More precisely, we suppose that $F$ is a function which operates on an $n - bit$ state, and that $F$ is composed of $n_r$ transformations:

$$F = R_{n_r} \circ \ldots \circ R_1.$$

Let $d_1 < n$ be the degree of the function composed of the last $r_1$ transformations, i.e., $F_{r_1} = R_{n_r} \circ \ldots \circ R_{n_r - r_1 + 1}$ and let $d_2 < n$ be the degree of the inverse of the first $r_2 = (n_r - r_1)$ transformations, i.e., $G_{r_2} = R_1^{-1} \circ \ldots \circ R_{r_2}^{-1}$. Then, we can find many zero-sums of size $2^{d+1}$ where $d = max(d_1, d_2)$ as follows:

1. Choose a set of $(n-d-1)$ bits in the intermediate state after $r_2$ rounds, and fix them to an arbitrary value;
2. For each of the $2^{d+1}$ possible intermediate states $z$ obtained when the other $(d+1)$ bits take all possible values, compute $r_2$ rounds backwards in order to obtain the $2^{d+1}$ input states $x = G_{r_2}(z)$.

The set of these input states is then the zero-sum of size $2^{d+1}$ of a function of degree $d_2$ and thus it vanishes. Now, the images of these input states under F correspond to the images of the intermediate states $z$ under $F_{r_1}$. Then, by computing $r_1$ rounds forwards, we obtain $2^{d+1}$ output states. The set of these output states is the zero-sum of size $2^{d+1}$ of $F_{r_1}$, of degree less than d. Thus, this sum vanishes, implying that the $x$ forms a zero-sum. It is worth

noticing that this technique provides several zero-sums having a particular property. Actually, for a given choice of the $(n-d-1)$ fixed bits in the intermediate state, taking all possible values for the corresponding constant leads to $2^{n-d-1}$ zero-sums of sizes $2^{d+1}$ which form a partition of the input space into zero-sums.

### 1.2 Modified algorithm

In this subsection we introduce new definitions and statement 1 which allows us to modify the algorithm for finding zero-sum.

**Definition 3.** *Let* $a = (a_0, a_1, \ldots, a_{n-1}) \in F_2^n$, *and* $\{n_1, n_2, \ldots, n_t\}$ *(where* $\sum_{i=1}^{t} n_i = n$*), be a fixed set of natural numbers which splits the vector* $a$ *into consecutive subvectors i.e.* $a = (a_0, a_1, \ldots, a_{n_1-1})||(a_{n_1}, \ldots, a_{n_2-1})|| \ldots ||(a_{n_t}, \ldots, a_{n-1})$, *we shall call a partition of the form* $(n_1, \ldots, n_t)$, *and if all subvectors are of equal length* $d$ *we denote that partition by* $(d)$. *If the partition refers to a particular vector, then we write* $(a_{n_1} \ldots, a_{n_t})$ *or* $a(d)$ *respectively.*

**Definition 4.** *Let* $V_1 \in F_2^n$ *and* $V_2 \in F_2^n$ *are subspaces. We say that* $H : F_2^n \to F_2^n$ *keeps the structure of the subspace* $V_1$ *if for any* $V_1^{'} \in \underline{V_1}$ *there is a* $V_2^{'} \in \underline{V_2}$ *such that:*

$$H(V_1^{'}) = V_2^{'} \text{ and denote } H(V_1 \to V_2).$$

**Definition 5.** *We say that a subspace* $V = V_1$ *of block type* $i_1, i_2, \ldots, i_t$ *is consistent with the partition* $(n_1, \ldots, n_t)$, *if* $V = \{(a_{n_1}, a_{n_2}, \ldots, a_{n_t})\}$, *where if* $i_j = 1$ *then* $a_{n_j}$ *takes all possible values from* $F_2^{n_j}$, *and if* $i_j = 0$ *then* $a_{n_j} = 0$. *Denote such subspaces as* $V^{(n_1,\ldots,n_t)}(i_1, \ldots, i_t)$ *and if the partition has the form (d), then* $V^d(i_1, \ldots, i_t)$.

**Definition 6.** *We say that a transformation* $G : F_2^n \to F_2^n$ *has a block structure if there exists at least one* $l$ *subspace of block type* $V$ *consistent to the partition* $(n_1, \ldots, n_t)$ *such that:*

$G(V \to V^{'})$, *where* $V^{'}$ *is a subspace of block type consistent to the partition* $(m_1, \ldots, m_l)$.

**Statement 2.** *Let* $G : F_2^n \to F_2^n$ *be an iterative transformation of the form:*

$$G = G_1 \circ G_2 \circ \ldots \circ G_{2 \cdot t}, \text{ where } G_i = X_i \circ S \circ L, \; i = \overline{1, 2 \cdot t}.$$

*Note that for calculation of the zero-sum, the values of the constant do not matter. Therefore, let* $X_i = X, i = \overline{1, 2 \cdot t}$. *Then if* $V \in F_2^n$ *is a subspace such that:* $S \circ L \circ X \circ S(V \to V^{'})$ *then in order to find the zero-sum it is necessary to construct a subspace of smaller dimension.*

Let us show this. Denote by $nl$ an algebraic degree of the function. Let:

$$\begin{gathered} h_1 = nl(G_1 \circ \ldots \circ G_t), \\ h_2 = nl(G_{t+1} \circ \ldots \circ G_{2 \cdot t}), \\ h_1^{'} = nl(G_1 \circ \ldots \circ G_{t-1}), \\ h_2^{'} = nl(G_{t+2} \circ \ldots \circ G_{2 \cdot t}), \\ d = max(h_1, h_2), \\ d^{'} = max(h_1^{'}, h_2^{'}). \end{gathered}$$

Using the algorithm for finding zero-sums from Section 1, it is easy to see, that the complexity of constructing the zero-sum became $2^{d^{'}}$, instead of $2^d$.

**Remark 1.** *For several cryptoalgorithms one can easily show that* $S \circ L \circ X \circ S$ *transformation is a block type. For example:* $Photon, Streebog, Stribob, AES$. *Examples where it is not easy to show:* $Kuznyechik, Present$.

### 1.3 Sponge construction

Extended Sponge functions. Sponge functions have been introduced by Bertoni et al. [5] as a new way for building hash functions from a fixed permutation. The internal state $S$ of $t$ bits, composed of the $c - bit$ capacity and the $r - bit$ bitrate ($t = c + r$), is, first, initialized with some fixed value. Then, after being appropriately padded and split the message into $r - bit$ chunks, one simply and iteratively processes all $r - bit$ message chunks by xoring them to the bitrate part of the internal state and then applying the $t - bit$ permutation $P$. Once all message chunks have been handled by this absorbing phase, one successively outputs $r$ bits of the final hash value by extracting r bits from the bitrate part of the internal state and then applying the permutation $P$ on it (squeezing process).

### 1.4 *Photon*

*Photon*, a family of sponge-like hash function proposals that was recently standardized by ISO. Authors define an AES-like function to be a fixed key permutation $P$ applied on an internal state of $d^2$ elements of $s$ bits each, which can be represented as a $(d \times d)$ matrix. $P$ is composed of 12 rounds,

each containing four layers as depicted in Figure 1: AddConstants (Add), SubCells (S), ShiftRows (Row), and MixColumnsSerial(Mix).

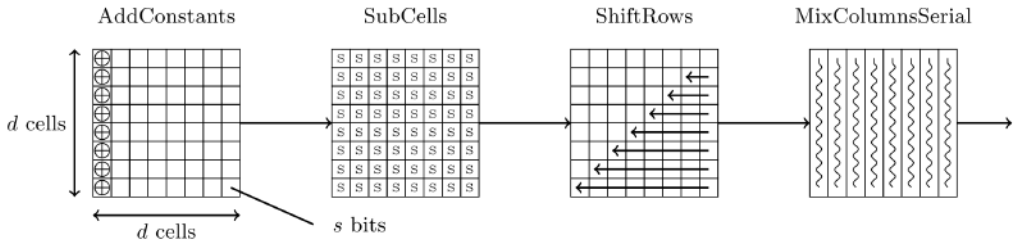



Figure 1.

Table 1 shows the main parameters of the *Photon* hash function family. Table 2 shows the degree of nonlinearity, depending on the number of rounds.

| | $t$ | $d$ | $s$ | $N_r$ | $IC_d(\cdot)$ | irr. polynomial | $Z_i$ coefficients |
|---|---|---|---|---|---|---|---|
| $P_{100}$ | 100 | 5 | 4 | 12 | $[0,1,3,6,4]$ | $x^4+x+1$ | $(1,2,9,9,2)$ |
| $P_{144}$ | 144 | 6 | 4 | 12 | $[0,1,3,7,6,4]$ | $x^4+x+1$ | $(1,2,8,5,8,2)$ |
| $P_{196}$ | 196 | 7 | 4 | 12 | $[0,1,2,5,3,6,4]$ | $x^4+x+1$ | $(1,4,6,1,1,6,4)$ |
| $P_{256}$ | 256 | 8 | 4 | 12 | $[0,1,3,7,15,14,12,8]$ | $x^4+x+1$ | $(2,4,2,11,2,8,5,6)$ |
| $P_{288}$ | 288 | 6 | 8 | 12 | $[0,1,3,7,6,4]$ | $x^8+x^4+x^3+x+1$ | $(2,3,1,2,1,4)$ |

Table 1.

| number of rounds | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| $P_{100}$ | 3 | 9 | 27 | 75 | 91 | 97 | 99 | 99 | 99 |
| $P_{144}$ | 3 | 9 | 27 | 81 | 123 | 137 | 141 | 143 | 143 |
| $P_{196}$ | 3 | 9 | 27 | 81 | 157 | 183 | 191 | 194 | 195 |
| $P_{256}$ | 3 | 9 | 27 | 81 | 197 | 236 | 249 | 253 | 255 |
| $P_{288}$ | 7 | 42 | 252 | 282 | 287 | 287 | 287 | 287 | 287 |

Table 2.

Transformation P:

$$P = G_1 \circ \ldots \circ G_5 \circ Add_6 \circ S \circ Row \circ Mix \circ Add_7 \circ S \circ Row \circ Mix \circ G_8 \circ \ldots \circ G_{12}.$$

It is easy to show that substutition $H = S \circ Row \circ Mix \circ Add_7 \circ S$ has a block structure such that:

$$H(V_1 \to V_2).$$
$V_1 = (V^4(i_0, \ldots, i_{48})$, where $i_j = 1$, if $(j = 0 \mod (8))$, $i_j = 0$, if $(j \neq 0 \mod (8))$ $j = \overline{0, 48}$.
$V_2 = (V^4(i_0, \ldots, i_{48})$, where $i_j = 1$, if $(j = 0 \mod (7))$, $i_j = 0$, if $(j \neq 0 \mod (7))$ $j = \overline{0, 48}$.

According to Table 2, the degree of nonlinearity $nl(G_1 \circ \ldots \circ G_6) = nl(G_7 \circ \ldots \circ G_{12}) = 183$, and $nl(G_1 \circ \ldots \circ G_5) = nl(G_8 \circ \ldots \circ G_{12}) = 157$.

It's obvious to see that (Figure 2.):

$$H(V_1 \to V_2).$$

$V_1 = (V^4(i_0, \ldots, i_{48})$, where $i_j = 0$, if $(j = 0 \mod (8))$, $i_j = 1$, if $(j \neq 0 \mod (8))$ $j = \overline{0, 48}$.
$V_2 = (V^4(i_0, \ldots, i_{48})$, where $i_j = 0$, if $(j = 0 \mod (7))$, $i_j = 1$, if $(j \neq 0 \mod (7))$ $j = \overline{0, 48}$.

In Figure 2, white color indicates 4-bit words which take all possible values, and black color 4-bit words whose values are fixed.

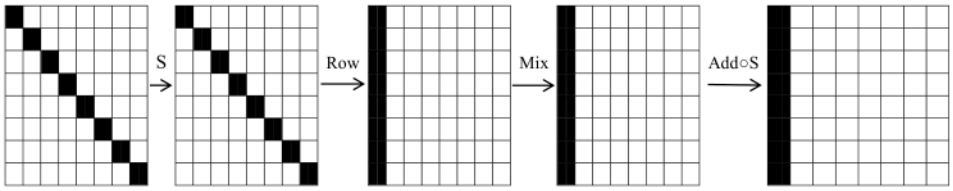



Figure 2.

Using statement 1 and the fact that the dimension of the subspace $V_1$ is $196 - 4 \cdot 7 = 168$, one can assert that we have found zero-sums of dimension 168 instead of the declared 183.

Also for the hash function $Photon(256)$ using this algorithm, the complexity of finding zero sums can be reduced from $2^{236}$ to $2^{224}$.

*We note that similar results for hash functions were obtained in [6] and [7]. The results of this paper were obtained independently of them.*

## 2 Conclusions

A modified algorithm for constructing zero sums was proposed in this paper. The algorithm is applicable to cryptographic algorithms of a certain type. Note that these results can be applied to the synthesis of cryptographic primitives.

*Also note that these results can be applied to the analysis of block ciphers, with the only difference being that the construction of zero sums can not be started from the intermediate states of the algorithm.*

## References

[1] J.-P. Aumasson and W. Meier. Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi. Presented at the rump session of Cryptographic Hardware and Embedded System – CHES 2009.

[2] J. Guo, T Peyrin, and Axel Poschmann. The PHOTON Family of Lighrweight Hash Function. 2011.

[3] J. Daemen, L.R. Knudsen and V. Rijmen. The block cipher Square. In Fast Sofrware Encryption – FSE'94, volume 1267 of Lecture Notes in Computer Science, pages 149-165. Springer-Verlag. 1997.

[4] Christina Boura, Anne Canteaut, and Christophe De Canniere. Higher-order differential properties of Keccak and Luffa. In FSE'11, volume 6733 of Lecture Notes in Computer Science, pages 252-269. Springer, 2011.

[5] Guido Beroni, Joan Daemen, Michael Peeters, and Gilles Van Assche. Sponge function. Ecrypt Hash Workshop 2007.

[6] Qinhju Wang, Lorenzo Grassi and Christian Rechberger. Zero-Sum Partitions of PHOTON Permutations. Technical University of Denmark.

[7] Novel method of constructing the zero-sum distinguishers. DONG Le, WU Wen-ling, WU Shuang, ZHOU Jian. Institute of Software, Chinese Academy of Sciences, Beijing 100190, China 2012.

# Exact Maximum Expected Differential and Linear Probability for 2-round Kuznyechik

Vitaly Kiryukhin

JSC «InfoTeCS», Moscow, Russia
v.a.kir@yandex.ru

**Abstract**

This paper presents the complete description of the best differentials and linear hulls in 2-round Kuznyechik. A comparison is made with similar results for the AES cipher.

**Keywords:** Kuznyechik, LSX, MDS codes, differential cryptanalysis, linear cryptanalysis.

## 1 Introduction

This paper presents the results of the development of low-complexity algorithms, that will allow to find the complete description of the best differential trails, differentials, linear characteristics, linear hulls and *exact* values of maximum expected differential and linear probability (MEDP, MELP) for 2-round Kuznyechik.

We proved that 2-round MEDP $= 2^{-86.66...}$, MELP $= 2^{-76.936...}$.

A comparison is made with similar cryptanalysis results for the AES cipher [1].

The main focus will be on the differential method. The results of the search for linear characteristics will be obtained in a similar way, due to the existence well-known duality between differential cryptanalysis and linear cryptanalysis [2].

## 2 Basic information

Kuznyechik block cipher [3] consists of a sequence of 9 rounds and a post-whitening key addition. Each round contains three operations:

$X$ – modulo 2 addition of an input block with an iterative key;

$S$ – parallel application of a fixed bijective substitution to each byte of the block;

$L$ – linear transformation which is defined as a LFSR over $GF(2^8)$. It can be represented as multiplication by the matrix $\mathbb{L}$ over $GF(2^8)$.

The block size is 128 bits ($n = 16$ bytes).

A 2-round differential trail can be represented as the following scheme:

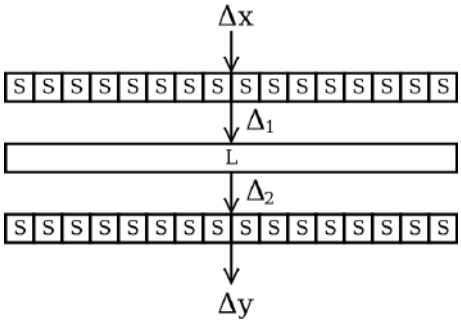



Figure 1: 2-round differential trail

$\Delta x = (x_1, ..., x_n)$ – the difference of input blocks in byte representation,

$\Delta_1 = (\alpha_1, ..., \alpha_n)$ – the difference of blocks after the nonlinear transformation on the first round,

$\Delta_2 = (\beta_1, ..., \beta_n) = (\alpha_1, ..., \alpha_n)\mathbb{L}$ – the difference of blocks after the linear transformation (matrix multiplication in row-by-row representation),

$\Delta y = (y_1, ..., y_n)$ – the difference of blocks after the nonlinear transformation on the second round.

Note that due to «linearity» and «invertibility» the linear transformation on the second round can be omitted without loss of generality.

The nonlinear transformation of each S-box is characterized by a matrix of transition probabilities (Differential Distribution Table). DDT is the set of local difference characteristics:

$$P(\alpha \to \beta) = \Pr(S(\chi \oplus \alpha) \oplus S(\chi) = \beta),\ \alpha, \beta, \chi \in \{0,1\}^8, \qquad (1)$$

where $\chi$ is a uniformly distributed random variable. S-box with nonzero input difference $\alpha \neq 0$ is called active.

2-round differential trail $\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y$ is a random variable, that has a probability (*EDCP* [1])

$$P(\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y) = \left(\prod_{i=1}^{n} P(x_i \to \alpha_i)\right)\left(\prod_{i=1}^{n} P(\beta_i \to y_i)\right). \quad (2)$$

The best differential trail has probability

$$P_{best}^{trail} = P_{best}(\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y) =$$
$$= \max_{(\Delta x, \Delta_1, \Delta_2, \Delta y) \setminus (0,0,0,0)} P(\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y).$$

Differential is the set of all differential trails that have the same $\Delta x$ and $\Delta y$.

Differential is characterized by the probability (*EDP* [1])

$$P(\Delta x \to \Delta y) = \sum_{i=1}^{T} \left( \left( \prod_{j=1}^{n} P(x_j \to \alpha_j^{(i)}) \right) \left( \prod_{j=1}^{n} P(\beta_j^{(i)} \to y_j) \right) \right), \quad (3)$$

where $T$ is the number of the differential trails in the differential.

The best differential has probability (*MEDP* [1]):

$$P_{best}^{diff} = P_{best}(\Delta x \to \Delta y) = \max_{(\Delta x, \Delta y) \setminus (0,0)} P(\Delta x \to \Delta y)$$

Our first goal is to find the most probable differential trail – the best differential trail.

Matrix $\mathbb{L}$ is part of the matrix $\mathbb{G} = \mathbb{E}|\mathbb{L}$. $\mathbb{G}$ is the generator matrix of the MDS-code $(32, 16, 17)$ over $GF(2^8)$. Thus, the minimum possible total weight of vectors $\Delta_1$ and $\Delta_2$ is equal to the minimum code distance $d = 17$. We will start searching for the most probable differential trail by finding all minimum byte weight codewords in $\mathbb{G}$.

## 3 Algorithm for finding codewords with the smallest byte weight

Let $(t, r)$ such, that $t+r = n+1$, $t > 0$, $r > 0$. Fix $k_1, \ldots, k_t, m_1, \ldots, m_r$ – locations of non-zero elements in the vectors $\Delta_1 = (\alpha_1, ..., \alpha_n)$ and $\Delta_2 = (\beta_1, ..., \beta_n)$ accordingly. Let's present the transformation $\Delta_1 \mathbb{L} = \Delta_2$ as a system of equations. Select the subsystem $\mathbb{S}_{n-r,t}$ in the system $\Delta_1 \mathbb{L} = \Delta_2$: $(\alpha_{k_1}, \ldots, \alpha_{k_t}) \cdot \mathbb{S}_{n-r,t} = (\underbrace{0, \ldots, 0}_{n-r})$. Solve the subsystem $\mathbb{S}_{n-r,t}$ . The set of solutions is $(\alpha_{k_1}^{(i)}, \ldots, \alpha_{k_t}^{(i)})$, $i = \overline{1, 255}$. Hence we have the set of $\Delta_1^{(i)}$ and the set of $\Delta_2^{(i)} = \Delta_1^{(i)} \mathbb{L}$ , $i = \overline{1, 255}$.

Let's denote these sets of solutions

$$M^{(n+1)}(k_1, \ldots, k_t, m_1, \ldots, m_r) = (\alpha_{k_1}^{(i)}, \ldots, \alpha_{k_t}^{(i)}, \beta_{m_1}^{(i)}, \ldots, \beta_{m_r}^{(i)}), \ i = \overline{1, 255}. \quad (4)$$

The union of such sets is the set

$$M^{(n+1)} = \bigcup_{(k_1, \ldots, k_t, m_1, \ldots, m_r)} M^{(n+1)}(k_1, \ldots, k_t, m_1, \ldots, m_r)$$

of all code vectors of minimum weight $n + 1$. The cardinality of the set $M^{(n+1)}$ is equal to $255 \cdot \sum_{(t,r):t+r=n+1} \binom{n}{t}\binom{n}{r} = 255 \cdot \binom{2n}{n+1}$. Note, that the

same expression for the number of codewords of minimal weight is obtained in [5] without constructing an algorithm for their search.

Pseudocode of the algorithm is presented in Appendix E.

## 4 Algorithm for finding the best differential trail

In general, we consider differential trails for 2 rounds

$$\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y.$$

We start with differential trails containing the minimum number of active S-boxes (minimal weight of $\Delta_1$ and $\Delta_2$).

To simplify the notation we denote $(\Delta_1, \Delta_2) = (\alpha_{k_1}, \ldots, \alpha_{k_t}, \beta_{m_1}, \ldots, \beta_{m_r})$, $t + r \geq d = 17$. Coordinates equal to zero are omitted in notation.

$$P_{max}(\Delta_1, \Delta_2) = \left(\prod_{j=1}^{t} \max_x P(x \to \alpha_{k_j})\right) \left(\prod_{j=1}^{r} \max_y P(\beta_{m_j} \to y)\right)$$

is the maximum probability of differential trail with a fixed vector $(\Delta_1, \Delta_2)$. Then the most probable differential trail $\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y$ has the probability:

$$P_{best}^{trail} = \max_{(\Delta_1, \Delta_2) \setminus (0,0)} P_{max}(\Delta_1, \Delta_2).$$

Let the vector $(\Delta_1, \Delta_2)$ has a weight $n + 1$:

$$P_{best}^{trail} \geq \max_{(\Delta_1, \Delta_2) \in M^{(n+1)}} P_{max}(\Delta_1, \Delta_2).$$

Two sets of differential trails were found in $M^{(n+1)}$. Each trail in both sets has a maximum probability:

$$\max_{(\Delta_1, \Delta_2) \in M^{(n+1)}} P_{max}(\Delta_1, \Delta_2) = \left(\frac{8}{256}\right)^{13} \left(\frac{6}{256}\right)^{4} = 2^{-86.66\ldots}.$$

The trails in the set have the same inner part $(\Delta_1, \Delta_2)$. There are no other trails that would have a maximum probability.

The found differential trails are presented in Appendix A.

**Lemma 1.** *Let $\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y$ be the differential trail in 2-round Kuznyechik. Let $P(\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y)$ be maximal among all trails. Then the weight $(\Delta_1, \Delta_2)$ is equal to $n + 1 = 17$.*

*Proof.* One can see that the estimate

$$P(\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y) \le \left( \max_{(\alpha,\beta)\setminus(0,0)} P(\alpha \to \beta) \right)^w. \qquad (5)$$

is true for any differential trail $\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y$, $\|\Delta_1\| + \|\Delta_2\| = w = t + r$.

In the case of Kuznyechik, $\max\limits_{(\alpha,\beta)\setminus(0,0)} P(\alpha \to \beta) = \left(\frac{8}{256}\right)$. Then for any $w \ge 18$ it holds that:

$$P(\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y) \le \left(\frac{8}{256}\right)^w \le$$

$$\le \left(\frac{8}{256}\right)^{18} < \max_{(\Delta_1,\Delta_2)\in M^{(n+1)}} P_{max}(\Delta_1, \Delta_2) = 2^{-86.66\ldots}.$$

Hence $P_{best}^{trail} = 2^{-86.66\ldots}$. Lemma 1 is proved. □

## 5 Algorithm for finding the best differential

Suppose that the best differential will also be achieved on a configuration containing the minimum number $w = n + 1 = 17$ of active S-boxes.

Each subset $M^{(n+1)}(k_1, \ldots, k_t, m_1, \ldots, m_r)$ contains exactly 255 code vectors. The sets $k_1, \ldots, k_t$ and $m_1, \ldots, m_r$ specify the positions of active S-boxes. Hence the differential $\Delta x \to \Delta y$ contains trails from *only one* subset $M^{(n+1)}(k_1, \ldots, k_t, m_1, \ldots, m_r)$. Consequently, in expression (3) $T = 255$.

Consider an algorithm that allows you to get rid of the exhaustive search. It is based on the «pruning» of the branches of the search tree by using the constructed upper bounds.

In the previous paragraph, the exact value of the best differential trail is given $P_{best}^{trail} = 2^{-86.66\ldots}$. This probability is the lower bound for the probability of the best differential. It is always possible to construct a differential, consisting of one best trail $P_{best}^{diff} \ge P_{best}^{trail}$. We will use the probability $P_{est}^{diff} = P_{best}^{trail}$ as a threshold value.

### 5.1 Algorithm for calculating the upper bound of the differential

Let a subset of codewords (4) is given. Calculate the upper bound of the differential.

Fix $u \le t$, $v \le r$. Select $t - u$ coordinates $\alpha$ and $r - v$ coordinates $\beta$ in

the equation (4):

$$part^{(i)} = (\alpha_{k_1}^{(i)}, \ldots, \alpha_{k_{t-u}}^{(i)}, \beta_{m_1}^{(i)}, \ldots, \beta_{m_{r-v}}^{(i)}),\ i = \overline{1, 255}.$$

For all $i = \overline{1, 255}$ we obtain an easily computable upper bound for the «part» of the differential trail

$$P(\Delta x \to part^{(i)} \to \Delta y) \leq$$
$$\leq \left(\prod_{j=1}^{t-u} \max_x P(x \to \alpha_{k_j}^{(i)})\right) \left(\prod_{j=1}^{r-v} \max_y P(\beta_{m_j}^{(i)} \to y)\right).$$

Let's order these estimates in descending order.

We will construct for each $x$ (and $y$) the sequence of transition probabilities. Let's use the S-box transition probability matrix (DDT):

$$P(x \to \alpha^{(1,x)}) \geq P(x \to \alpha^{(2,x)}) \geq \ldots \geq P(x \to \alpha^{(255,x)}),\ x = \overline{1, 255}, \quad (6)$$

$$P(\beta^{(1,y)} \to y) \geq P(\beta^{(2,y)} \to y) \geq \ldots \geq P(\beta^{(255,y)} \to y),\ y = \overline{1, 255}. \quad (7)$$

$$X^{(q)} = \max_x P(x \to \alpha^{(q,x)}),\ Y^{(q)} = \max_y P(\beta^{(q,y)} \to y). \quad (8)$$

Consider the differential (3). Let the summands be ordered in descending order. Then

$$P(\Delta x \to \Delta y) \leq \min_{u,v} \left(\sum_{q=1}^{255} \left(X^{(q)}\right)^u \left(Y^{(q)}\right)^v \left(P(\Delta x \to part^{(q)} \to \Delta y)\right)\right). \quad (9)$$

If the resulting upper bound (9) is less than the threshold $P_{est}^{diff}$, then the subset is no longer considered.

In practice, the values $u$ and $v$ are selected experimentally depending on the cipher substitution. For Kuznyechik $u = v = 2$ are close to optimal parameters. For such values, approximately $\frac{9}{10}$ subsets are excluded from being considered.

### 5.2 Algorithm for constructing the differential

Suppose that for some subset $M^{(n+1)}(k_1, \ldots, k_t, m_1, \ldots, m_r)$ the estimate is greater than the threshold value $P_{est}^{diff}$. Then the following estimate also holds

$$P(\Delta x \to \Delta y) \leq \sum_{i=1}^{255} \left(\prod_{j=1}^{t} \max_x P(x \to \alpha_{k_j}^{(i)}) \prod_{j=1}^{r} \max_y P(\beta_{m_j}^{(i)} \to y)\right). \quad (10)$$

We will sequentially search through possible non-zero values $x_{k_1}, \ldots, x_{k_t}$ and $y_{m_1}, \ldots, y_{m_r}$. The maximum values $\max\limits_x P(x \to \alpha_{k_j}^{(i)})$ (and $\max\limits_y P(\beta_{m_j}^{(i)} \to y)$) will be replaced by the immediate values $P(x_{k_j} \to \alpha_{k_j}^{(i)})$ ($P(\beta_{m_j}^{(i)} \to y_{m_j})$ accordingly). We will also use the pruning of the branches of the search tree.

Denote

$$P(a_1, a_2, \ldots a_s) = P(x_{k_1} = a_1, x_{k_2} = a_2, \ldots, x_{k_s} = a_s, x_{k_{s+1}}, \ldots, x_{k_t} \to \Delta y),$$

$$P(a_1, a_2, \ldots a_s) \leq$$

$$\leq \sum_{i=1}^{255} \left( \prod_{j=1}^{s} P(a_j \to \alpha_{k_j}^{(i)}) \prod_{j=s+1}^{t} \max_x P(x \to \alpha_{k_j}^{(i)}) \prod_{j=1}^{r} \max_y P(\beta_{m_j}^{(i)} \to y) \right).$$

In the estimate (10), we fix the first factor with the number $k_1$(the place of the first nonzero element). Let $x_{k_1} = 1$. Then we replace $\max\limits_x P(x \to \alpha_{k_1}^{(i)})$ by $P(1 \to \alpha_{k_1}^{(i)})$. After that we have the estimate $P(a_1 = 1)$. If the estimate $P(a_1 = 1)$ is less than the threshold value $P_{est}^{diff}$, then we perform a search among the elements $x_{k_1} = 2, 3, ...255$. We will search until the element $x_{k_1} = a_1$, $P(a_1) \geq P_{est}^{diff}$ is found. If such $x_{k_1}$ is not found, then the subset $M^{(n+1)}(k_1, \ldots, k_t, m_1, \ldots, m_r)$ is excluded from being considered.

Let such $x_{k_1} = a_1$ is found. We perform similarly search of the second factor. Consider the bytes $x_{k_2} = 1, 2, \ldots, a_2, \ldots, 255$. Substituting $P(a_2 \to \alpha_{k_2}^{(i)})$ instead of $\max\limits_x P(x \to \alpha_{k_2}^{(i)})$ into the estimate $P(a_1)$. Do this until $a_2$ : $P(a_1, a_2) \geq P_{est}^{diff}$ is found. If such an element is not found then return to the previous step and try to accomplish this algorithm for the remaining bytes $x_{k_1} > a_1$.

We continue the recursive search. We replace the «s+1»-th factor in $P(a_1, a_2, \ldots a_s)$ with the value $P(a \to \alpha_{k_{s+1}}^{(i)})$, $a = 1, 2, \ldots, 255$. Multipliers $\max\limits_y P(\beta_{m_j}^{(i)} \to y)$ are replaced by values $P(\beta_{m_j}^{(i)} \to b)$, $b = 1, 2, \ldots, 255$.

If the algorithm substituted all the elements $a_1, \ldots, a_t, b_1, \ldots, b_r$ and did not reject the subset of codewords, then we obtained an exact estimate $P(a_1, \ldots, a_t \to b_1, \ldots, b_r)$ and the differential

$$P(\Delta x \to \Delta y) = \sum_{i=1}^{255} \left( \prod_{j=1}^{t} P(a_j \to \alpha_j^{(i)}) \right) \left( \prod_{j=1}^{r} P(\beta_j^{(i)} \to b_j) \right) \geq P_{est}^{diff}. \tag{11}$$

In this case, the value $P_{est}^{diff}$ is updated. We return to the previous step of the algorithm and continue the search in the subset

$M^{(n+1)}(k_1, \ldots, k_t, m_1, \ldots, m_r)$.

The last step of the algorithm: $P_{best}^{diff} = P_{est}^{diff}$ .

It was shown that if the number of active substitutions is $n + 1 = 17$, then each best differential contains only one differential trail.

The best differential trails are presented in Appendix A. Pseudocodes of algorithms are presented in Appendix E.

**Lemma 2.** *Let $\Delta x \to \Delta y$ is the differential in 2-round Kuznyechik. Let $P(\Delta x \to \Delta y)$ be maximal among all differentials. Then the number of active S-boxes in $\Delta x \to \Delta y$ is equal to $n + 1 = 17$.*

The main idea of the proof is to construct an upper bound for the differential $\Delta x \to \Delta y$ containing $n + 2 = d + 1 = 18$ active S-boxes. The upper estimate is built by using: two majorants (8); the MDS code property (byte weight of the sum of codewords is not less than $n + 1$); the rearrangement inequality [6]. The proof of the Lemma is presented in Appendix D.

## 6 The comparison with AES

The comparison of the results given in this paper for Kuznyechik with the results of the AES cipher analysis is of particular interest [1].

Note the following differences between 2-round versions of the ciphers [3, 4].

**Kuznyechik** – one MDS-matrix $16 \times 16$; pseudorandom, non-analytical S-box; DDT and LAT do not have obvious patterns.

**AES** – byte permutation layer and four MDS-matrix $4 \times 4$; all nontrivial rows and columns in DDT (and LAT) have the same distribution of values.

Differences in linear and non-linear transformations lead to different approaches for calculating differential and linear characteristics.

In the case of AES the actual work is reduced to a single MDS-matrix $4 \times 4$. This allows you to construct the entire set of codewords. In the case of Kuznyechik, due to the use of the algorithm (3), only low-weight codewords are iterated over. After that, it is analytically shown that the differential on codewords of greater weight will be worse than the constructed one.

The best differential in AES consists of 75 differential trails. The estimate (6) is used in the construction of the differential. The estimate (10) will be the same for any subset of code words and is therefore not used. MEDP = $2^{-28.272\ldots}$, MELP = $2^{-27.287\ldots}$.

The best differential in Kuznyechik consists of a *single* differential trail, but the best linear hull consists of 48 linear characteristics. Due to the algorithm 5.1 it is shown that for the majority of considered subsets of codewords

the best differential on them is not achieved. For the remaining subsets, an attempt is made to construct the best differential (algorithm 5.2). This is due to a sequence of transitions from the estimate (10) to the exact value (11). We got: MEDP $= 2^{-86.66...}$, MELP $= 2^{-76.936...}$.

## 7 Conclusion

The article presented: the algorithm for finding codewords with the small byte weight; algorithms for finding the complete description of the best differential trails (linear characteristics), differentials (linear hulls) in 2-round Kuznyechik.

The best differentials (linear hulls) and their probabilities were found. It was shown that the best differential contains one differential trail; the best linear hull contains 48 linear characteristics (Appendix A and B). We proved that 2-round MEDP $= 2^{-86.66...}$, MELP $= 2^{-76.936...}$. The estimate (5) for a differential trail (linear characteristic) is not achieved for 2-round Kuznyechik.

For any LSX cipher, the $N$-round MEDP (MELP) is the upper bound for $(N+1)$-round MEDP (MELP). Therefore, the 2-round MEDP (MELP) of Kuznyechik is the upper bound for any larger number of rounds. Obtaining a more precise upper bounds is the subject of further research.

## Acknowledgments

The author is very grateful to Igor Arbekov and Anton Naumenko for valuable comments and suggestions on the text of the article.

## References

[1] Liam Keliher and Jiayuan Sui. *Exact Maximum Expected Differential and Linear Probability for 2-Round Advanced Encryption Standard (AES),* Cryptology ePrint archive, Report 2005 / 321, 2005, `https://eprint.iacr.org/2005/321`.

[2] E. Biham, *On Matsui's linear cryptanalysis*, Advances in Cryptology – EUROCRYPT'94, in: Lecture Notes in Comput. Sci., Vol. 950, Springer, Berlin, 1995, pp. 341-355.

[3] GOST R 34.12-2015 - National standard of the Russian Federation – Information technology – Cryptographic data security – Block ciphers, 2015.

[4] National Institute of Standards and Technology. Advanced Encryption Standard (AES) (FIPS PUB 197), 2001.

[5] F.J.MacWilliams, N.J.A.Sloane. *The Theory of Error-Correcting Codes.* North Holland, Amsterdam, 1977

[6] Hardy G.H., Littlewood J.E., Pólya G. *Inequalities, Cambridge Mathematical Library (2. ed.)*, Cambridge: Cambridge University Press, 1952

## Appendix

## A The best differentials

| | |
|---|---|
| 0 8 0 0 0 0 8 0 8 8 8 0 6 0 0 8 | $P(\Delta x \to \Delta_1) \cdot 256$ |
| 0019000000002d00b8b8950072000028 | $\Delta_1$ |
| 2a00000d2337f74d0082a80000009d1b | $\Delta_2$ |
| 8 0 0 8 8 8 8 6 0 8 6 0 0 0 6 8 | $P(\Delta_2 \to \Delta y) \cdot 256$ |

Table 1: First optimal internal part($\Delta_1 \to \Delta_2$). It generates one of the best differential.

| | |
|---|---|
| 0 8 8 6 0 8 8 8 0 8 8 0 0 8 0 8 | $P(\Delta x \to \Delta_1) \cdot 256$ |
| 00a5def70085853700ec0300009c005a | $\Delta_1$ |
| 0068ea0d00f700dd006d000000000090 | $\Delta_2$ |
| 0 6 6 8 0 8 0 6 0 8 0 0 0 0 0 8 | $P(\Delta_2 \to \Delta y) \cdot 256$ |

Table 2: Second optimal internal part($\Delta_1 \to \Delta_2$). It generates 24 best differentials.

## B Application to Linear Cryptanalysis

There is a certain duality between differential and linear cryptanalysis [2]. It allows us to apply the algorithms described above to calculate linear characteristics.

We make the appropriate substitutions.

Differential probability (1), are replaced by linear probability. DDT is replaced by Linear Approximation Table (LAT). Input/output differences $\alpha$ and $\beta$ are replaced by input/output masks $\alpha'$ and $\beta'$ correspondingly.

$$P(\alpha' \to \beta') = (2\Pr(\alpha' \bullet \chi = \beta' \bullet S(\chi)) - 1)^2,\ \alpha', \beta', \chi \in \{0,1\}^8,$$

where $\bullet$ is the inner product over $\{0,1\}$.

By analogy with the differential trail a linear characteristic for 2 rounds is introduced:

$$\mathbf{a} \to \mu_1 \to \mu_2 \to \mathbf{b}.$$

Its probability (by analogy with (2)) is equal to

$$P(\mathbf{a} \to \mu_1 \to \mu_2 \to \mathbf{b}) = \left(\prod_{j=1}^{n} P(\mathbf{a}[j] \to \mu_1[j])\right)\left(\prod_{j=1}^{n} P(\mu_2[j] \to \mathbf{b}[j])\right),$$

where $[j]$ is j-th coordinate of the corresponding vector.

The linear hull (similar to differential) is the set of all linear characteristics having input mask $\mathbf{a}$ and output mask $\mathbf{b}$.

$$(\mathbf{a} \to \mathbf{b}) = \{\mathbf{a} \to \mu_1^{(i)} \to \mu_2^{(i)} \to \mathbf{b},\ i = \overline{1,T}\}.$$

The probability of the linear hull $(\mathbf{a} \to \mathbf{b})$ is equal to:

$$P(\mathbf{a} \to \mathbf{b}) = \sum_{i=1}^{T}\left(\left(\prod_{j=1}^{n} P(\mathbf{a}[j] \to \mu_1^{(i)}[j])\right)\left(\prod_{j=1}^{n} P(\mu_2^{(i)}[j] \to \mathbf{b}[j])\right)\right),$$

where $T$ is the number of linear characteristics.

You need to replace all formulas in the sections (4) and (5) according to the above analogies.

The maximum probability of the local linear characteristic of Kuznyechik is

$$P_{max}(\alpha' \to \beta') = \max_{(\alpha',\beta')\setminus(0,0)} P(\alpha' \to \beta') =$$
$$= \left(2\left(\frac{128+28}{256}\right) - 1\right)^2 = \left(\frac{56}{256}\right)^2.$$

The trivial estimate of the two-round linear characteristic is

$$\max_{(\mathbf{a},\boldsymbol{\mu_1},\boldsymbol{\mu_2},\mathbf{b})\setminus(0,0,0,0)} P(\mathbf{a} \to \mu_1 \to \mu_2 \to \mathbf{b}) \le \left(\frac{56}{256}\right)^{2\cdot 17} = 2^{-74.549\ldots}.$$

The following results are obtained by executing the algorithms.

The best linear characteristic has a probability equal to

$$\max_{(\mathbf{a},\mu_1,\mu_2,\mathbf{b})\setminus(0,0,0,0)} P(\mathbf{a}\to\mu_1\to\mu_2\to\mathbf{b}) =$$
$$= \left(\frac{56}{256}\right)^{2\cdot 8}\left(\frac{52}{256}\right)^{2\cdot 7}\left(\frac{48}{256}\right)^{2\cdot 2} = 2^{-76.936\ldots}.$$

The linear hull $(\mathbf{a}\to\mathbf{b})$ has a nontrivial form and (unlike the differential method) contains 48 linear characteristics $\mathbf{a}\to\mu_1^{(i)}\to\mu_2^{(i)}\to\mathbf{b}$, $i=\overline{1,48}$. The exact probability of the linear hull is

$$\max_{(\mathbf{a},\mathbf{b})\setminus(0,0)} P(\mathbf{a}\to\mathbf{b}) = 2^{-76.936\ldots}\cdot\left(1+2^{-57.6654\ldots}\right).$$

| | |
|---|---|
| 00 00 28 28 00 00 00 28 28 24 26 00 26 00 28 00 | $256\cdot\frac{\sqrt{P(\mathbf{a}\to\mu_1)}}{2}$ |
| 00 00 69 a7 00 00 00 55 67 8b e9 00 93 00 69 00 | $\mu_1$ |
| cc 00 f8 00 f6 3f 4c 31 e1 45 00 fb 00 00 00 00 | $\mu_2$ |
| 26 00 26 00 28 26 28 26 24 26 00 28 00 00 00 00 | $256\cdot\frac{\sqrt{P(\mu_2\to\mathbf{b})}}{2}$ |

Table 3: Optimal internal part$(\mu_1\to\mu_2)$. It generates 4 best linear characteristics.

The optimal inner part $(\mu_1\to\mu_2)$ generates the best linear hull.

| | |
|---|---|
| 00 00 b8 4c 00 00 00 48 66 d1 f7 00 cc 00 b8 00 | **a** |
| 93 00 e0 00 1c 91 a7 b8 62 e8 00 36 00 00 00 00 | **b** |

Table 4: The best linear hull $(\mathbf{a},\mathbf{b})$

The best linear hull $(\mathbf{a},\mathbf{b})$ consist of 48 linear characteristics $\mathbf{a}\to\mu_1^{(i)}\to\mu_2^{(i)}\to\mathbf{b}$, which are listed below (Table 5 and 6).

| $i$ | $\mu_1^{(i)}$ | $\mu_2^{(i)}$ | $\log_2 P(\mathbf{a} \to \mu_1^{(i)} \to \mu_2^{(i)} \to \mathbf{b})$ |
|---|---|---|---|
| 1 | 0000052d000000cd14902b003e000500 | 6e007d006cad40bcf0b8001800000000 | -135.596... |
| 2 | 00000e7e000000c238e2cb009a000e00 | 7a009e00a8c3bff5e392009100000000 | -166.150... |
| 3 | 000010900000002940043 1001e001000 | dd00530003abfcfb868400c100000000 | -152.862... |
| 4 | 000018d8000000dc6006c80011001800 | 52009b00e31f8267c5c6004000000000 | -153.891... |
| 5 | 00001aca0000007068e7670062001a00 | 0100a900db327c40a53700f100000000 | -145.669... |
| 6 | 00001df5000000117496e3002f001d00 | 3c00e6008fb2c2db357e005800000000 | -145.247... |
| 7 | 00001fe7000000bd7c774c005c001f00 | 6f00d400b79f3cfc558f00e900000000 | -171.728... |
| 8 | 000025ce0000009f9498490002002500 | 1700db006a387b893f73005900000000 | -161.130... |
| 9 | 00002ab90000000ba8eb340040002a00 | a5005c00de0cbb8eec78007100000000 | -155.482... |
| 10 | 00002c8f0000003cb00b0600d5002c00 | 50000a00967b7ae74ca8006100000000 | -149.107... |
| 11 | 00002e9d00000090b8eaa900a6002e00 | 03003800ae5684c02c5900d000000000 | -135.819... |
| 12 | 000036450000004cd8ec6100b7003600 | 5100a3004d4906a7e99f009000000000 | -153.365... |
| 13 | 0000374c0000001adc7dd7006f003700 | 9900ba0051be7955d906002900000000 | -139.528... |
| 14 | 00003e0d000000b9f8ee9800b8003e00 | de006b00adfd783baadd001100000000 | -164.616... |
| 15 | 0000431e0000005ecf60dd00d3004300 | 6900a4002833f7bf0d3d008a00000000 | -153.084... |
| 16 | 0000473a000000c5df61400035004700 | cf00c0005869c8f1cd1c002b00000000 | -145.714... |
| 17 | 0000484d00000051e3123d0077004800 | 7d004700ec5d08f61e17000300000000 | -152.101... |
| 18 | 00004e7b00000066fbf20f00e2004e00 | 88001100a42ac99fbec7001300000000 | -143.365... |
| 19 | 0000509500000008d8314f50066005000 | 2f00dc000f428a91dbd1004300000000 | -146.685... |
| 20 | 000057aa000000ec9f6571002b005700 | 120093005bc2340a4b9800ea00000000 | -139.832... |
| 21 | 000058dd00000078a3160c0069005800 | a0001400eff6f40d989300c200000000 | -156.920... |
| 22 | 000059d40000002ea787ba00b1005900 | 68000d00f3018bffa80a007b00000000 | -149.914... |
| 23 | 000061ef000000a0478910009c006100 | 43003000168b32ada207007a00000000 | -153.247... |
| 24 | 000064c20000006d53193b00a2006400 | 2d004d007a26721152bf006200000000 | -166.535... |

Table 5: Linear characteristics included in the best linear hull. $i = \overline{1, 24}$

| $i$ | $\mu_1^{(i)}$ | $\mu_2^{(i)}$ | $\log_2 P(\mathbf{a} \to \mu_1^{(i)} \to \mu_2^{(i)} \to \mathbf{b})$ |
|---|---|---|---|
| 25 | 000069a700000055678be90093006900 | cc00f800f63f4c31e14500fb00000000 | -76.9363... |
| 26 | 00006abc000000af6bfbf00038006a00 | 5700d300d2e5cde4b12d00f300000000 | -148.417... |
| 27 | 00006c8a00000098731bc200ad006c00 | a20085009a920c8d11fd00e300000000 | -153.506... |
| 28 | 00006d83000000ce778a740075006d00 | 6a009c008665737f2164005a00000000 | -152.558... |
| 29 | 00006e98000000347bfa6d00de006e00 | f100b700a2bff2aa710c005200000000 | -153.506... |
| 30 | 00007b25000000d02f6e7700fe007b00 | 42009900cdb94eed0730008b00000000 | -168.150... |
| 31 | 000088420000007e6522b200ff008800 | a8001500f8a59248f9e8004600000000 | -165.453... |
| 32 | 00008a50000000d26dc31d008c008a00 | fb002700c0886c6f991900f700000000 | -149.061... |
| 33 | 00008b59000000846952ab0054008b00 | 33003e00dc7f139da980004e00000000 | -157.394... |
| 34 | 00008f7d0000001f79533600b2008f00 | 95005a00ac252cd369a100ef00000000 | -142.908... |
| 35 | 0000a7d6000000b8d959ad008100a700 | 630034004a04697ae528002f00000000 | -158.535... |
| 36 | 0000ae970000001bfdcae2005600ae00 | 2400e500b647681496f3001700000000 | -158.862... |
| 37 | 0000b170000000a681bdae000a00b100 | 4b00310001d854e8c37c00fe00000000 | -145.329... |
| 38 | 0000c7300000004e9a410b00c500c700 | e8001d004078242577b600ec00000000 | -150.920... |
| 39 | 0000d384000000fcca44a7003d00d300 | 93002a003389e7903113008c00000000 | -162.943... |
| 40 | 0000d6a900000031ded48c000300d600 | fd0057005f24a72cc1ab009400000000 | -149.247... |
| 41 | 0000dee1000000c4fed675000c00de00 | 72009f00bf90d9b082e9001500000000 | -146.676... |
| 42 | 0000e8a400000088263a1400bb00e800 | 23003c00f2d9df176b76008500000000 | -159.470... |
| 43 | 0000e9ad000000de22aba2006300e900 | eb002500ee2ea0e55bef003c00000000 | -158.558... |
| 44 | 0000f07c00000054463cdc00aa00f000 | 7100a70011c65d70aeb000c500000000 | -153.587... |
| 45 | 0000f458000000cf563d41004c00f400 | d700c300619c623e6e91006400000000 | -149.620... |
| 46 | 0000f743000000355a4d5800e700f700 | 4c00e8004546e3eb3ef9006c00000000 | -160.150... |
| 47 | 0000f834000000a1663e2500a500f800 | fe006f00f17223ecedf2004400000000 | -152.195... |
| 48 | 0000fe02000000967ede17003000fe00 | 0b003900b905e2854d22005400000000 | -153.084... |

Table 6: Linear characteristics included in the best linear hull. $i = \overline{25, 48}$

## C Codewords with minimum binary weight

Let $\mathbb{G} = \mathbb{E}|\mathbb{L}$ is a linear binary code, codeword length – 256 bits, infoword length – 128 bits.

$\mathbb{L}$ is $128 \times 128$ binary matrix, which defines the linear transformation of Kuznyechik.

It is shown (algorithm of the section (3)) that in a linear binary code $\mathbb{G}$ there are no codewords of binary weight 17, 18, 19, 20.

Two codewords with binary weight equal to 29 are found.

| 0 2 0 2 2 0 0 0 0 2 1 0 1 2 0 1 | $w$ |
|---|---|
| 009000a00300000000009010001090004 | $x$ |
| 150400090100010900000000003a00090 | $y = x\mathbb{L}$ |
| 3 1 0 2 1 0 1 2 0 0 0 0 2 2 0 2 | $w$ |

Table 7: The codeword with a binary weight equal to 29

| 2 0 2 2 0 0 0 0 2 1 0 1 2 0 1 3 | $w$ |
|---|---|
| 9000a00300000000009010001090000415 | $x$ |
| 040009010001090000000003a0009000 | $y = x\mathbb{L}$ |
| 1 0 2 1 0 1 2 0 0 0 0 2 2 0 2 0 | $w$ |

Table 8: Another codeword with a binary weight equal to 29

## D The proof of Lemma 2

**Lemma 3.** *Let $\Delta x \to \Delta y$ is the differential in 2-round Kuznyechik. Let $P(\Delta x \to \Delta y)$ is maximal among all differentials. Then the number of active S-boxes in $\Delta x \to \Delta y$ is equal to $n + 1 = 17$.*

*Proof* Denote $P_{best}^{diffA}$ the best differential with $A$ active S-boxes.

It is shown that among differentials containing trails of weight $n+1 = 17$, the best probability is

$$P_{best}^{diff17} = \left(\frac{8}{256}\right)^{13} \left(\frac{6}{256}\right)^{4} = 2^{-86.660\ldots}.$$

We will show that

$$P_{best}^{diff} = P_{best}^{diff17} < P_{best}^{diffA},\ n + 2 \le A \le 2n.$$

Consider an arbitrary differential $\Delta x \to \Delta y$ with 18 active S-boxes. The differential consists of trails of the form $\Delta x \to \Delta_1 \to \Delta_2 \to \Delta y$. The difference $\Delta x$ and all the $\Delta_1$ differences have the same set of active S-boxes. $(k_1, \ldots, k_t)$ is the set of their positions. Similarly for $\Delta y$ and $\Delta_2$, let's denote the positions of active S-boxes $(m_1, \ldots, m_r)$, $t + r = 18$.

Using the algorithm (3), you can find all pairs $(\Delta_1, \Delta_2)$ corresponding to this set of active S-boxes. All differential trails $\Delta x \to \ldots \to \Delta y$ can only pass through these pairs. During the algorithm execution the system of equations

with $18 - n = 2$ free variables will be solved. The number of solutions, and accordingly the number of pairs $(\Delta_1, \Delta_2)$, will not exceed $255^{18-n} = 255^2$.

Let's present the set of pairs found as a table $\boldsymbol{D}$. Table size is equal to $255^2 \times 18$. Each row corresponds to a pair $(\Delta_1^{(i)}, \Delta_2^{(i)}) = (\alpha_{k_1}^{(i)}, \ldots, \alpha_{k_t}^{(i)}, \beta_{m_1}^{(i)}, \ldots, \beta_{m_r}^{(i)})$, $i \leq 255^2$, and each column corresponds to the active S-box.

By definition, the probability of a differential with 18 active S-boxes is:

$$P(\Delta x \to \Delta y) = \sum_{i=1}^{T} \left( \left( \prod_{j=1}^{t} P(x_{k_j} \to \alpha_{k_j}^{(i)}) \right) \left( \prod_{j=1}^{r} P(\beta_{m_j}^{(i)} \to y_{m_j}) \right) \right),$$
$$T \leq 255^2, \ t + r = 18.$$

Let the $\Delta x$ and $\Delta y$ are fixed. Then each element of the table can be matched with the probability $P(x_{k_j} \to \alpha_{k_j}^{(i)})$ (or $P(\beta_{m_j}^{(i)} \to y_{m_j})$). Let us denote this probability $P_{i,j}$, then the probability of the differential is:

$$P(\Delta x \to \Delta y) = \sum_{i=1}^{T} \prod_{j=1}^{r+t} P_{i,j}. \tag{12}$$

We give an upper bound of the (12).

Note that there are no more than 255 identical bytes in each column of the table $\boldsymbol{D}$. Otherwise, there are rows with a pair of identical bytes. This corresponds to the existence of a codeword with a weight less than $n+1 = 17$. It contradicts the MDS-code definition.

Let the input $x_{k_j}$ or output $y_{m_j}$ bytes are fixed. Then the same bytes in the table column match the same probabilities.

Denote $p_8 = \frac{8}{256}$, $p_6 = \frac{6}{256}$, $p_4 = \frac{4}{256}$, $p_2 = \frac{2}{256}$.

Let's use the majorants (8). They take the following values:

$$X = p_8, \underbrace{p_6, \ldots, p_6}_{5}, \underbrace{p_4, \ldots, p_4}_{21}, \underbrace{p_2, \ldots, p_2}_{87}, \underbrace{0, \ldots, 0}_{141}; \tag{13}$$

$$Y = p_8, p_8, \underbrace{p_6, \ldots, p_6}_{7}, \underbrace{p_4, \ldots, p_4}_{27}, \underbrace{p_2, \ldots, p_2}_{92}, \underbrace{0, \ldots, 0}_{127}. \tag{14}$$

You can see that $Y$ is always greater than $X$. To get the highest estimate we consider the case when 2 columns of the table are estimated using $X$ (and 16 columns – $Y$).

The number of nonzero elements in the majorant $X$ is $v = 114$. This allows us to refine the maximum number of differential trails in the differential

$T \le v^2 = 12996$. And also refine the values of majorants:

$$X = \underbrace{p_8, \underbrace{p_6, \ldots, p_6}_{5}, \underbrace{p_4, \ldots, p_4}_{21}, \underbrace{p_2, \ldots, p_2}_{87}}_{v=114}; \tag{15}$$

$$Y = \underbrace{p_8, p_8, \underbrace{p_6, \ldots, p_6}_{7}, \underbrace{p_4, \ldots, p_4}_{27}, \underbrace{p_2, \ldots, p_2}_{78}}_{v=114}. \tag{16}$$

We divide the columns of the table into two groups:

$$\sum_{i=1}^{T} \prod_{j=1}^{r+t} P_{i,j} = \sum_{i=1}^{T} \underbrace{(P_{i,1} \cdot P_{i,2})}_{\mathbb{I}} \cdot \underbrace{\left( \prod_{j=3}^{18} P_{i,j} \right)}_{\mathbb{III}}. \tag{17}$$

We multiply the elements of the group $\mathbb{I}$ in pairs:

$$P_{i,1} \cdot P_{i,2} = P_i^{(I)}, \ \forall i = \overline{1, T}.$$

Arrange in each row of $\mathbb{III}$ all factors in non-increasing order.

Arrange the elements of each sequence $P_1^{(I)}, \ldots, P_T^{(I)}, P_{1,j}, \ldots, P_{T,j}, \ \forall j = \overline{3, 18}$ (columns in $\boldsymbol{D}$) in a non-increasing order. Denote the elements of the resulting sequences $\hat{P}_1^{(I)}, \ldots, \hat{P}_T^{(I)}, \ \hat{P}_{1,j}, \ldots, \hat{P}_{T,j}, \ \forall j = \overline{3, 18}$.

From the rearrangement inequality [6] it follows that

$$\sum_{i=1}^{T} \underbrace{(P_{i,1} \cdot P_{i,2})}_{\mathbb{I}} \cdot \underbrace{\left( \prod_{j=3}^{18} P_{i,j} \right)}_{\mathbb{III}} \le \sum_{i=1}^{T} \underbrace{\hat{P}_i^{(I)}}_{\mathbb{I}} \cdot \underbrace{\left( \prod_{j=3}^{18} \hat{P}_{i,j} \right)}_{\mathbb{III}}. \tag{18}$$

Let's estimate $\hat{P}_1^{(I)}, \ldots, \hat{P}_T^{(I)}$ using $X$ (13). Knowing that all pairs in the first and second columns are different, we replace the elements of the sequence by the $X \times X$:

$$p_8^2, \ \underbrace{p_8 p_6, \ldots, p_8 p_6}_{10 \text{ lines}}, \ \underbrace{p_6, \ldots, p_6}_{25 \text{ lines}}, \ \underbrace{p_8 p_4, \ldots, p_8 p_4}_{42 \text{ lines}}, \ \ldots. \tag{19}$$

Let's estimate the group $\mathbb{III}$.

We note that the following inequality holds:

$$\hat{P}_i = \hat{P}_i^{(I)} \cdot \left( \prod_{j=3}^{18} \hat{P}_{i,j} \right) \le \hat{P}_{i+1} = \hat{P}_{i+1}^{(I)} \cdot \left( \prod_{j=3}^{18} \hat{P}_{i+1,j} \right), \ \forall i = \overline{1, T-1}. \tag{20}$$

Assume that the coordinates of all elements $p_8$ in $\mathbb{III}$ are known (Fig.2.a).

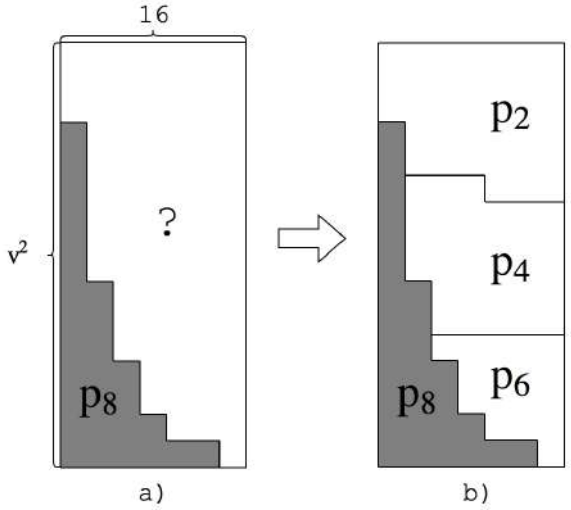



Figure 2: Reordering elements in $\mathbb{III}$.

We describe the *procedure for reordering* all elements $p_6$, $p_4$ and $p_2$ in $\mathbb{III}$.

1) Select the element in the first row $\hat{P}_{1,z} \neq p_8$, $z = \overline{3,18}$. Let $z$ be the smallest (left column). If in the first row all elements are equal to $p_8$, we consider the second row, etc.

2) Find the maximum of all elements in $\mathbb{III}$, which have not been reordered before:

$$\hat{P}_{i',j'} = \max_{i,j} \hat{P}_{i,j}, \;\; \hat{P}_{i,j} \neq p_8, \;\; i, i' = \overline{1,T}, \;\; j, j' = \overline{3,18}.$$

3) We will exchange the values of the elements $\hat{P}_{1,z}$ and $\hat{P}_{i',j'}$. If $i' = 1$, then (18) does not change due to commutativity of multiplication. If $i' \neq 1$, then due to (20) then estimate (18) does not decrease. Note that after the exchange of elements can be broken inequalities (20).

4) Arrange the elements in columns $\hat{P}_1^{(I)}, \ldots, \hat{P}_T^{(I)}$, $\hat{P}_{1,j}, \ldots, \hat{P}_{T,j}$, $\forall j = \overline{3,18}$ by non-increasing. As a consequence of rearrangement inequality, (20) will be true. The value $\sum_{i=1}^{T} \hat{P}_i^{(I)} \cdot \left( \prod_{j=3}^{18} \hat{P}_{i,j} \right)$ will not decrease. The sequence $\hat{P}_1^{(I)}, \ldots, \hat{P}_T^{(I)}$ , the coordinates of the elements $p_8$ and the value of the element with coordinates $(1, z)$ do not change.

The element with coordinates $(1, z)$ has been reordered.

We choose in the first row the next element not equal to $p_8$. We will perform the above steps 1 – 4.

Perform steps 1 – 4 sequentially for each element of the table not equal to $p_8$ and which has not been reordered before.

The result of the procedure will be the table $\widehat{\boldsymbol{D}}$. An exemplary view of the table $\widehat{\boldsymbol{D}}$ is shown in the figure 2.b. At each step of the procedure, the

estimate (18) does not decrease. Suppose that there is a table $\widetilde{\boldsymbol{D}}$, which gives a greater estimate. If $\widetilde{\boldsymbol{D}}$ coincide with $\widehat{\boldsymbol{D}}$ within the accuracy of permutation of the same elements, then estimates (18) are the same, too. If $\widetilde{\boldsymbol{D}}$ does not coincide with $\widehat{\boldsymbol{D}}$, then apply the *reorder procedure* to the table $\widetilde{\boldsymbol{D}}$. Due to the steps that do not decrease the estimate (18), the table $\widehat{\boldsymbol{D}}$, will be built.

Thus it is proved that for a given arrangement of all elements $p_8$, the *reordering procedure* allows us to obtain the greatest estimate (18).

Let us now consider the possible arrangement of elements $p_8$ in the group $\mathbb{III}$.

The numbers of the elements $p_8$ in the tables $\boldsymbol{D}$ and $\widehat{\boldsymbol{D}}$ are the same. The number of rows containing the same number of elements $p_8$ also coincides.

Let $w_i$ be the number of elements $p_8$ in the $i$-th row of the table $\widehat{\boldsymbol{D}}$, $w_i \geq w_{i+1}$, $i = \overline{1, T-1}$. Then

$$\sum_{i=1}^{T} w_i \leq v \cdot 16 \cdot 2 = 3648, \tag{21}$$

16 – the number of columns in the group $\mathbb{III}$, 2 – the number of elements $p_8$ in (16). Hence,

$$\left|\{i \ : w_i > 0, \ i = \overline{1, T}\}\right| \leq v \cdot 16 \cdot 2 = 3648. \tag{22}$$

The number of rows containing exactly 2 elements $p_8$ can be estimated as a $\binom{16}{2} \cdot 2^2$ – the number of pairs multiplied by the number of variants in the pair. Assume that the number of such pairs is greater. There are two different rows (two different codewords) that contain the same pair of bytes. Therefore, the sum of such codewords will give a codeword with a weight of 16 or less. It contradicts the MDS-code definition.

Let us estimate the number of rows with a greater number of elements. The maximum number of pairs is known – $\binom{16}{2} \cdot 2^2$. On the other hand, let $i$-th row contains $w_i$ elements $p_8$, then this row contains $\binom{w_i}{2}$ different pairs of elements $p_8$. Then the number of rows containing exactly $w$ elements $p_8$ is limited:

$$\left|\{i \ : w_i = w, \ i = \overline{1, T}\}\right| \leq \binom{16}{2} \cdot 2^2 / \binom{w}{2}, \ 2 \leq w \leq 16. \tag{23}$$

And also:

$$\left|\{i \ : w_i \geq w, \ i = \overline{1, T}\}\right| \leq \binom{16}{2} \cdot 2^2 / \binom{w}{2}, \ 2 \leq w \leq 16. \tag{24}$$

In addition, there should be a limit for the total number of pairs of elements $p_8$ in the table $\widehat{\boldsymbol{D}}$:

$$\sum_{i=1}^{T} \binom{w_i}{2} \leq \binom{16}{2} \cdot 2^2 = 480. \tag{25}$$

It is possible to show that the number of rows containing exactly $\omega = 8$ elements $p_8$, no more than $\rho \leq 5$. In each column of the table $\widehat{\boldsymbol{D}}$, no more than two different byte values correspond to the value of $p_8$. Any row must have at most one intersection (the same byte in the same column) with any other row. Initially, the number of bytes that were not selected is equal to $\nu = 2 \cdot 16 = 32$.

Choose the first row that contains exactly 8 elements $p_8$. Subtract $\omega = 8$ from $\nu$.

Choose the second row that intersects the first row. Subtract $\omega - 1 = 7$ from $\nu$.

Select the third row that intersects the first row and the second row. The minimum number that can be subtracted from $\nu$ is $\omega - 2 = 6$.

And so on:

$$\nu - (\omega \cdot \rho - \sum_{i=1}^{\rho-1} i) \geq 0,$$

$$\nu - \omega\rho + \frac{\rho(\rho-1)}{2} \geq 0,$$

$$\frac{1}{2}\rho^2 - (\omega + \frac{1}{2}) \cdot \rho + \nu \geq 0.$$

Then

$$\frac{1}{2}\rho^2 - (8 + \frac{1}{2}) \cdot \rho + 32 \geq 0 \tag{26}$$

Hence, $\rho \in \{0, 1, 2, 3, 4, 5\}$. If $\rho = 6$ then (26) less than zero.

Similarly, when $\omega = 9$ that $\rho \leq 4$. I.e. it is possible to show that the number of rows containing exactly 9 elements $p_8$, no more than 4. If $\omega = 10$ or $\omega = 11$ then $\rho \leq 3$. If $\omega \in \{12, 13, 14, 15, 16\}$ then $\rho \leq 2$.

Also, the following inequalities are true:

$$\begin{aligned} \left|\{i \ : w_i \geq 8, \ i = \overline{1,T}\}\right| &\leq 5 \\ \left|\{i \ : w_i \geq 9, \ i = \overline{1,T}\}\right| &\leq 4 \\ \left|\{i \ : w_i \geq 10, \ i = \overline{1,T}\}\right| &\leq 3 \\ \left|\{i \ : w_i \geq 12, \ i = \overline{1,T}\}\right| &\leq 2 \end{aligned} \tag{27}$$

$$w_i \le \min\left(2 \cdot 16 - w_1 - (w_2 - 1) + 2,\ w_{i-1}\right),\ i = \overline{3, T} \tag{28}$$

Let $w_i > 2 \cdot 16 - w_1 - (w_2 - 1) + 2$. Then the $i$-th row must have at least two identical bytes with the first row or second row. It contradicts the MDS-code definition.

Let's iterate all possible sets $w_i$, $i = \overline{1, T}$. We will take into account the restrictions (21), (23), (25), (27), (28).

We choose the maximum estimate among all sets $w_i$, $i = \overline{1, T}$.

$$\sum_{i=1}^{T} \hat{P}_i^{(I)} \cdot \left( \prod_{j=3}^{18} \hat{P}_{i,j} \right) \le 2^{-87.469\ldots} < P_{best}^{diff17} = 2^{-86.660\ldots}. \tag{29}$$

Note that it is possible to obtain more rough estimate without any additional search. We will not use restrictions (21), (25). Take the maximum values of the inequalities (24) and (27). The inequality (27) shows that the greatest $w_1, \ldots, w_5 = (16, 16, 11, 9, 8)$. Upper bounds in the inequality (24): exactly 7 elements $p_8$ – 17 rows, 6 elements – 10 rows, 5 elements – 16 rows, 4 elements – 32 rows, 3 elements – 80 rows, 2 elements – 320 rows, 1 element – 3168 rows.

$$\sum_{i=1}^{T} \hat{P}_i^{(I)} \cdot \left( \prod_{j=3}^{18} \hat{P}_{i,j} \right) \le 2^{-87.012\ldots} < P_{best}^{diff17} = 2^{-86.660\ldots}. \tag{30}$$

The best estimate for a differential with 19 active S-boxes ($P_{best}^{diff19}$) cannot be greater than the best estimate for a differential with 18 active S-boxes ($P_{best}^{diff18}$).

$$P_{best}^{diff19} \le \sum_{i=1}^{255} P(x_{k_1} \to \alpha_{k_1}^{(i)}) \cdot P_{best}^{diff18} =$$

$$= P_{best}^{diff18} \cdot \sum_{i=1}^{255} P(x_{k_1} \to \alpha_{k_1}^{(i)}) = P_{best}^{diff18} \cdot 1, \quad \forall k_1, x_{k_1}, \alpha_{k_1}.$$

Similarly for cases of $20, \ldots, 32$ active S-boxes.

Hence, the original lemma is proved:

$$P_{best}^{diff} = P_{best}^{diff17}.$$

**Lemma 4.** *Let* $(\mathbf{a} \to \mathbf{b})$ *is the linear hull in 2-round Kuznyechik. Let* $P(\mathbf{a} \to \mathbf{b})$ *be maximal among all linear hulls. Then the number of active S-boxes in*

$(\mathbf{a} \to \mathbf{b})$ *is equal to* $n + 1 = 17$.

*Proof* The proof is analogous to the Lemma of the best differential.
$p_8$ is replaced by $p_{28}^{'} = \left(\frac{2 \cdot 28}{256}\right)^2$.
$p_6$, $p_4$, $p_2$ is replaced by $p_{26}^{'} = \left(\frac{2 \cdot 26}{256}\right)^2$, ... , $p_2^{'} = \left(\frac{2 \cdot 2}{256}\right)^2$.
Majorants (13) and (14) are replaced by

$$X' = \underbrace{p_{28}^{'},\ p_{26}^{'},\ p_{24}^{'}, p_{24,}^{'}\ p_{22}^{'},\ p_{20}^{'}, p_{20}^{'}, p_{20}^{'},\ p_{18}^{'}, p_{18}^{'}, p_{18}^{'}, p_{18}^{'},\ \ldots, \underbrace{p_2^{'}, \ldots, p_2^{'}}_{40}}_{242},\ \underbrace{0, \ldots, 0}_{13}$$

and

$$Y' = \underbrace{p_{28}^{'}, p_{28}^{'},\ p_{24}^{'}, p_{24,}^{'}\ p_{22}^{'}, p_{22}^{'}, p_{22}^{'},\ p_{20}^{'}, p_{20}^{'}, p_{20}^{'}, p_{20}^{'},\ \ldots, \underbrace{p_2^{'}, \ldots, p_2^{'}}_{7}}_{247},\ \underbrace{0, \ldots, 0}_{8}$$

correspondingly.

Estimate of $P(\mathbf{a}, \mathbf{b})$ similar to (29):

$$P(\mathbf{a}, \mathbf{b}) \leq 2^{-77.310\ldots} < P_{best}^{lin} = 2^{-76.936\ldots}.$$

## E Pseudocode of algorithms

### Algorithm for finding codewords with the smallest byte weight

**Algorithm 1** Algorithm for finding codewords with the smallest byte weight

**Input:** $k[1 \dots t]$ – nonzero $x$ coordinates, $m[1 \dots r]$ – nonzero $y$ coordinates, $\mathbb{L}[1 \dots n, 1 \dots n]$, $t + r = n + 1$ // Matrix $\mathbb{L}$ in row-by-row representation

**Output:** $M^{(n+1)}(k_1, \dots, k_t, m_1, \dots, m_r)$

**function** *find_ codewords*$(k[1 \dots t], m[1 \dots r], \mathbb{L}[1 \dots n, 1 \dots n])$
$m'[1...n-r] := \{i : \ i \notin m, \ 1 \le i \le n\}$ // zero $y$ coordinates
$\mathbb{S}[1 \dots n-r, 1 \dots t]$
**for** $i := 1$ **to** $n - r$ **do**
  **for** $j := 1$ **to** $t$ **do**
    $\mathbb{S}[i][j] := \mathbb{L}[m'[i]][k[j]]$
  **end for**
**end for**
$\mathbb{S} :=$ identity_form($\mathbb{S}$)// Gauss method over $GF(2^8)$
// $\mathbb{S} = \begin{pmatrix} 1 & \cdots & 0 & c_1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & c_{n-r} \end{pmatrix}$
codewords := {}
$\alpha[1 \dots t] := [0 \dots 0]$
**for all** $e$ **in** $GF(2^8) \backslash 0$ **do**
  $\alpha[t] := e$
  **for** $i := 1$ **to** $t - 1$ **do**
    $\alpha[i] := e \times \mathbb{S}[i][t]$ // $\alpha_i = \alpha_t \times c_i$
  **end for**
  $\beta[1 \dots r] := L(\alpha)$ // zero coordinates are not specified
  codewords.add($(\alpha, \beta)$)
**end for**
**return** codewords

The above algorithm could be easily generalized to finding small weight $w > n + 1$ codewords. In this case, the number of free variables in each subsystem $\mathbb{S}_{n-r,t}$ increases. Accordingly, the number of codewords generated by a single subsystem increases to $255^{w-n}$. These codewords can include words that weigh less than $w$. This requires additional verification and increases the complexity of the algorithm.

The algorithm can be applied to an arbitrary MDS-code $(2n, n, n + 1)$ over any finite field $\mathbb{F}$.

We estimate the time complexity of the algorithm: Gaussian algorithm – $O(t^3)$; substitution of values – $O(ord(\mathbb{F})^{w-n})$; linear transformation – $O(n^2)$. The total complexity of the algorithm is $O(t^3 + ord(\mathbb{F})^{w-n} + n^2) = O(n^3 + ord(\mathbb{F})^{w-n})$.

One of the applications of this algorithm is the search in MDS-code codewords with small *binary* weight. The results are presented in Appendix B.

## Algorithm for finding the best differential trail

**Algorithm 2** Algorithm for finding the best differential trail

**Input:** $\mathbb{L}[1\dots n, 1\dots n]$, $\mathrm{DDT}[1\dots 255, 1\dots 255]$
// $\mathrm{DDT}[\alpha_i, \beta_j] = P(\alpha_i \to \beta_j)$, $i, j = \overline{1, 255}$, $\alpha_i, \beta_j \in \{0,1\}^8 \backslash 0$
**Output:** best_diff_trails, $P_{best}^{trail}$

**function** *find_best_diff_trails*($\mathbb{L}[1\dots n, 1\dots n]$, $\mathrm{DDT}[1\dots 255, 1\dots 255]$ )
best_diff_trails := {}
$P_{best}^{trail} := 0$
**for** $t := 1$ **to** $n$ **do**
  $r := n + 1 - t$
  **for all** $k[1\dots t]$ **in** combinations$(n, t)$ **do**
    **for all** $m[1\dots r]$ **in** combinations$(n, r)$ **do**
      codewords := *find_codewords*($k[1\dots t]$, $m[1\dots r]$, $\mathbb{L}$)
      // codewords$[i]$ =$(\Delta_1^{(i)}, \Delta_2^{(i)})$= $(\alpha_{k_1}^{(i)} \dots \alpha_{k_t}^{(i)}, \beta_{m_1}^{(i)} \dots \beta_{m_r}^{(i)})$, $i = \overline{1, 255}$
      **for all** $\alpha[1\dots t]$, $\beta[1\dots r]$ **in** codewords **do**
        $P_{max}(\Delta_1, \Delta_2) := get\_P\_max(\alpha[1\dots t], \beta[1\dots r], \mathrm{DDT})$
        **if** $P_{max}(\Delta_1, \Delta_2) = P_{best}^{trail}$ **then**
          best_diff_trails.add($(\alpha[1\dots t], \beta[1\dots r])$)
        **end if**
        **if** $P_{max}(\Delta_1, \Delta_2) > P_{best}^{trail}$ **then**
          $P_{best}^{trail} := P_{max}(\Delta_1, \Delta_2)$
          best_diff_trails := $\{(\alpha[1\dots t], \beta[1\dots r])\}$
        **end if**
      **end for**
    **end for**
  **end for**
**end for**
**return** best_diff_trails, $P_{best}^{trail}$

Time complexity of the algorithm 2 is

$$O\left(\underbrace{\sum_{t=1}^{n}\binom{n}{t}\binom{n}{n+1-t}}_{\text{all combinations}}\underbrace{\left(n^3+\mathrm{ord}(\mathbb{F})\right)}_{\text{find\_codewords}}\cdot\underbrace{(n+1)}_{\text{get\_P\_max}}\right) =$$
$$= O\left(\binom{2n}{n+1}\cdot n^4\right) = O\left(\frac{2^{2n}}{\sqrt{n}}n^4\right)$$

**Algorithm 3** Algorithm for calculating $P_{max}(\Delta_1, \Delta_2)$

**function** *get_ P_ max*($\alpha[1 \dots t]$, $\beta[1 \dots r]$, DDT$[1 \dots 255, 1 \dots 255]$)
// $(\Delta_1, \Delta_2) = (\alpha_{k_1} \dots \alpha_{k_t}, \beta_{m_1} \dots \beta_{m_r})$
$P_{max}(\Delta_1, \Delta_2) := 1$
**for** $i := 1$ **to** $t$ **do**
  $P_{max}(\Delta_1, \Delta_2) := P_{max}(\Delta_1, \Delta_2) \times \max_x(\text{DDT}[x][\alpha[i]])$
**end for**
**for** $j := 1$ **to** $r$ **do**
  $P_{max}(\Delta_1, \Delta_2) := P_{max}(\Delta_1, \Delta_2) \times \max_y(\text{DDT}[\beta[j]][y])$
**end for**
// the values $\max_x(\text{DDT}[x][y])$, $\max_y(\text{DDT}[x][y])$ can easily be cached
**return** $P_{max}(\Delta_1, \Delta_2)$

The complexity of the algorithm is trivial – $O(t + r) = O(n)$

## Algorithm for calculating the upper bound of the differential

**Algorithm 4** Algorithm for calculating the upper bound of the differential

**Input:** $M^{(n+1)}(k_1, \dots, k_t, m_1, \dots, m_r)$, DDT$[1 \dots 255, 1 \dots 255]$
**Output:** $P_{est} \geq P(\Delta x \to \Delta y)$

**function** *get_ upper_ bound*(codewords$[1 \dots 255]$, DDT$[1 \dots 255, 1 \dots 255]$)
P_parts$[1 \dots 255] := \{\}$
**for** $i := 1$ **to** $255$ **do**
  $\alpha[1 \dots t]$, $\beta[1 \dots r]$ := codewords$[i]$
  P_parts$[i]$ := *get_ P_ max*($\alpha[1 \dots t - u]$, $\beta[1 \dots r - v]$, DDT)// Let $u = v = 2$
**end for**
P_parts$[1 \dots 255]$ := non_increasing_sort(P_parts$[1 \dots 255]$)
$X[1 \dots 255]$ := *get_ majorant*(DDT$[1 \dots 255, 1 \dots 255]$, input)
$Y[1 \dots 255]$ := *get_ majorant*(DDT$[1 \dots 255, 1 \dots 255]$, output)
$P_{est} := 0$
**for** $i := 1$ **to** $255$ **do**
  $P_{est} := P_{est} + X[i]^u \times Y[i]^v \times$ P_parts$[i]$
**end for**
**return** $P_{est}$

The values returned by the function *get_ majorant* can be cached. Therefore, the complexity of the algorithm 4 is equal to $O(ord(\mathbb{F}) \cdot n)$.

**Algorithm 5** Algorithm for calculating $X$ and $Y$

**Input:** DDT$[1\dots255, 1\dots255]$, input ($X$) or output ($Y$)
**Output:** $X[1\dots255]$ or $Y[1\dots255]$, 8

**function** *get_majorant*(DDT$[1\dots255, 1\dots255]$, input/output)
**if** output **then**
  DDT := transpose(DDT)
**end if**
**for** $i := 1$ **to** 255 **do**
  DDT$[i][1\dots255]$ := non_increasing_sort(DDT$[i][1\dots255]$) // sort rows
**end for**
majorant$[1\dots255]$ := $[0,\dots,0]$
**for** $i := 1$ **to** 255 **do**
  majorant$[i]$ := $\max_j$(DDT$[j][i]$) // select the maximum in the column
**end for**
// zero values can be removed
**return** majorant

Time complexity of the algorithm 5 is $O(ord(\mathbb{F})^2)$.

## Algorithm for constructing the differential

**Algorithm 6** Algorithm for constructing the differential

**Input:** $M^{(n+1)}(k_1,\dots,k_t,m_1,\dots,m_r)$, DDT$[1\dots255, 1\dots255]$, $P_{est}^{diff}$
**Output:** best_differentials, $P_{est}^{diff}$

**function** *construct_differentials*(codewords$[1\dots255]$, DDT$[1\dots255, 1\dots255]$, $P_{est}^{diff}$)
row_index := $\{1,\dots,255\}$
row_est$[1\dots255]$ := $[0,\dots,0]$
**for** $i := 1$ **to** 255 **do**
  $\alpha[1\dots t]$, $\beta[1\dots r]$ := codewords$[i]$
  row_est$[i]$ := *get_P_max*($\alpha[1\dots t]$, $\beta[1\dots r]$, DDT)
**end for**
best_differentials := $\{\}$
external_bytes$[1\dots t+r]$ := $[0,\dots,0]$ // $\Delta x$ and $\Delta y$
*recursive_search*(1, row_index, row_est)
**return** best_differentials, $P_{est}^{diff}$

The complexity of the algorithm 6 is determined by the complexity of algorithm 7.

Denote the complexity of the algorithm for constructing the differential as $C_{diff}$. In general case, algorithm 7 performs an exhaustive search of all inputs $\Delta x$ and outputs $\Delta y$. In this case $C_{diff} = O(ord(\mathbb{F})^n)$. But in our practice, the average number of operations performed by the algorithm for constructing

the differential is approximately equal to $ord(\mathbb{F})^2$. A more accurate estimate of the complexity is the subject of further research.

**Algorithm 7** Recursive search of the differential

1: *variables from Algorithm 6:*
2: codewords$[1 \dots 255]$ // codewords$[i]$=codeword$[1 \dots t+r]$, $i = \overline{1, 255}$
3: DDT$[1 \dots 255, 1 \dots 255]$
4: external_bytes$[1 \dots t+r]$
5: $P_{est}^{diff}$
6: best_differentials = {}
7: **procedure** *recursive_search*(column, row_index, row_est)
8: **if** column $> t + r$ **then**
9: $\Delta x :=$ external_bytes$[1 \dots t]$, $\Delta y :=$ external_bytes$[t+1 \dots t+r]$
10: $P(\Delta x \to \Delta y) :=$ sum(row_est)
11: **if** $P(\Delta x \to \Delta y) = P_{est}^{diff}$ **then**
12: best_differentials.add$((\Delta x, \Delta y))$
13: **end if**
14: **if** $P(\Delta x \to \Delta y) > P_{est}^{diff}$ **then**
15: best_differentials = $\{(\Delta x, \Delta y)\}$
16: **end if**
17: **return**
18: **end if**
19: **for** $a := 1$ **to** 255 **do**
20: external_bytes[column] := $a$
21: new_row_index := {}, new_row_est$[1 \dots 255] := [0, \dots, 0]$, $P_{est} := 0$
22: **for all** $i$ **in** row_index **do**
23: codeword$[1 \dots t+r]$ := codewords$[i]$
24: $P_{trail} :=$ row_est$[i]$
25: internal_byte := codeword[column]
26: **if** column $\leq t$ **then**
27: $P_{trail} := P_{trail} \times$ DDT$[a]$[internal_byte]$/\max_x$(DDT$[x]$[internal_byte])
28: **else**
29: $P_{trail} := P_{trail} \times$ DDT[internal_byte]$[a]/\max_y$(DDT[internal_byte]$[y]$)
30: **end if**
31: **if** $P_{trail} > 0$ **then**
32: $P_{est} := P_{est} + P_{trail}$
33: new_row_index.add$(i)$
34: new_row_est$[i] := P_{trail}$
35: **end if**
36: **end for**
37: **if** $P_{est} \geq P_{est}^{diff}$ **then**
38: *recursive_search*(column+1, new_row_index, new_row_est)
39: **end if**
40: **end for**

## Algorithm for finding the best differential

**Algorithm 8** Algorithm for finding the best differential

**Input:** $\mathbb{L}[1\ldots n, 1\ldots n]$, DDT$[1\ldots 255, 1\ldots 255]$ , $P_{best}^{trail}$
**Output:** best_differentials, $P_{best}^{diff}$

**function** *find_best_differentials*($\mathbb{L}[1\ldots n, 1\ldots n]$, DDT$[1\ldots 255, 1\ldots 255]$ )
best_differentials := {}
best_diff_trails, $P_{best}^{trail}$ := *find_best_diff_trails*($\mathbb{L}$, DDT)
$P_{est}^{diff} := P_{best}^{trail}$
**for** $t := 1$ **to** $n$ **do**
  $r := n + 1 - t$
  **for all** $k[1\ldots t]$ **in** combinations$(n, t)$ **do**
    **for all** $m[1\ldots r]$ **in** combinations$(n, r)$ **do**
      codewords := *find_codewords*($k[1\ldots t]$, $m[1\ldots r]$, $\mathbb{L}$)
      $P_{est}$ := *get_upper_bound*(codewords, DDT)
      **if** $P_{est} < P_{est}^{diff}$ **then**
        **continue**
      **end if**
      differentials, $P_{est}$ := *construct_differentials*(codewords, DDT, $P_{est}^{diff}$)
      **if** $P_{est} = P_{est}^{diff}$ **then**
        best_differentials := best_differentials $\cup$ differentials
      **end if**
      **if** $P_{est} > P_{est}^{diff}$ **then**
        $P_{est}^{diff} := P_{est}$
        best_differentials := differentials
      **end if**
    **end for**
  **end for**
**end for**
$P_{best}^{diff} := P_{est}^{diff}$
**return** best_differentials, $P_{best}^{diff}$

Time complexity of the algorithm 8 is

$$O\left(\underbrace{\sum_{t=1}^{n}\binom{n}{t}\binom{n}{n+1-t}}_{\text{all combinations}}\left(\underbrace{n^3+\text{ord}(\mathbb{F})}_{\text{find\_codewords}}+\underbrace{\text{ord}(\mathbb{F})\cdot n}_{\text{get\_upper\_bound}}+\underbrace{C_{diff}}_{\text{construct\_differentials}}\right)\right)=$$

$$=O\left(\binom{2n}{n+1}\cdot C_{diff}\right)=O\left(\frac{2^{2n}}{\sqrt{n}}\cdot C_{diff}\right)$$

# Evaluation of the Maximum Productivity for Block Encryption Algorithms

Vladimir Fomichev[1,2,3,4], Alisa Koreneva[1],
Alfinur Miftakhutdinova[3], and Dmitry Zadorozhny[1]

[1]Security Code LLC, Moscow, Russia
fomichev.2016@yandex.ru, alisa.koreneva@gmail.com
[2]National Research Nuclear University «MEPhI», Moscow, Russia
[3]Financial University under the Government of the Russian Federation, Moscow, Russia
[4]Institute of Problems of Informatics (Russian Academy of Sciences), Moscow, Russia

**Abstract**

The increasing volumes of information processed in computer systems lead to the actual problem related to synthesis of block ciphers. The urgent task is to improve the encryption productivity. The paper is devoted to the development of the approach for solving the designated task. This approach is to use shift registers with several feedbacks as a round transformation of block cipher. We proposed a class of nonlinear transformations generalizing the Feistel network and constructed on the autonomous shift register of length $n$ over a set of 32-dimensional binary vectors with $m$ feedbacks, $32 \geq n > m \geq 1$. To assess the ultimate (maximum possible) productivity of the corresponding encryption algorithms, we investigated the dependence of productivity on a number of characteristics, such as the size of data blocks, the speed of the round confusion function, the exponent of the mixing digraph of the round transformation. The proposed integral characteristic of the ultimate encryption productivity can be used by developers for choosing the parameters of specified block encryption algorithms. The obtained results show that with increasing the block size, the encryption productivity grows but slower than the block size. We determined that the algorithm based on the shift register of length 15 with 5 feedbacks over $V_{32}$ is the most productive in the classes under research. The cryptographic properties of the proposed algorithms, not related with increasing of productivity, are not discussed in detail.

**Keywords:** block ciphers, encryption productivity, shift registers, full mixing, exponent of digraph.

## 1 Introduction

Block ciphers are widely used for encryption of large volumes of data. Therefore, a relevant direction in cryptology is connected with acceleration of implementations of iterative block encryption algorithms and construction of high-performance algorithms based on SP-networks, Feistel networks or some of its generalizations. In accordance with the construction of iterative

block ciphers [1, ch.15] the main factors determining the productivity of algorithms are as follows: the size of data blocks, computational complexity of software and/or hardware implementation for the round, the number of encryption rounds.

Selection of the most appropriate high-performance encryption algorithm in a particular cryptosystem is performed with a purpose of ensuring a sufficiently high cryptographic strength. Generally, this condition is not met for a small number of encryption rounds. Therefore, there must be a reasonable compromise between cryptographic strength and encryption productivity. Particularly, the compromise solution implies the determination of a sufficient number of encryption rounds at the selected round transformation. In this paper, we compared the potential of different classes of block algorithms in terms of achieving a certain encryption productivity. We chose several classes of algorithms and proved the choice of algorithms with acceptable estimates of cryptographic strength and the highest encryption productivity.

The paper suggests a way to compare the potential of different classes of block algorithms in terms of achieving a certain encryption productivity. Several classes of similar algorithms in which the number of encryption rounds is a parameter are considered. We justify the choice of algorithms with the highest ultimate performance from several classes of algorithms with acceptable estimates of the cryptographic strength.

Denote by $N$ the set of positive integers, $n \in N$, by $V_n$ — the set of all binary vectors of length $n$, $n > 1$, by $\Gamma(g)$ — a mixing digraph of transformation $g : V_n \to V_n$, by $\exp \Gamma$ — an exponent of digraph $\Gamma$.

## 2 Theoretical evaluation of block encryption algorithms productivity

For positive integers $n$, $r$ and $m$, $n > m \geq 1$ let $R(n, r, m)$ be a class of autonomous shift registers of length $n$ over the $V_r$ with $m$ feedbacks. The class $R(2, r, 1)$ corresponds to original Feistel networks; classes $R(n, r, 1)$, $R(n, r, n/2)$ and $R(n, r, n-1)$ are associated with type-1, type-2 and type-3 of Generalized Feistel Networks respectively [2],[3], [4].

Let us consider an $h$-round block encryption algorithm with a nonlinear round transformation $g$ based on the shift register from $R(n, r, m)$. For simplicity, we imply that the register feedbacks are the same and implemented by function $f(x_1, ..., x_r)$ for $m > 1$. Denote by $\tau(f)$ the time (in sec) of calculating the value of function $f(x_1, ...., x_r)$, assuming that time is the same for all inputs; $\pi(n, r, m, h)$ – the productivity of $h$-round block algorithm based

on the shift register from $R(n, r, m)$ (bits per second).

**Proposition 1.** *If the implementation time of shift for coordinates of the binary vector is substantially less than $\tau(f)$, then*

$$\pi(n, r, m, h) \approx nr/hm\tau(f) \tag{1}$$

*Proof.* Each round of the block encryption algorithm based on the shift register from $R(n, r, m)$ consists of $n$ operations over the set of vectors from $V_r$: $m$ operations of calculation of the values of the function $f(x_1, \ldots, x_r)$ and $n - m$ operations of shifts. Therefore, under these conditions the runtime of one round can be estimated by $m\tau(f)$ and the runtime of $h$ rounds is estimated by $hm\tau(f)$. The output of this encryption algorithm is $nr$-bit vector. Hence, the productivity of $h$-round algorithm based on the shift register from $R(n, r, m)$ is estimated by the specified value. □

An important property of the block algorithm is the dependence of each bit of the output block on all bits of the input block. Let us call this property the complete mixing of the inputs. If the round keys are mixed with the data using the XOR operation, then the mixing matrices of all rounds of the algorithm coincide. An important characteristic of such algorithms is the exponent of the mixing matrix of the round substitution (for any fixed key). The calculation of the exponent makes it possible to substantially narrow down the search for the least number of rounds after which the algorithm can achieve a full mixing of inputs. Similar concepts are also defined for the decryption algorithm. We can talk about full mixing of inputs by the decryption algorithm. For brevity, we assume that the $h$-round block encryption (decryption) algorithm provides the full mixing of inputs if $\exp \Gamma(g) \leq h$ (or $\exp \Gamma(g^{-1}) \leq h$), where $g$ is the round transformation. In fact, these inequalities are only necessary conditions for the full mixing. Let $h = s + h - s$, where $s = 1, \ldots, h - 1$. The analysis shows that the $h$-round block encryption algorithm is resistant to the meet-in-the-middle attack only if the $s$-round encryption algorithm or $(h - s)$-round decryption algorithm provides the full mixing of inputs. The resistance to any method is equivalent to the statement: the computational complexity of this method is not less than the computational complexity of a brute force attack. Hence, we get the necessary condition (in the form of a lower bound) for the resistance of the $h$-round block encryption algorithm to the meet-in-the-middle attack:

$$h \geq \exp \Gamma(g) + \exp \Gamma(g^{-1}) - 1. \tag{2}$$

In this regard, let us assume that the $h$-round block encryption algorithm

has the property of full two-way mixing if relation (2) holds. Let us denote by $v(g, n, r, m)$ the highest productivity of the block encryption algorithm based on the shift register from $R(n, r, m)$ with a round transformation $g$ provided the full two-way mixing.

**Proposition 2.** *The following relation holds:* $v(n, r, m, g) \approx nr/h_0 m\tau(f)$, *where* $h_0 = \exp\Gamma(g) + \exp\Gamma(g^{-1}) - 1$.

*Proof.* The higher the performance of the $h$-round block encryption algorithm is the fewer rounds are implemented. The least number of rounds provided the full two-way mixing is equal to $h_0$. Hence, we obtain the required estimation in accordance with Proposition 1. □

Note that in practice the lower bound (2) can not be achieved. So the ultimate productivity specified in Proposition 2 can not be achieved too. The mixing digraph $\Gamma(g)$ of the round transformation $g$ has $nr$ vertices — in practical cases, at least 64 and can reach the value of 1024 or more. For large values of $nr$ (for example $n \geq 8$, $r \geq 32$) it is convenient (in terms of evaluation) to additionally consider the block mixing digraph $\Gamma_B(g)$ with $n$ vertices. The value of the exponent of block mixing digraph is easier to calculate. In our case ( $8 \leq n \leq 32$) the state of shift register is divided into $r$-bit blocks $B_1, \ldots, B_n$ and $\Gamma_B(g)$ has arc $(i, j)$ if and only if the output of $B_j$ essentially depends on input of $B_i, i, j \in \{1, \ldots, n\}$). In accordance with the definition $\exp\Gamma_B(g) \leq \exp\Gamma(g)$, these values are very close in many cases. Hence, the upper bound for the maximum productivity of block encryption algorithm with the round transformation $g$ based on the shift register from $R(n, r, m)$ and provided the complete two-way mixing is

$$v(n, r, m, g) \leq nr/h_b m\tau(f), \tag{3}$$

where $h_b = \exp\Gamma_B(g) + \exp\Gamma_B(g^{-1}) - 1$.

The maximum productivity of encryption algorithms based on shift registers from $R(n, r, m)$ is not the same, it depends on the characteristics of the round transformation $g$. So the relevant problem is the description of shift registers from $R(n, r, m)$ with the marginal productivity close to the maximum. Important tasks are as follows: 1) choice of the feedback function with the lowest value of $\tau(f)$; 2) determination of the ratio of $n$ and $m$, such that the value of $h_0 m$ is the least (with increasing the number $m$ of feedbacks, the value of $h_0$ has an obvious tendency to decrease).

## 3 Algorithms based on Generalized Feistel Networks

We considered algorithms from the $R(8,32,3)$, $R(15,32,5)$, $R(16,32,5)$, $R(30,32,9)$ and $R(32,32,9)$ classes, established some properties of these algorithms and evaluated the maximum encryption productivity.

### 3.1 Requirements for the round transformations from $R(n,32,m)$ and the construction of the encryption algorithm

The round function must provide a number of properties such as: bijectivity, nonlinearity of all the coordinate functions, the best (or close to) mixing properties. The list does not pretend to be complete.

Let us consider a register transformation over $V_{32n}$ with feedback functions $(f(S,q_j) \boxplus X_k), 1 \le j \le m,\ k \in \{1,\dots,n\}$, where $\boxplus$ — addition modulo $2^{32}$, $S$ — the sum modulo $2^{32}$ of the some subblocks of the input block $X$, $q_j$ – the round key at round $j$. The confusion function is implemented similarly to the functions in the GOST 28147-89 algorithm:

$$f(S,q_j) = T^{11}(W_{8,4}(S \boxplus q_j)), \tag{4}$$

where $T^{11}$ — 11-bit circular shift to most significant bits, $W_{8,4}$ — the value of eight 4-bit $s$-boxes of GOST 28147-89. Experimental evaluation showed that the productivity of the encryption is close to the maximum when the number $m$ of feedback functions defined by the equation $m = \lceil n/4 \rceil + 1$.

As an example, we describe the 256-3 and 480-5 algorithms from the $R(8,32,3)$ and $R(15,32,5)$ classes respectively. In the 256-3 algorithm, the $i$-th round implements the transformation $g(q_1^i, q_2^i, q_3^i)$ depending on the round keys $q_1^i, q_2^i, q_3^i \in V_{32},\ 1 \le i \le h$. Let $X = (X_0, X_1, \dots, X_7)$ and $Y = (Y_0, Y_1, \dots, Y_7)$ be input and output blocks of the round respectively, $X_k, Y_k \in V_{32},\ 0 \le k \le 7$. In case of the $q_1^i, q_2^i, q_3^i$ round keys, the direct and inverse round transformation based on the shift register of length 8 over $V_{32}$ are described by formulas (see fig.3.1):

$$g(q_1^i, q_2^i, q_3^i)(X_0,\dots,X_7) = \\ (X_1, f(S,q_1) \boxplus X_2, X_3, X_4, f(S,q_2) \boxplus X_5, X_6, X_7, f(S,q_3) \boxplus X_0), \tag{5}$$

$$g^{-1}(q_1^i, q_2^i, q_3^i)(Y_0,\dots,Y_7) = \\ = (f(S',q_3) \boxplus Y_7, Y_0, f(S',q_1) \boxplus Y_1, Y_2, Y_3, f(S',q_2) \boxplus Y_4, Y_5, Y_6), \tag{6}$$

where $S = (X_1 \boxplus X_3 \boxplus X_4 \boxplus X_6 \boxplus X_7)$, $S' = (Y_0, Y_2, Y_3, Y_5, Y_6)$.

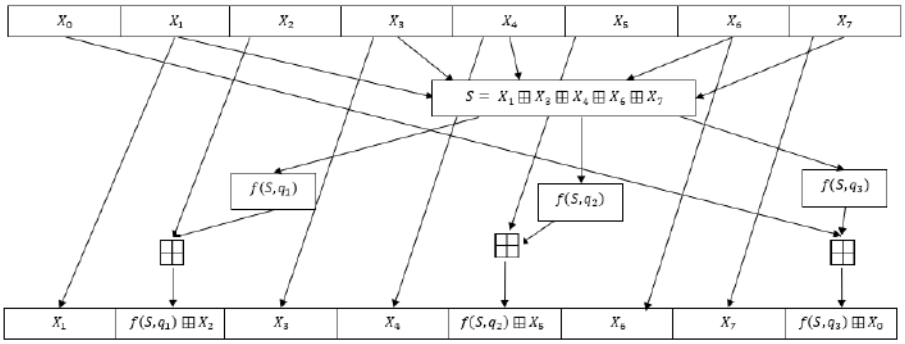



Figure 1: Round of the 256-3 algorithm

In case of the $q_1, \ldots, q_5$ round keys, the direct and inverse round transformation based on the shift register of length 15 over the set $V_{32}$ are described by formulas (see fig.2):

$$g(q_1, \ldots, q_5)(X_0, \ldots, X_{14}) = \\ (X_1, f(S, q_1) \boxplus X_2, X_3, X_4, f(S, q_2) \boxplus X_5, X_6, X_7, f(S, q_3) \boxplus X_8, \\ X_9, X_{10}, f(S, q_4) \boxplus X_{11}, X_{12}, X_{13}, f(S, q_5) \boxplus X_{14}, X_0) \quad (7)$$

$$g^{-1}(q_1, \ldots, q_5)(Y_0, \ldots, Y_{14}) = (Y_{14}, Y_0, f(S', q_1) \boxplus Y_1, Y_2, Y_3, f(S', q_2) \boxplus \\ Y_4, Y_5, Y_6, f(S', q_3) \boxplus Y_7, Y_8, Y_9, f(S', q_4) \boxplus Y_{10}, Y_{11}, Y_{12}, f(S', q_5) \boxplus Y_{13}), \quad (8)$$

where $S = X_0 \boxplus X_1 \boxplus X_3 \boxplus X_4 \boxplus X_6 \boxplus X_7 \boxplus X_9 \boxplus X_{10} \boxplus X_{12} \boxplus X_{13}$, $S' = Y_0 \boxplus Y_2 \boxplus Y_3 \boxplus Y_5 \boxplus Y_6 \boxplus Y_8 \boxplus Y_9 \boxplus Y_{11} \boxplus Y_{12} \boxplus Y_{14}$.

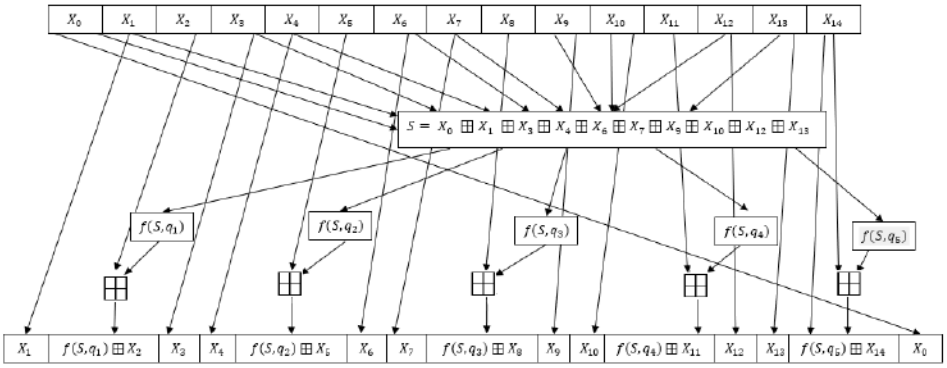



Figure 2: Round of the 480-5 algorithm

To prove the bijectivity of the round transformation of both ciphers, it is enough to show the injectivity. For example, it follows from (5) that $g(q_1, q_2, q_3)(X_0, \ldots, X_7) \neq g(q_1, q_2, q_3)(X'_0, \ldots, X'_7)$, if $(X_0, \ldots, X_7) \neq$ $\neq (X'_0, \ldots, X'_7)$. Definitely, if $(X_1, X_3, X_4, X_6, X_7) \neq (X'_1, X'_3, X'_4, X'_6, X'_7)$, then the inequality is achieved for the 0th, 2th, 3th, 5th, 6th components of

values. Otherwise, if $(X_0, X_2, X_5) \neq (X'_0, X'_2, X'_5)$, inequality is true for the 1st, 4th, 7th components.

The bijectivity of the 480-5 cipher round transformation is proved similarly. The nonlinearity of round transformations follows from the properties of the feedback function.

### 3.2 Evaluation of mixing properties

Mixing properties of algorithms are evaluated by exponents of block mixing digraphs $\Gamma_B(g(q_1, \ldots, q_m))$ and $\Gamma_B(g^{-1}(q'_1, \ldots, q'_m))$ for round transformation $g(q_1, \ldots, q_m)$ and its inverse transformation. For an exponent of a strongly connected digraph $\Gamma$ with loops the value which directly follows from the Theorem 2 in [5, p. 121-122] is true:

$$\exp \Gamma \leq \max_{i,j \in \{1,\ldots,n\}} \min_{p \in \Pi} d_{i,p,j}, \tag{9}$$

where $\Pi$ — the set of vertices with a loop in the digraph $\Gamma$, $d_{i,p,j}$ — the length of the shortest path from $i$ to $j$ passing through the vertex $p$, where $i, j, p \in \{1, \ldots, n\}$. The exponents of mixing digraphs of direct and inverse transformation of the 256-3 algorithm are measured using (9). For clarity of representation of block mixing digraphs, we use: bold arrows $(i, S)$ in figure 3 are not the arcs of the digraph and indicate that the subblock $X_i$ is used to calculate the sum $S \mod 2^{32}$ in accordance with the formula (4); arcs $(S, j)$ indicate that the sum $S \mod 2^{32}$ is used to calculate $X_j$ using the formula (4), $0 \leq i, j \leq 7$. Thus, an arc in digraph is either a simple arrow (not bold), or a concatenation of the bold arrow with a simple arrow. For example, the shortest path from 0 to 0 is the path (0,7,1,0) of length 3 (the arc (7,1) is the concatenation of the bold arrow (7, $S$) with the simple arrow ($S$, 1)).

Using (5) and (6) we built the digraphs $\Gamma_B(g)$ and $\Gamma_B(g^{-1})$ (Fig. 3) and calculated the values $\min\limits_{p \in \Pi} d_{i,p,j}$, $0 \leq i, j \leq 7$ (table 1). It is obtained that $\exp \Gamma_B(g) = \exp \Gamma_B(g^{-1}) = 4$. In accordance with (3) the number of encryption rounds must be not less than $h_b$, where $h_b \geq 7$.

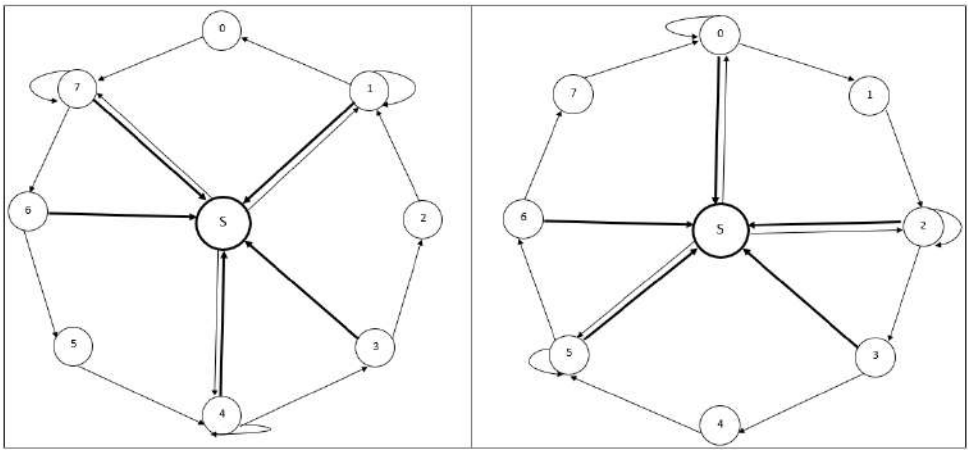



Figure 3: Mixing block digraphs $\Gamma_B(g)$ and $\Gamma_B(g^{-1})$ of the 256-3 algorithm

| | digraph $\Gamma_B(g)$ | | | | | | | | digraph $\Gamma_B(g^{-1})$ | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $j$ / $i$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | **3** | **2** | **4** | **3** | **2** | **3** | **2** | **1** | 0 | 1 | 1 | 2 | 3 | 1 | 2 | 3 |
| 1 | **1** | **0** | **3** | **2** | **1** | **3** | **2** | **1** | 2 | 3 | 1 | 2 | 3 | 2 | 3 | 4 |
| 2 | **2** | **1** | **4** | **3** | **2** | **4** | **3** | **2** | 1 | 2 | 0 | 1 | 2 | 1 | 2 | 3 |
| 3 | **3** | **2** | **1** | **2** | **1** | **3** | **2** | **1** | 1 | 2 | 1 | 2 | 1 | 2 | 2 | 3 |
| 4 | **2** | **1** | **2** | **1** | **0** | **3** | **2** | **1** | 2 | 3 | 2 | 3 | 4 | 1 | 2 | 3 |
| 5 | **3** | **2** | **3** | **2** | **1** | **4** | **3** | **2** | 1 | 2 | 1 | 2 | 3 | 0 | 1 | 2 |
| 6 | **2** | **1** | **3** | **2** | **1** | **1** | **2** | **1** | 1 | 2 | 1 | 2 | 3 | 1 | 2 | 1 |
| 7 | **2** | **1** | **3** | **2** | **1** | **2** | **1** | **0** | 1 | 2 | 2 | 3 | 4 | 2 | 3 | 4 |

Table 1: The length of the shortest path $\min\limits_{p\in\Pi} d_{i,p,j}$ of the 256-3 algorithm

Similar estimates are obtained for the 480-5 algorithm using (7) and (8). The bold arrows $(i, S)$ and simple arcs $(S, j)$ in figure 4 are defined in the same way as figure 3, $0 \le i, j \le 14$.

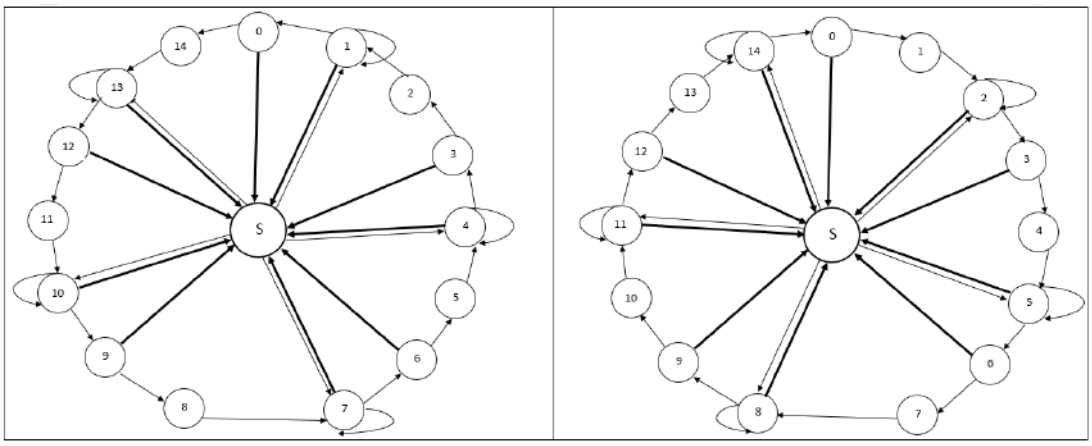



Figure 4: Mixing digraphs $\Gamma_B(g)$ and $\Gamma_B(g^{-1})$ of the 480-5 algorithm

| | digraph $\Gamma_B(g)$ | | | | | | | | | | | | | | | digraph $\Gamma_B(g^{-1})$ | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 0 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 1 | 2 | 1 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 |
| 1 | 1 | 0 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 2 | 3 | 4 | 2 | 3 | 4 | 2 | 3 | 4 | 2 |
| 2 | 2 | 1 | 4 | 3 | 2 | 4 | 3 | 2 | 4 | 3 | 2 | 4 | 3 | 2 | 3 | 2 | 3 | 0 | 1 | 2 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 |
| 3 | 2 | 1 | 1 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 3 | 1 | 2 | 1 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 |
| 4 | 2 | 1 | 2 | 1 | 0 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 3 | 4 | 2 | 3 | 4 | 1 | 2 | 3 | 2 | 3 | 4 | 2 | 3 | 4 | 2 |
| 5 | 3 | 2 | 3 | 2 | 1 | 4 | 3 | 2 | 4 | 3 | 2 | 4 | 3 | 2 | 4 | 2 | 3 | 1 | 2 | 3 | 0 | 1 | 2 | 1 | 2 | 3 | 1 | 2 | 3 | 1 |
| 6 | 2 | 1 | 3 | 2 | 1 | 1 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 1 | 1 | 2 | 3 | 1 | 2 | 3 | 1 |
| 7 | 2 | 1 | 3 | 2 | 1 | 2 | 1 | 0 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 3 | 4 | 2 | 3 | 4 | 2 | 3 | 4 | 1 | 2 | 3 | 2 | 3 | 4 | 2 |
| 8 | 3 | 2 | 4 | 3 | 2 | 3 | 2 | 1 | 4 | 3 | 2 | 4 | 3 | 2 | 4 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 0 | 1 | 2 | 1 | 2 | 3 | 1 |
| 9 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 1 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 1 | 1 | 2 | 3 | 1 |
| 10 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 2 | 1 | 0 | 3 | 2 | 1 | 3 | 3 | 4 | 2 | 3 | 4 | 2 | 3 | 4 | 2 | 3 | 4 | 1 | 2 | 3 | 2 |
| 11 | 3 | 2 | 4 | 3 | 2 | 4 | 3 | 2 | 3 | 2 | 1 | 4 | 3 | 2 | 4 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 0 | 1 | 2 | 1 |
| 12 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 1 | 2 | 1 | 3 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 1 | 1 |
| 13 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 3 | 2 | 1 | 2 | 1 | 0 | 3 | 2 | 3 | 2 | 3 | 4 | 2 | 3 | 4 | 2 | 3 | 4 | 2 | 3 | 4 | 1 |
| 14 | 3 | 2 | 4 | 3 | 2 | 4 | 3 | 2 | 4 | 3 | 2 | 3 | 2 | 1 | 4 | 1 | 2 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 0 |

Table 2: The length of the shortest path $\min\limits_{p\in\Pi} d_{i,p,j}$ of the 480-5 algorithm

Using (5) and (6) we built the digraphs $\Gamma_B(g)$ and $\Gamma_B(g^{-1})$ (Fig.4) and calculated the values $\min\limits_{p\in\Pi} d_{i,p,j}$, $0 \le i,j \le 14$ (table 2). It is obtained that $\exp\Gamma_B(g) = \exp\Gamma_B(g^{-1}) = 5$. In accordance with (3) the number of encryption rounds must be not less than $h_b$, where $h_b \ge 9$.

### 3.3 Evaluation of productivity characteristics of algorithms

Round transformations of block encryption algorithms from the $R(8,32,3)$, $R(15,32,5)$, $R(16,32,5)$, $R(30,32,9)$ and $R(32,32,9)$ classes are investigated by the methods presented in subsection 3.2. A number of properties of the corresponding algorithms are defined, including the characteristics of the maximum encryption productivity. Table 3 presents preliminary estimation of the ratio of the time of software implementations. The number of rounds is calculated by the formula $h_b = \exp\Gamma_B(g) + \exp\Gamma_B(g^{-1}) - 1$. The evaluation is given in comparison with GOST 28147-89 and carried out in two ways: by formulas (1) and (3) and as the ratio of times of the software implementations. In the second case, we use ECB-mode for encryption of the same plaintext of length $T = 21120$ bytes.

Let's explain the number of rounds 17 for GOST 28147-89 in Table 3. For all proposed algorithms, it is assumed that the key schedule provides the dependence of each bit of each round key on all the bits of the encryption key. This assumption does not correct for GOST 28147-89, because the bits of encryption key are mixed only for the first 8 rounds. Therefore, for the block algorithms considered, the model of mixing the inputs is adequate, for the algorithm GOST 28147-89 it is necessary to use the mathematical model of mixing the bits of the encryption key, not the inputs. According to this

model, the full mixing is provided after 9 rounds, both for the encryption and for the decryption, that means that $h_b = 17$.

| Algorithm | The number of rounds for the full mixing of inputs, $h_b$ | The gain in productivity when comparing the estimates (3), number of times | The gain in productivity when comparing the time of software implementation, number of times |
|---|---|---|---|
| GOST 28147-89 | 17 | 1 | 1 |
| 256-3 | 7 | 3.238 | 3.635 |
| 480-5 | 7 | 3.643 | 4.008 |
| 512-5 | 9 | 3.022 | 3.444 |
| 960-9 | 9 | 3.148 | 3.522 |
| 1024-9 | 9 | 3.358 | 3.956 |

Table 3. Comparison of productivity characteristics for different algorithms

The estimates show that all algorithms have a higher marginal productivity than GOST 28147-89. This means that increasing the size of input blocks increases productivity despite increasing the number of feedbacks. The 480-5 algorithm has the highest maximum productivity. One of the factors why 480-5 has the highest value is that the number of subblocks of length 32 ($n = 15$) in the input block of this algorithm is a multiple of the number of feedbacks ($m = \lceil n/4 \rceil + 1 = 5$), which allows achieving the smallest value of the exponent of the mixing matrix of the round transformation.

## 4 Conclusion

In this paper, we proposed a characteristic of the maximum productivity of block encryption algorithms with the minimum number of rounds required for the two-way mixing of the input data by the encryption algorithm. The proposed characteristic depends on such parameters as the size of data blocks, the speed of the round function and the exponent of the mixing digraph of the round transformation. This productivity characteristic can be used by developers to determine the parameters for block encryption algorithms. The obtained evaluations show that with increasing the input block size up to 1024 bits (the number of feedbacks of high-performance algorithms grows more slowly, reaching 9), the maximum encryption productivity grows, but slower than the block size.

## 5 Acknowledgments

The author Vladimir Fomichev gratefully acknowledges the support of Russian Foundation for Basic Research grant 16-01-00226.

## References

[1] Fomichev V. M., Melnikov D. A. Kriptograficheskie metody zaschity informatsii [Cryptographic methods of information security], ed. Fomichev V. M., URAIT, Moscow, 454 pp., 2016. In Russian.

[2] Berger T. P., Francq J., Minier M., Thomas G. Extended Generalized Feistel Networks Using Matrix Representation to Propose a New Lightweight Block Cipher: Lilliput. IEEE Transactions on Computers, 65 (7), pp. 2074–2089. 2016.

[3] Berger T. P., Minier M., Thomas G. Extended Generalized Feistel Networks Using Matrix Representation. LNCS, SAC 2013, 8282, Springer, pp. 289–305. 2014.

[4] Suzaki T., Minematsu K. Improving the Generalized Feistel. LNCS, FSE 2010, 6147, eds. S. Hong, Iwata T., Springer, Heidelberg, pp. 19–39. 2010.

[5] Fomichev V. M. Properties of paths in graphs and multigraphs. Prikladnaya Diskretnaya Matematika, 7(1), pp. 118–124. 2010. In Russian.

# Security Bounds for Standardized Internally Re-keyed Block Cipher Modes and Their Practical Significance

Liliya Ahmetzyanova, Evgeny Alekseev, Grigory Karpunin, Igor Oshkin, Grigory Sedov, Stanislav Smyshlyaev, and Ekaterina Smyshlyaeva

Crypto-Pro LLC, Moscow, Russia
{lah, alekseev, karpunin, oshkin, sedovgk, svs, ess}@cryptopro.ru

**Abstract**

In 2018 the CTR-ACPKM and OMAC-ACPKM-Master internally re-keyed block cipher modes were adopted in Russian Standardization System and now they are passing through the last formal standardization stages in IETF. The main distinctive feature of these modes is that during each message processing, the key, used for data blocks transformation, is periodically changed. In the current paper we obtain the security bounds for these modes in the standard IND-CPNA and PRF security models. Particular attention is paid to the interpretation of the obtained reductions from the viewpoint of mode resistance to the cryptanalytic methods of various types.

**Keywords:** CTR, OMAC, re-keying, block cipher modes, provable security.

## 1 Introduction

The effectiveness of many cryptanalytic methods (see, e.g. [21, 13, 23]) depends heavily on amount of data processed under a single key, therefore this amount of data should be limited. A certain maximal amount of data, which can be safely encrypted under a single key, is called «key lifetime». The trivial way to increasing the key lifetime (such as renegotiation) can reduce the total performance due to termination of application data transmission, the use of random number generators and many other resource-intensive additional calculations.

For the protocols based on block ciphers an efficient way to increasing the key lifetime is to use various re-keying mechanisms. Re-keying mechanisms can be applied on the different protocol levels: on the block cipher level (*fresh re-keying* [16]), on the block cipher mode of operation level (*internal re-keying* [7]), and on the message processing level (*external re-keying* [7]). From the viewpoint of cryptographic and operational properties each of these

approaches has its own advantages and disadvantages (see [25] for details). For instance, the external re-keying approach doesn't require the development of new block ciphers or modes of operation and allows to apply a quite simple modular security analysis for resulting protocol [5], while the fresh re-keying approach often changes the internal structure of the used block cipher, that requires a nontrivial security analysis [16]. In the Russian protocols, the internal re-keying approach is widely spread. For example, it is used in the protocols TLS [3], IPsec [1], CMS [2]. This approach is associated with the development of a special class of *internally re-keyed block cipher modes of operation*. Their main feature is that during each message processing, the key, used for data blocks transformation, is periodically changed.

The current paper contains the security analysis of the CTR-ACPKM and OMAC-ACPKM-Master internally re-keyed modes, which were adopted in Russian Standardization System and are currently being considered in IETF. In addition, the CTR-ACPKM mode is also supposed to be used in the Russian ciphersuites of the TLS 1.2 protocol [4]. The analysis of the CTR-ACPKM and OMAC-ACPKM-Master modes was carried out in the paradigm of provable security, in other words, lower security bounds were obtained in security models relevant for encryption modes (the IND-CPNA model [10, 24]) and message authentication modes (the PRF model [11]). The obtained bounds show that, under security of the used block cipher, the cryptographic properties of the CTR-ACPKM and OMAC-ACPKM-Master modes are improved compared to the properties of the associated CTR and OMAC modes without internal re-keying.

In addition to the concrete lower security bounds, this paper also contains arguments about the meaning of these bounds from the viewpoint of attacks applied to cryptographic constructions of these types. So, for each obtained bound we show the relation between proof framework and resistance to the attacks based on modes architectural features. These arguments show that the heuristic (based on the concrete cryptanalytic methods) and theoretical-complexity approaches to security analysis of cryptographic protocols are not contradictory to each other and, according to M. Bellare [12], «emerge as opposite sides of the same coin, and complement each other».

## 2 Preliminaries and Basic Security Notions

By $\{0,1\}^u$ we denote the set of $u$-component bit strings and by $\{0,1\}^*$ we denote the set of all bit strings of finite length. Let $0^u$ be the string, consisting of $u$ zeros. For bit strings $U$ and $V$ we denote by $U\|V$ their concatenation.

Let $|U|$ be the bit length of the string $U$. We denote by $|U|_u = \lceil |U|/u \rceil$ the length of the string $U$ in $u$-bit blocks.

For a bit string $U$ and a positive integer $l \leqslant |U|$ let $\mathrm{msb}_l(U)$ ($\mathrm{lsb}_l(U)$) be the string, consisting of the leftmost (rightmost) $l$ bits of $U$. For nonnegative integers $l$ and $i$ let $\mathrm{str}_l(i)$ be $l$-bit representation of $i$ with the least significant bit on the right. For a nonnegative integer $l$ and a bit string $U \in \{0,1\}^l$ let $\mathrm{int}(U)$ be an integer $i$ such that $\mathrm{str}_l(i) = U$. Let $\mathrm{Inc}(U)$ be the function, which takes the input $U \in \{0,1\}^u$ and outputs the string $\mathrm{msb}_{u/2}(U) \| \mathrm{str}_{u/2}(\mathrm{int}(\mathrm{lsb}_{u/2}(U)) + 1 \bmod 2^{u/2})$. For nonnegative integers $u$ let $\mathrm{pad}_u(U)$ be the function which takes $U \in \{0,1\}^*$ and outputs the string $U$, if $u \mid |U|$, and the string $U \| 10^{u|U|_u - |U| - 1}$, otherwise.

For any set $S$, define $Perm(S)$ as the set of all bijective mappings on $S$ (permutations on $S$), and $Func(S)$ as the set of all mappings from $S$ to $S$. A block cipher $E$ (or just a cipher) with block size $n$ and key size $k$ is the permutation family $\left(E_K \in Perm(\{0,1\}^n) \mid K \in \{0,1\}^k\right)$, where $K$ is a key. If the value $s$ is chosen from a set $S$ uniformly at random, then we denote $s \xleftarrow{\mathcal{U}} S$.

For a bit string $U$ we denote by $U[i] \in \{0,1\}^n$, $1 \leqslant i \leqslant \lceil |U|/n \rceil - 1$, and $U[\lceil |U|/n \rceil] \in \{0,1\}^h$, $h \leqslant n$, such strings that $U = U[1] \| U[2] \| \ldots \| U[\lceil |U|/n \rceil]$ and call them «blocks» of the string $U$.

We model an adversary using an interactive probabilistic algorithm that has access to one or more oracles. The resources of $A$ are measured in terms of time and query complexities. For a fixed model of computation and a method of encoding the time complexity includes the description size of $A$. The query complexity usually includes the number of queries and the maximal length of queries or the total length of queries. Denote by $\mathsf{Adv}^{\mathrm{M}}_{\mathrm{S}}(A)$ the measure of the success of the adversary $A$ in realizing a certain threat, defined by the model M, for the cryptographic scheme S. The formal definition of this measure will be given in each specific case.

A block cipher is usually regarded as a family of permutations, which on its own does not provide such application-level security properties as integrity, confidentiality or authenticity (see, e.g. [9]). The cipher is usually used as a base function for constructing other schemes or protocols that solve the above-mentioned cryptographic challenges. The security of such constructions is proven under assumption that the block cipher is secure. In a paradigm of the practice-oriented provable security (see [12]) we should quantify the security as a function of the used cipher security for appropriate models.

Standard security model for block ciphers is PRP-CPA («Pseudo Ran-

dom Permutation under Chosen Plaintext Attack») (see, e.g. [9]). The formal definition can be found in Appendix A. In the case of the block cipher with no attacks known, the advantage $\mathsf{Adv}_E^{\text{PRP-CPA}}(A)$ is estimated by the characteristics of the general exhaustive key search attack, so $\mathsf{Adv}_E^{\text{PRP-CPA}}(A) \leqslant t/2^k$ for any adversary $\mathcal{A}$ with the time complexity at most $t$, making at least $\lceil k/n \rceil$ queries.

## 3 Internal Re-keying

Internal re-keying is an approach to increasing the key lifetime by using a transformation of a data processing key (a section key) during each separate message processing. Such key transformation mechanisms are built into the particular base mode of operation and depend heavily on the internal features of its structure, therefore they are called «internal» re-keying mechanisms.

Each message is processed starting with the same key (the first section key) and each section key is updated using the certain key update technique after processing certain amount of message blocks (a section). The mode parameter, hereinafter called section size and denoted as $N$, is measured in blocks and is fixed within a specific protocol depending on the requirements of the system capacity and the key lifetime. In the current paper we consider two internally re-keyed block cipher modes: the CTR-ACPKM encryption mode and the OMAC-ACPKM-Master message authentication mode.

### 3.1 Internally Re-keyed CTR-ACPKM mode

In this section we define the CTR-ACPKM mode with section size $N$ (CTR-ACPKM$_N$). The mode structure is shown in Figure 1. During the processing of the input plaintext $P$ of the block length $m = |P|_n$ under the initial key $K$ the message is divided into $l = \lceil m/N \rceil$ sections (denoted as $P = P^1 \| P^2 \| ... \| P^l$, where $P^i \in \{0,1\}^{nN}$ for $i \in \{1, 2, ..., l-1\}$, $P^l \in \{0,1\}^r$, $r \leqslant nN$). The first section of each message is processed under the section key $K^1 = K$ using the CTR subroutine. The $(i+1)$-th section of message is continued to be processed using the CTR subroutine under the $K^{i+1}$ section key, which is calculated using ACPKM transformation as follows:

$$K^{i+1} = \text{ACPKM}(K^i) = \text{msb}_k(E_{K^i}(D_1) \| \ldots \| E_{K^i}(D_s)),$$

where $s = \lceil k/n \rceil$ and $D_1, D_2, \ldots, D_s \in \{0,1\}^n$ are arbitrary pairwise different constants such that the $(n/2)$-th bit (counting from the right) side of each $D_i$ is equal to 1. The plaintext length must be at most $2^{n/2-1}$ blocks. Note that the internal state (counter) of the CTR-ACPKM$_N$ mode is not

reset for each new section and the condition on the $D_1, D_2, \ldots, D_s$ constants allows to prevent collisions of block cipher permutation inputs in cases of key transformation and message processing (see [25, 6] for details).

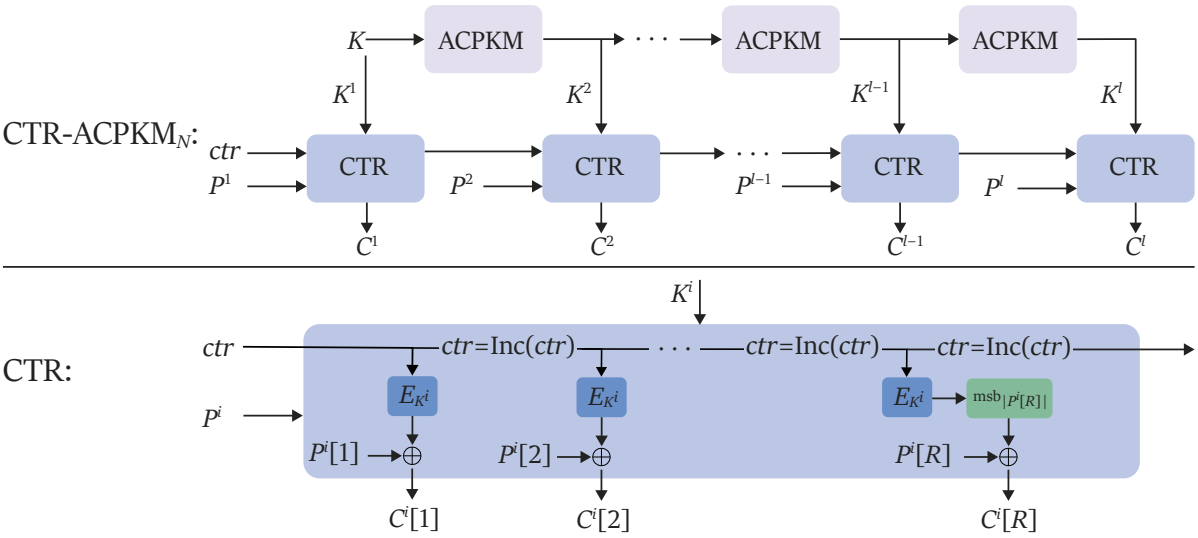



Figure 1: The CTR-ACPKM$_N$ encryption mode takes a key $K \in \{0,1\}^k$, a nonce $IV \in \{0,1\}^{n/2}$ (here $ctr = IV\|0^{n/2}$) and a plaintext $P \in \{0,1\}^*$ as inputs and returns a ciphertext $C \in \{0,1\}^{|P|}$ as an output. The ACPKM subroutine generates next section key $K^{i+1}$ using the previous section key $K^i$. The CTR subroutine processes sections $P^1, \ldots, P^l$ of the plaintext $P$ under the corresponding section keys. Each section consists of $R$ blocks, where $R = N$ for intermediate sections and $R \leqslant N$ for a final section.

### 3.2 Internally Re-keyed OMAC-ACPKM-Master mode

In this section we define the OMAC-ACPKM-Master mode with section size $N$ and master-key change frequency $T^*$ (OMAC-ACPKM-Master$_{N,T^*}$). The mode structure is shown in Figure 2. During the processing of the input message $M$ of the block length $m = |M|_n$ under the initial key $K$ the message $M$ is divided into $l = \lceil m/N \rceil$ sections (denoted as $M = M^1\|M^2\|\ldots\|M^l$, where $M^i \in \{0,1\}^{nN}$ for $i \in \{1,2,...,l-1\}$, $M^l \in \{0,1\}^r$, $r \leqslant nN$). The $j$-th section of each message is processed with the key material $K^j\|K_1^j$, $j \in \{1,...,l\}$, $K^j \in \{0,1\}^k$, $K_1^j \in \{0,1\}^n$, which is calculated with the ACPKM-Master$_{T^*}$ algorithm as follows:

$$K^1\|K_1^1\|\ldots\|K^l\|K_1^l = $$
$$= \text{ACPKM-Master}_{T^*}(K,d,l) = \text{CTR-ACPKM}_{T^*}(K, 1^{n/2}, 0^{dln}),$$

where $d = \lceil k/n \rceil + 1$. Note that the parameters $d$ and $l$ must satisfy the inequality $d \cdot l \leqslant 2^{n/2-1}$. All intermediate sections are processed using the CBCMAC subroutine and the final section is processed using the TMAC

subroutine. The message length must be at most $N \cdot \frac{2^{n/2-1}}{d}$ blocks.

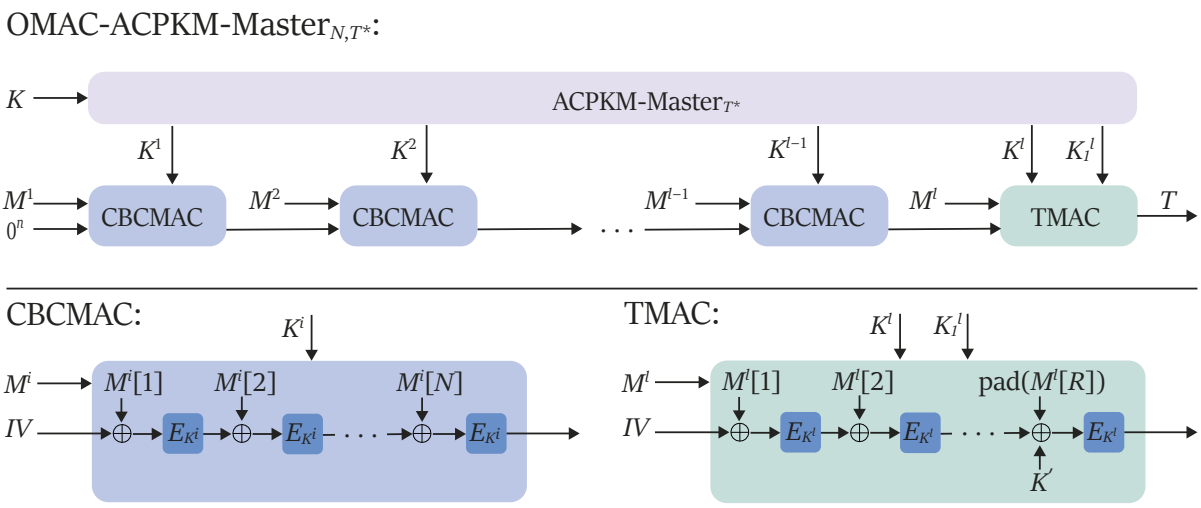



Figure 2: The OMAC-ACPKM-Master$_{N,T^*}$ authentication mode takes a key $K \in \{0,1\}^k$ and a message $M \in \{0,1\}^*$ and outputs a tag $T \in \{0,1\}^n$. The ACPKM-Master$_{T^*}$ subroutine generates section key material using the master key $K$. The CBCMAC subroutine processes the intermediate sections $M^1, \ldots, M^{l-1}$ of the message $M$. The TMAC subroutine processes the final section $M^l$, size of which must be less than a section size $nN$ (here $R = |M^l|_n$). The key $K'$ in TMAC are equal to $K_1^l$, if the last block $M^l[R]$ is complete, or to $K_1^l \cdot \alpha$, otherwise (the «$\cdot$» operation corresponds to the multiplication operation for the binary Galois field of $2^n$ elements, the $\alpha$ element is a primitive element of the corresponding field).

## 4 Security Analysis of Internally Re-keyed Modes

This section contains the security analysis results for the proposed internally re-keyed modes. This analysis was carried out in the «provable security» paradigm. In contrast to the heuristic approach the provable security approach considers the resistance of cryptographic scheme not to certain cryptanalytic methods, but to *all* methods covered by the used security model. The security bounds, obtained as a result of such an analysis, allow to predict worst-case methods and, basing on this prediction, to limit the data available to the adversary for achieving necessary safety margin for real systems.

The formal proofs of the obtained security bounds are presented in Appendix D.1, D.2. In the main part of the current paper we present the arguments about meaning of the proofs stages from the viewpoint of resistance to possible methods. Inclusion of these arguments in the paper is due to a significant number of issues related to this topic, which generate great interest around the world (see, e.g. [22, 19, 17, 15]). The interpretation given below is intended to deepen understanding of the so called «provable security» con-

cept. However, this interpretation can not be considered as a new tool for security analysis. Such an analysis should be carried only by constructing a rigorous theoretic-complexity reduction.

The main idea behind the interpretations presented in paragraphs marked «Practical meaning» is as follows. We consider the obtained rigorous reduction as a procedure of identifying several certain properties of the analyzed cryptographic scheme, at least one of which any method, covered by the used security model, has to exploit. This identifying is not unique and different reductions can lead to the different security bounds. Therefore, the proof framework should ideally be based on an optimal reduction which leads to the tight security bounds (for such a bound there exists a certain method which success probability and computational resources are exactly on the boundary). Having defined these properties, we obtain the target bound by summing the success probability bounds for methods exploiting one property only.

We've paid a special attention to the proof sketch of the the OMAC-ACPKM-Master mode due to its obscurity and multistage while the proof for CTR-ACPKM is much more transparent and requires much smaller additional explainations. Moreover, the arguments similar to the OMAC-ACPKM-Master mode can be applied to CTR-ACPKM too.

### 4.1 Security Analysis of OMAC-ACPKM-Master mode

The security analysis of the OMAC-ACPKM-Master mode has been carried out in the PRF («Pseudorandom Function») model, which is strictly formalized in Appendix D.2. Informally, in this model the adversary has to distinguish the target mode under a random unknown key from a «truly» random function, having the capability to adaptively choose messages and obtain their tags (in terms of model — «make queries»). The distinguishability threat, considered in the model, is «easier» to implement than the other more intuitively understandable threats, such as key recovery or typical forgeries (universal, selective, existential). Therefore, the PRF model is considered as the most relevant (known at the time) for the deterministic authentication modes [11].

**Theorem 1.** *Let $N$ and $T^*$ be the parameters of OMAC-ACPKM-Master mode. Then for any adversary $A$ with time complexity at most $t$ that makes queries, where the maximal message length is at most $m$ blocks and the total*

*message length is at most $\sigma$ blocks, there exists an adversary $B$ such that*

$$\mathsf{Adv}^{PRF}_{OMAC\text{-}ACPKM\text{-}Master_{N,T^*}}(A) \leqslant \left(l + \left\lceil \frac{dl}{T^*} \right\rceil\right) \cdot \mathsf{Adv}^{PRP\text{-}CPA}_{E}(B) + f(d, l, T^*) + \\ + \frac{4\left(\sigma_1^2 + \ldots + \sigma_l^2\right)}{2^n},$$

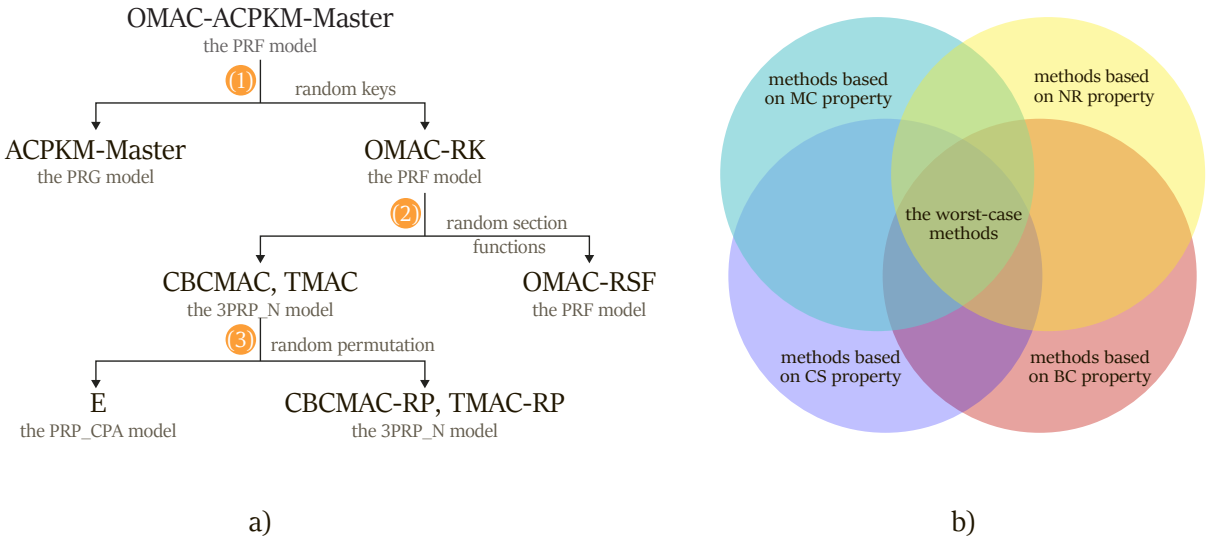
*where*

$$f(d, l, T^*) = \begin{cases} \left\lceil \dfrac{dl}{T^*} \right\rceil \dfrac{(T^* + d - 1)^2}{2^n}, & \text{if } \lceil dl/T^* \rceil > 1, \\ \dfrac{(dl)^2}{2^n}, & \text{if } \lceil dl/T^* \rceil = 1. \end{cases}$$

*$d = \lceil k/n \rceil + 1$, $l = \lceil m/N \rceil$, $dl \leqslant 2^{n/2-1}$, $\sigma_j$ is the total block length of data processed under the section key $K^j$ and $\sigma_j \leqslant 2^{n-1}$, $\sigma_1 + \ldots + \sigma_l = \sigma$. The adversary $B$ makes at most $\max(\sigma_1, T^* + d - 1)$ queries. Furthermore, the time complexity of $B$ is at most $t + cn(\sigma + lT^*d)$, where $c$ is a constant that depends only on the model of computation and the method of encoding.*

The full proof can be found in Appendix D.2.

*Proof.* (sketch) The proof consists of three steps. On each step we idealize a certain component of a target mode in order to obtain a structure which is close to a truly random function at the end. The «cost» of each idealization is estimated by the advantage of the worst-case adversary, distinguishing the original component from the idealized one. This estimation is carried out by extending capabilities of the adversary which granted to access to all internal mode states except for states related to the idealized component. The idealization steps are presented in Figure 3a. The right arrows indicate the idealization results and the left arrows indicate the «costs» of the corresponding idealizations.




a) b)

Figure 3: a) Proof framework for the OMAC-ACPKM-Master mode. b) The properties crucial for security of the OMAC-ACPKM-Master mode.

**The first step.** On the step (1) in Figure 3a (Lemma 10) we idealize the ACPKM-Master section key generation subroutine, supposing that the resulting changed mode (the abstract OMAC-RK mode) operates with «truly» random section keys. This process corresponds to the reduction that shows, that the existence of any method $A$, «PRF-breaking» the target mode OMAC-ACPKM-Master, implies the existence of the following methods:

- method $B$, distinguishing the section keys, produced by ACPKM-Master, from «truly» random ones (the PRG model).
- method $C$, distinguishing the abstract OMAC-RK mode from «truly» random function (the PRF model).

The «cost» of this idealization is the advantage of the worst-case method $B$.

*Practical meaning.* The first idealization step can be interpreted as identifying of the mode property which we denote by «Non-Randomness of section keys» (**NR**). This property defines «how much» the target OMAC-ACPKM-Master mode is distinguished from the abstract OMAC-RK mode. The advantage of the worst-case method $B$ can be thought as an upper bound of the advantage of the worst-case method which «PRF-breaks» the target mode and is based only on the **NR** property.

**The second step.** On the step (2) in Figure 3a (Lemma 4), we idealize the section processing functions (CBCMACs for intermediate sections and TMACs for final sections), assuming that the outputs of each such function are independent random values. This process corresponds to the reduction that shows that the existence of any method $A$, «PRF-breaking» the target OMAC-RK mode, implies the existence of the following methods:

– method $B$, distinguishing values obtained after any section processing by CBCMAC or TMAC, from random ones (the 3PRF model).
– method $C$, distinguishing the abstract OMAC-RSF mode, where all sections are processed by independent «truly» random functions, from one «truly» random function (the PRF model). Note that the OMAC-RSF mode is a combinatorial object, the PRF-insecurity of which can be estimated from the theoretic-informational point of view.

The «cost» of this idealization is the advantage of the worst-case method $B$ which is estimated at the third step.

*Practical meaning.* The second idealization step result can be interpreted as identifying of the mode property which we denote by «Correlation between Sections» (**CS**). This property defines «how much» the OMAC-RSF mode is distinguished from the «truly» random function. The advantage of the worst-case method $C$ can be thought as an upper bound of the advantage of the worst-case method which PRF-breaks the target mode and is based only on the **CS** property.

**The third step.** On the last step, (3) in Figure 3a (Lemma 7) for estimating the «cost» of the second idealization, we idealize the used block cipher $E$ under a random key $K^i$, considering it as a random permutation $P$. This process corresponds to the reduction that shows, that the existence of any method $A$, «3PRF-breaking» the CBCMAC and TMAC subroutines, implies the existence of the following methods:

– method $B$, distinguishing the block cipher permutation under random key from «truly» random permutation (the PRP-CPA model).
– method $C$, distinguishing section processing functions CBCMAC-RPs and TMAC-RPs under a single random permutation from *independent* truly random functions (the 3PRF model). Note that the considered task is combinatorial and its 3PRF-insecurity can be estimated from the theoretic-informational point of view.

The «cost» of this idealization is the advantage of the worst-case method $B$. This advantage is estimated using known methods of cryptanalysis (linear, differential, integral) where the adversary's capabilities are covered by the PRP-CPA model.

*Practical meaning.* Consider the mode properties which defines «how much» the OMAC-RK mode is distinguished from the abstract OMAC-RSF mode. By the third idealization (changing the used block cipher by the random permutations family) we identify the last two properties: «Block Cipher design» (**BC**) and «Mode design Combinatorics» (**MC**). Thus, the advantage of the worst-case method $B$ can be used to upper bound the advantage

of the worst-case method which PRF-breaks the target mode and is based only on the **BC** property. Similarly, the advantage of the worst-case method $C$ can be used to upper bound the advantage of the worst-case method based only on the **MC** property.

Thus, the obtained reductions show that any method covered by the PRF model should

be based on at least one of the following four mode properties: **NR**, **CS**, **BC** or **MC**. In order to obtain the total bound for advantage of the worst-case methods, «PRF-breaking» the target mode, we sum up (following the reduction) the success probabilities of the worst-case methods, which exploit only one of the properties mentioned above (see Figure 3b).

Thus, the obtained bound can be interpreted as follows:

$$\mathsf{Adv}^{\mathrm{PRF}}_{\text{OMAC-ACPKM-Master}_{N,T^*}}(A) \leqslant \underbrace{f(d,l,T^*) + \left\lceil \frac{dl}{T^*} \right\rceil \cdot \mathsf{Adv}^{\text{PRP-CPA}}_{E}(B)}_{\textbf{NR property}} + \underbrace{l \cdot \mathsf{Adv}^{\text{PRP-CPA}}_{E}(B)}_{\textbf{BC property}} + \underbrace{\frac{4\left(\sigma_1^2 + \ldots + \sigma_l^2\right)}{2^n}}_{\textbf{MC and CS properties}}.$$

□

## 4.2 Security Analysis of the CTR-ACPKM mode

The security analysis of the CTR-ACPKM mode has been carried out in the IND-CPNA («Indistinguishability under Chosen Plaintext and Nonce Attack») model, which is strictly formalized in Appendix D.1. This model is similar to the standard IND-CPA security model [10] but considers nonce-respecting adversaries [24]. Informally, in this model the adversary has to distinguish the obtained ciphertexts from the «garbage», having the capability to adaptively choose plaintexts and nonces (in a unique manner). The IND-CPNA is the strongest standard security model (known at the time) which allows to analyze the cryptographic properties of the mode from the viewpoint of computational «closeness» to the ideal one-time pad encryption [24].

**Theorem 2.** *Let $N$ be the parameter of CTR-ACPKM mode. Then for any adversary $A$ with time complexity at most $t$ that makes queries, where the maximal message length is at most $m$ ($m \leqslant 2^{n/2-1}$) blocks and the total*

*message length is at most $\sigma$ blocks, there exists an adversary $B$ such that*

$$\mathsf{Adv}^{\textit{IND-CPNA}}_{\textit{CTR-ACPKM}_N}(A) \leqslant$$
$$\leqslant l \cdot \mathsf{Adv}^{\textit{PRP-CPA}}_{E}(B) + \frac{(\sigma_1 + s)^2 + \ldots + (\sigma_{l-1} + s)^2 + (\sigma_l)^2}{2^{n+1}}$$

*where $s = \lceil k/n \rceil$, $l = \lceil m/N \rceil$, $\sigma_j$ is the total data block length processed under the section key $K^j$ and $\sigma_j \leqslant 2^{n-1}$, $\sigma_1 + \ldots + \sigma_l = \sigma$. The adversary $B$ makes at most $\sigma_1 + s$ queries. Furthermore, the time complexity of $B$ is at most $t + cn(\sigma + ls)$, where $c$ is a constant that depends only on the model of computation and the method of encoding.*

The full proof can be found in Appendix D.1.

### 4.3 Comparative bounds analysis

In the current section we consider the obtained bounds for the internally re-keyed CTR-ACPKM and OMAC-ACPKM-Master modes in more detail and compare them with the security bounds for the CTR (see [10]) and OMAC (see [18]) modes without re-keying. The bounds are presented in Table 1. The bounds for the internally re-keyed modes show that the insecurity of the modes reaches minimum if $\sigma_1 = \ldots = \sigma_l$, i.e. if all messages are of the same length.

| Mode | $\mathsf{Adv}^{\mathrm{PRF}}_{\mathrm{Mode}}(A)$ |
|---|---|
| OMAC | $\approx \frac{4\sigma^2 + 1}{2^{n+1}}$ |
| OMAC-ACPKM-Master$_{N,\infty}$ | $\approx \frac{4\,(\sigma_1^2 + \ldots + \sigma_l^2)}{2^{n+1}} + \frac{(dl)^2}{2^n}$ |
| Mode | $\mathsf{Adv}^{\mathrm{IND\text{-}CPNA}}_{\mathrm{Mode}}(A)$ |
| CTR | $\approx \frac{\sigma^2}{2^{n+1}}$ |
| CTR-ACPKM$_N$ | $\approx \frac{(\sigma_1 + s)^2 + \ldots + \sigma_l^2}{2^{n+1}}$ |

Table 1: Security bounds for the OMAC, CTR modes and the internally re-keyed CTR-ACPKM$_N$, OMAC-ACPKM-Master$_{N,\infty}$ modes with the section size $N$ (under secure block cipher). Here $s = \lceil k/n \rceil$, $d = \lceil k/n \rceil + 1$, $\sigma$ is the total plaintexts block length, $m$ is the maximal plaintext block length and $\sigma_j$ is the total block length of data, processed under the section key $K^j$ ($\sigma_1 + \ldots + \sigma_l = \sigma$, where $l = \lceil m/N \rceil$).

Consider the case $l = 1$ (no re-keying during message processing). Note, that for the CTR and CTR-ACPKM$_N$ modes the bounds are equal that is explained by total equivalence of the considered modes, while the bound for OMAC-ACPKM-Master$_{N,\infty}$ is worse than the bound for OMAC by term $\frac{d^2}{2^n}$ that is explained by additional key generation with the used block cipher.

Using the property $\sigma_1 \geqslant \sigma_2 \geqslant \ldots \geqslant \sigma_{l-1} \geqslant N$, we obtain the following restrictions on the parameters $s$ and $N$ (see Table 2) for which the bounds for internally re-keyed modes are not worse than the bound for the modes without re-keying for any parameter $\sigma$ and structure of processed messages (their number or lengths). The calculations for both modes are presented in Appendix E. Note that these restrictions cover all practical cases.

| Number of sections | CTR-ACPKM$_N \succeq$ CTR | OMAC-ACPKM-Master$_{N,\infty} \succeq$ OMAC |
|---|---|---|
| $l = 2$ | $s \leqslant \min(\sqrt{2N}, \sigma_2)$ | $d \leqslant \min(N, 2\sigma_2)$ |
| $l = 3$ | $s \leqslant \min(\sqrt{2N}, N/2)$ | $d \leqslant \min(N, 16)$ |
| $l \geqslant 4$ | $s \leqslant \min(\sqrt{2N})$ | $d \leqslant N$ |

Table 2: Restrictions on the parameters of the internally re-keyed modes. Here $N$ is the section size, $s = \lceil k/n \rceil$, $d = \lceil k/n \rceil + 1$. $A \succeq B$ denotes that the bound for mode $A$ is better than the bound for mode $B$.

## 5 Conclusion

Results obtained in this paper show that, under security of the used block cipher, the cryptographic properties of the standardized CTR-ACPKM and OMAC-ACPKM-Master modes are improved compared to the properties of the associated CTR and OMAC modes without internal re-keying for all practical cases.

One of the most interesting open problems is the analysis of the re-keying influence on a multi-key, or multi-user, security [8, 20] of the proposed modes. This notion challenges cryptographic algorithms to maintain high levels of security when used with many different keys, by many different users.

## References

[1] «Information processing systems. Cryptographic protection. Technical specification for the use of GOST 28147-89 for attachments encryptions in the IPSEC ESP protocol» (in Russian). Technical specification, Federal Agency for Technical Regulation and Metrology (ROSSTANDART), 2013.

[2] «Information technology. Cryptographic protection of information. Using algorithms GOST 28147-89, GOST R 34.11 and GOST R 34.10 in cryptographic messages of the CMS format» (in Russian). Recommendations for standardization, Federal Agency for Technical Regulation and Metrology (ROSSTANDART), 2014.

[3] «Information technology. Cryptographic protection of information. Using ciphersuites based on GOST 28147-89 for Transport Layer Security (TLS)» (in Russian). Recommendations for standardization, Federal Agency for Technical Regulation and Metrology (ROSSTANDART), 2014.

[4] «Information technology. Cryptographic protection of information. The use of Russian cryptographic algorithms in the Transport Lyer Security protocol (TLS 1.2)» (in Russian). The project of recommendations on standardization, Technical Committee For Standardization «Cryptography and Security Mechanisms», 2017.

[5] Abdalla, Michel and Bellare, Mihir. Increasing the Lifetime of a Key: A Comparative Analysis of the Security of Re-keying Techniques. In Okamoto, Tatsuaki, editor, *Advances in Cryptology — ASIACRYPT 2000*, pages 546–559, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[6] L. Ahmetzyanova, E. Alekseev, I. Oshkin, S. Smyshlyaev, and L. Sonina. On the properties of the CTR encryption mode of the Magma and Kuznyechik block ciphers with re-keying method based on CryptoPro Key Meshing. *Mat. Vopr. Kriptogr.*, 8:39–50, 2017.

[7] L. R. Ahmetzyanova, E. K. Alekseev, I. B. Oshkin, and S. V. Smyshlyaev. Increasing the Lifetime of Symmetric Keys for the GCM Mode by Internal Re-keying. Cryptology ePrint Archive, Report 2017/697, 2017. `https://eprint.iacr.org/2017/697`.

[8] M. Bellare, D. J. Bernstein, and S. Tessaro. Hash-function based prfs: Amac and its multi-user security. Cryptology ePrint Archive, Report 2016/142, 2016. `https://eprint.iacr.org/2016/142`.

[9] M. Bellare and P. Rogaway. Introduction to modern cryptography, chapter 2: Block Ciphers, 2004. URL: `http://www-cse.ucsd.edu/users/mihir/cse207`.

[10] M. Bellare and P. Rogaway. Introduction to modern cryptography, chapter 4: Symmetric Encryption, 2004. URL: http://www-cse.ucsd.edu/users/mihir/cse207.

[11] M. Bellare and P. Rogaway. Introduction to modern cryptography, chapter 7: Message Authentication, 2004. URL: http://www-cse.ucsd.edu/users/mihir/cse207.

[12] Bellare, Mihir. *Practice-Oriented Provable-Security*, pages 1–15. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[13] E. Biham and A. Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In *Proceedings of the 10th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '90, pages 2–21, London, UK, UK, 1991. Springer-Verlag.

[14] D. Chang and M. Nandi. A Short Proof of the PRP/PRF Switching Lemma. Cryptology ePrint Archive, Report 2008/078, 2008. https://eprint.iacr.org/2008/078.

[15] Damgård, Ivan. A "proof-reading" of Some Issues in Cryptography. In Arge, Lars and Cachin, Christian and Jurdziński, Tomasz and Tarlecki, Andrzej, editor, *Automata, Languages and Programming*, pages 2–11, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[16] S. Dziembowski, S. Faust, G. Herold, A. Journault, D. Masny, and F.-X. Standaert. Towards Sound Fresh Re-Keying with Hard (Physical) Learning Problems. Cryptology ePrint Archive, Report 2016/573, 2016. https://eprint.iacr.org/2016/573.

[17] O. Goldreich. On Post-Modern Cryptography. Cryptology ePrint Archive, Report 2006/461, 2006. https://eprint.iacr.org/2006/461.

[18] Iwata, Tetsu and Kurosawa, Kaoru. Stronger Security Bounds for OMAC, TMAC, and XCBC. In Johansson, Thomas and Maitra, Subhamoy, editor, *Progress in Cryptology - INDOCRYPT 2003*, pages 402–415, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[19] Koblitz, Neal and Menezes, Alfred J. Another Look at "Provable Security". *Journal of Cryptology*, 20(1):3–37, Jan 2007.

[20] A. Luykx, B. Mennink, and K. G. Paterson. Analyzing multi-key security degradation. Cryptology ePrint Archive, Report 2017/435, 2017. https://eprint.iacr.org/2017/435.

[21] M. Matsui. Linear Cryptanalysis Method for DES Cipher. In *Workshop on the Theory and Application of Cryptographic Techniques on Advances in Cryptology*, EUROCRYPT '93, pages 386–397, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[22] P. Q. Nguyen. Cryptanalysis vs. Provable Security. In *Proceedings of the 7th International Conference on Information Security and Cryptology*, Inscrypt'11, pages 22–23, Berlin, Heidelberg, 2012. Springer-Verlag.

[23] C. Ramsay and J. Lohuis. TEMPEST attacks against AES. Covertly stealing keys for 200 euro, 2017.

[24] Rogaway, Phillip. Nonce-Based Symmetric Encryption. In Roy, Bimal and Meier, Willi, editor, *Fast Software Encryption*, pages 348–358, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[25] S. V. Smyshlyaev. Re-keying Mechanisms for Symmetric Keys. Internet-Draft draft-irtf-cfrg-re-keying-10, Internet Engineering Task Force, Jan. 2018. Work in Progress.

## Appendix

## A Additional notations

Introduce additional notions which are used in further proofs. For an algorithm $A$ by $A \Rightarrow s$ we denote an event, that occurs if the algorithm $A$ returns value $s$ as its execution result. Denote by $A^{\mathcal{O}_1,\mathcal{O}_2,\cdots}$ an adversary $A$ that interacts with oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$ by making queries. Notation $b \xleftarrow{\$} A^{\mathcal{O}_1,\mathcal{O}_2,\cdots}$ means that the algorithm $A$, after interacting with oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$, outputs bit $b \in \{0,1\}$. For the deterministic algorithm $A$ by $A \to x$ or by $x \leftarrow A$ is denoted that $A$ returns the value $x$. For a set $S \subseteq \{0,1\}^*$ by $S_n$ we denote the set of all strings $s \in S$ such as $|s|$ is multiple of $n$ and $|s| > 0$. Let $\{0,1\}^{\leqslant n}$ be the set of all bit strings with legth less or equal to $n$.

For any set $S$ and distribution $\mathfrak{D}$ on $S$ by $s \xleftarrow{\mathfrak{D}} S$ we denote that value $s$ is chosen in set $S$ by random according to the distribution $\mathfrak{D}$.

## B Security Models

For a cipher $E$ with parameters $n$ and $k$ define

$$\mathsf{Adv}_E^{\text{PRP-CPA}}(A) = \Pr\left[\mathbf{Exp}_E^{\text{PRP-CPA}-1}(A) = 1\right] - \Pr\left[\mathbf{Exp}_E^{\text{PRP-CPA}-0}(A) = 1\right],$$

where the experiments $\mathbf{Exp}_E^{\text{PRP-CPA}-1}(A)$ and $\mathbf{Exp}_E^{\text{PRP-CPA}-0}(A)$ are defined in the following way:

| $\underline{\mathbf{Exp}_E^{\text{PRP-CPA}-b}(A)}$ | $\underline{\mathsf{OracleP}^b(M),\ M \in \{0,1\}^n}$ |
|---|---|
| **if** $b = 0$ **then** | **if** $b = 0$ **then** |
| $\quad P \xleftarrow{\mathcal{U}} Perm(\{0,1\}^n)$ | $\quad$ **return** $P(M)$ |
| **else** | **else** |
| $\quad K \xleftarrow{\mathcal{U}} \{0,1\}^k$ | $\quad$ **return** $E_K(M)$ |
| **end if** | **end if** |
| $b' \xleftarrow{\$} A^{\mathsf{P}^b}$ | |
| **return** $b'$ | |

The PRF notion is defined in the same way as PRP-CPA except for the random permutation $P \xleftarrow{\mathcal{U}} Perm(\{0,1\}^n)$, which is replaced by the random function $F \xleftarrow{\mathcal{U}} Func(\{0,1\}^n)$:

$$\mathsf{Adv}_E^{\text{PRF}}(A) = \Pr\left[\mathbf{Exp}_E^{\text{PRF}-1}(A) = 1\right] - \Pr\left[\mathbf{Exp}_E^{\text{PRF}-0}(A) = 1\right].$$

**Definition 1.** A symmetric encryption scheme $\mathsf{SE}$ *for a set of keys* $\boldsymbol{K}$*, a set of plaintexts* $\mathcal{P}$*, a set of ciphertexts* $\boldsymbol{C}$*, and a set of nonces* $\boldsymbol{N}$ *consists of three algorithms* $\{\mathsf{SE}.\mathcal{K},\ \mathsf{SE}.\mathcal{E},\ \mathsf{SE}.\mathcal{D}\}$*, as follows:*

- $\mathsf{SE}.\mathcal{K}() \xrightarrow{\$} K$*: The randomized key generation algorithm that returns a key* $K \in \boldsymbol{K}$*. This algorithm is often defined by taking security parameter as input. But in this work it will be omitted.*
- $\mathsf{SE}.\mathcal{E}(K, P, IV) \to C$*: The deterministic encryption algorithm, takes a key* $K \in \boldsymbol{K}$*, a plaintext* $P \in \mathcal{P}$*, and a nonce* $IV \in \boldsymbol{N}$ *to return a ciphertext* $C \in \boldsymbol{C}$*.*
- $\mathsf{SE}.\mathcal{D}(K, C, IV) \to P$*: The deterministic decryption algorithm, takes a key* $K \in \boldsymbol{K}$*, a ciphertext* $C \in \boldsymbol{C}$*, and a nonce* $IV \in \boldsymbol{N}$ *to return a plaintext* $P \in \mathcal{P}$*.*

**Definition 2.** *Let* $\mathsf{SE} = \{\mathsf{SE}.\mathcal{K},\ \mathsf{SE}.\mathcal{E},\ \mathsf{SE}.\mathcal{D}\}$ *be a symmetric encryption scheme and let* $A$ *be an adversary. The advantage of* $A$ *for the scheme* $\mathsf{SE}$

*in the IND-CPNA model (IND-CPNA*-advantage*) is defined as*

$$\mathsf{Adv}^{\mathit{IND\text{-}CPNA}}_{\mathsf{SE}}(A) = \\ = \Pr\left[\mathbf{Exp}^{\mathit{IND\text{-}CPNA}-1}_{\mathsf{SE}}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{\mathit{IND\text{-}CPNA}-0}_{\mathsf{SE}}(A) \Rightarrow 1\right],$$

*where the experiment* $\mathbf{Exp}^{\mathit{IND\text{-}CPNA}-b}_{\mathsf{SE}}(A),\ b \in \{0,1\}$ *is defined as follows*

| $\underline{\mathbf{Exp}^{\mathit{IND\text{-}CPNA}-b}_{\mathsf{SE}}(A)}$ | $\underline{\mathsf{Oracle\ Encrypt}^b(P, IV)}$ |
|---|---|
| $K \xleftarrow{\$} \mathsf{SE}.\mathcal{K}()$ | $C \xleftarrow{\$} \mathsf{SE}.\mathcal{E}(K, P, IV)$ |
| $b' \xleftarrow{\$} A^{\mathsf{Encrypt}^b}$ | ***if*** $b = 0$ ***then*** |
| ***return*** $b'$ | $R \xleftarrow{\mathcal{U}} \{0,1\}^{\lvert C\rvert}$ |
| | ***return*** $R$ |
| | ***end if*** |
| | ***return*** $C$ |

**Definition 3.** A deterministic message-authentication scheme $\mathsf{MA}$ *for a set of keys* $\boldsymbol{K}$*, a set of messages* $\boldsymbol{M}$*, and a set of tags* $\boldsymbol{T}$ *consists of two algorithms* $\{\mathsf{MA}.\mathcal{K},\ \mathsf{MA}.\mathcal{T}\}$*, as follows*

- $\mathsf{MA}.\mathcal{K}() \xrightarrow{\$} K$*: The randomized key generation algorithm that returns a key* $K \in \boldsymbol{K}$*.*
- $\mathsf{MA}.\mathcal{T}(K, M) \xrightarrow{\$} T$*: The randomized MAC algorithm, takes a key* $K \in \boldsymbol{K}$ *and message* $M \in \boldsymbol{M}$ *to return a tag* $T \in \boldsymbol{T}$*.*

**Definition 4.** *Let* $\mathsf{MA} = \{\mathsf{MA}.\mathcal{K},\ \mathsf{MA}.\mathcal{T}\}$ *be a message-authentication scheme and let* $A$ *be an adversary. The advantage of* $A$ *for the scheme* $\mathsf{MA}$ *in the PRF model (PRF-advantage) is defined as*

$$\mathsf{Adv}^{\mathit{PRF}}_{\mathsf{MA}}(A) = \Pr\left[\mathbf{Exp}^{\mathit{PRF}-1}_{\mathsf{MA}}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{\mathit{PRF}-0}_{\mathsf{MA}}(A) \Rightarrow 1\right],$$

*where the experiment* $\mathbf{Exp}^{\mathit{PRF}-b}_{\mathsf{MA}}(A),\ b \in \{0,1\}$ *is defined as follows*

$\underline{\mathbf{Exp}^{PRF-1}_{\mathsf{MA}}(A)}$
$K \xleftarrow{\$} \mathsf{MA}.\mathcal{K}()$
$b' \xleftarrow{\$} A^{\mathsf{F}^1}$
***return*** $b'$
$\underline{\text{Oracle } \mathsf{F}^1(M)}$
***return*** $\mathsf{MA}.\mathcal{T}(K, M)$

$\underline{\mathbf{Exp}^{PRF-0}_{\mathsf{MA}}(A)}$
$Rnd \leftarrow \emptyset$
$b' \xleftarrow{\$} A^{\mathsf{F}^0}$
***return*** $b'$

$\underline{\text{Oracle } \mathsf{F}^0(M)}$
***if*** $\nexists\ T' \in \boldsymbol{T} : (M, T') \in Rnd$
***then***
  $T \xleftarrow{\mathcal{U}} \boldsymbol{T}$
  $Rnd \leftarrow Rnd \cup \{(M, T)\}$
***else***
  $T \leftarrow T'$
***end if***
***return*** $T$

## C Internally Re-keyed Modes

$\underline{\text{CTR-ACPKM}_N(K, IV, X)}$
1: $CTR \leftarrow IV \| \text{str}_{n/2}(0)$
2: $b \leftarrow |X|_n$
3: $K^1 \leftarrow K$
4: for $j \leftarrow 2$ to $\lceil b/N \rceil$ do
5: $\quad K^j \leftarrow \text{ACPKM}(K^{j-1})$
6: for $i \leftarrow 1$ to $b$ do
7: $\quad j \leftarrow \lceil i/N \rceil$
8: $\quad CTR_i \leftarrow \pi(CTR, i-1)$
9: $\quad G_i \leftarrow E_{K^j}(CTR_i)$
10: $Y \leftarrow X \oplus \text{msb}_{|X|}(G_1 \| \ldots \| G_b)$
11: **return** $Y$

$\underline{\text{CTR-ACPKM}_N.\mathcal{K}()}$
1: Key generation:
2: $\quad K \xleftarrow{\mathcal{U}} \{0,1\}^k$
3: **return** $K$

$\underline{\text{CTR-ACPKM}_N.\mathcal{E}(K, IV, M)}$
1: Ciphertext computation:
2: $\quad C \leftarrow \text{CTR-ACPKM}_N(K, IV, M)$
3: **return** $C$

$\underline{\text{CTR-ACPKM}_N.\mathcal{D}(K, IV, C)}$
1: Plaintext computation:
2: $\quad M \leftarrow \text{CTR-ACPKM}_N(K, IV, C)$
3: **return** $M$

Pseudocode 4: The CTR-ACPKM Mode.

Gen_Subkey$(K, r)$

1: if $r = n$ then
2: **return** $K$
3: if $r < n$ then
4: **return** $K \cdot \alpha$

OMAC-ACPKM-Master$_{N,T^*}.\mathcal{K}()$

1: Key generation:
2: $K \xleftarrow{\mathcal{U}} \{0,1\}^k$
3: **return** $K$

OMAC-ACPKM-Master$_{N,T^*}.\mathcal{T}(K, M)$

1: Tag computation:
2: $b = |M|_n$
3: $C_0 \leftarrow 0^n$
4: $l \leftarrow \lceil b/N \rceil$
5: $K^1 \| K_1^1 \| ... \| K^l \| K_1^l \leftarrow$
6: ACPKM-Master$_{T^*}(K, k/n + 1, l)$
7: for $i \leftarrow 1$ to $b - 1$ do
8: $j \leftarrow \lceil i/N \rceil$,
9: $C[i] \leftarrow E_{K^j}(M[i] \oplus C[i-1])$,
10: $SK \leftarrow$ Gen_Subkey$(K_1^l, |M[b]|)$
11: if $|M[b]| = n$ then
12: $M'[b] \leftarrow M[b]$
13: else
14: $M'[b] \leftarrow M[b] \| 1 \| 0^{n-1-|M[b]|}$
15: $T \leftarrow E_{K^l}(M'[b] \oplus C[b-1] \oplus SK)$
16: **return** $T$

Pseudocode 5: The OMAC-ACPKM-Master Mode.

# D Security Analysis

## D.1 Security Analysis of the CTR-ACPKM mode

*Proof.* Define hybrid experiments $Hybrid_j(A),\ j = 0, 1, \ldots, \lceil m/N \rceil$. In the experiment $Hybrid_j(A)$ the oracle in the IND-CPNA notion is replaced by the oracle, which operates in the following way:

- The oracle chooses key $K^{j+1} \xleftarrow{\mathcal{U}} \{0,1\}^k$;
- In response to a query $(P, IV)$ the oracle returns $C$, where

$$C = M \oplus \mathrm{msb}_{|P|}(C' \| C^{j+1} \| \ldots \| C^{\lceil m/N \rceil}),$$

here $C' \xleftarrow{\mathcal{U}} \{0,1\}^{nNj}$ and $C^i,\ i = (j+1), \ldots, \lceil m/N \rceil$, is the result of the $i$-th section processing under the $K^i$ section key. Note that the $(j+1)$-th section is processed under the «truly» random $K^{j+1}$ key and each next key is produced according to ACPKM.

The result of any experiment described above is what the adversary $A$ returns as a result.

Note that the $Hybrid_0(A)$ experiment totally coincides with the $\mathbf{Exp}^{\text{IND-CPNA}-1}_{\text{CTR-ACPKM}_N}(A)$ experiment, and the experiment $Hybrid_{\lceil m/N \rceil}(A)$ coincides with $\mathbf{Exp}^{\text{IND-CPNA}-0}_{\text{CTR-ACPKM}_N}(A)$ experiment, i.e. the following inequalities hold:

$$\Pr\left[Hybrid_0(A) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\text{IND-CPNA}-1}_{\text{CTR-ACPKM}_N}(A) \Rightarrow 1\right],$$

$$\Pr\left[Hybrid_{\lceil m/N \rceil}(A) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\text{IND-CPNA}-0}_{\text{CTR-ACPKM}_N}(A) \Rightarrow 1\right].$$

Construct a set of adversaries $B_j$, $j = 1, \ldots, \lceil m/N \rceil$, for the block cipher in the PRF model, which uses $A$ as a black box.

After receiving a query $(P, IV)$ from $A$ the adversary $B_j$ processes this query as in the $Hybrid_j(A)$ experiment but the encrypted blocks for masking the $j$-th section and blocks of the $(j+1)$-th section key are obtained by making queries to the oracle provided by the PRF experiment. Note that $B_j$, $j = 1, \ldots, \lceil m/N \rceil - 1$, makes at most $\sigma_j + s$ queries and $B_{\lceil m/N \rceil}$ makes at most $\sigma_{\lceil m/N \rceil}$ queries. The adversary $B_j$ returns 1, if the adversary $A$ returns 1, and returns 0, otherwise.

Note that

$$\Pr\left[\mathbf{Exp}_E^{\mathrm{PRF}-1}(B_j) \Rightarrow 1\right] = \Pr\left[Hybrid_{j-1}(A) \Rightarrow 1\right],$$

$$\Pr\left[\mathbf{Exp}_E^{\mathrm{PRF}-0}(B_j) \Rightarrow 1\right] = \Pr\left[Hybrid_j(A) \Rightarrow 1\right].$$

The last equality is proceeded from that the input blocks for producing the $K^{j+1}$ section key and the input blocks for masking the $j$-th section are different for the random function. Therefore, the $K^{j+1}$ variable distribution is statistically indistinguishable from the «truly» random one.

Then for the advantages of the adversaries $B_j$

$$\sum_{j=1}^{\lceil m/N \rceil} \mathsf{Adv}_E^{\mathrm{PRF}}(B_j) =$$

$$= \sum_{j=1}^{\lceil m/N \rceil} \Pr\left[Hybrid_{j-1}(A) \Rightarrow 1\right] - \sum_{j=1}^{\lceil m/N \rceil} \Pr\left[Hybrid_j(A) \Rightarrow 1\right] =$$

$$= \Pr\left[Hybrid_0(A) \Rightarrow 1\right] - \Pr\left[Hybrid_{\lceil m/N \rceil}(A) \Rightarrow 1\right] =$$

$$= \mathsf{Adv}_{\mathrm{CTR\text{-}ACPKM}_N}^{\mathrm{IND\text{-}CPNA}}(A).$$

From the PRP/PRF switching lemma [14] for any block cipher $E$ and any adversary $B'$ making at most $q$ queries we have

$$\mathsf{Adv}_E^{\mathrm{PRF}}(B') \leqslant \mathsf{Adv}_E^{\mathrm{PRP\text{-}CPA}}(B') + \frac{q(q-1)}{2^{n+1}} \leqslant \mathsf{Adv}_E^{\mathrm{PRP\text{-}CPA}}(B') + \frac{q^2}{2^{n+1}}.$$

Thus,

$$\mathsf{Adv}^{\text{IND-CPNA}}_{\text{CTR-ACPKM}_N}(A) =$$

$$= \sum_{j=1}^{\lceil m/N \rceil} \mathsf{Adv}^{\text{PRF}}_E(B_j) \leqslant \sum_{j=1}^{\lceil m/N \rceil - 1} \left( \mathsf{Adv}^{\text{PRP-CPA}}_E(B_j) + \frac{(\sigma_i + s)^2}{2^{n+1}} \right) +$$

$$+ \mathsf{Adv}^{\text{PRP-CPA}}_E(B_{\lceil m/N \rceil}) + \frac{(\sigma_{\lceil m/N \rceil})^2}{2^{n+1}} \leqslant$$

$$\leqslant \left\lceil \frac{m}{N} \right\rceil \mathsf{Adv}^{\text{PRP-CPA}}_E(B) + \frac{(\sigma_1 + s)^2 + \ldots + (\sigma_{\lceil m/N \rceil - 1} + s)^2 + (\sigma_{\lceil m/N \rceil})^2}{2^{n+1}},$$

where $B$ is an adversary which makes at most $\sigma_1 + s$ queries. The last relation is due to $\sigma_1 \geqslant \ldots \geqslant \sigma_{\lceil m/N \rceil}$ and $\mathsf{Adv}^{\text{PRP-CPA}}_E(B') \leqslant \mathsf{Adv}^{\text{PRP-CPA}}_E(B'')$ for such adversaries $B'$ and $B''$ with the same computational resources that the queries number made by $B'$ is less than the queries number made by $B''$. □

## D.2 Security Analysis of the OMAC-ACPKM-Master mode

**Definition 5.** A pseudorandom generator $\mathcal{G}$ *for a set of states* $\boldsymbol{K}$ *consists of two algorithms* $(\mathcal{G}.\mathcal{K}, \mathcal{G}.\mathcal{N})$. *The randomized algorithm* $\mathcal{G}.\mathcal{K}$ *generates the generator state* $K^* \in \boldsymbol{K}$. *The deterministic algorithm* $\mathcal{G}.\mathcal{N}$ *takes the state* $K^* \in \boldsymbol{K}$ *and the number* $h \in \mathbb{N}$ *returns the string of block length* $h$.

**Definition 6.** *Let* $\mathcal{G} = (\mathcal{G}.\mathcal{K}, \mathcal{G}.\mathcal{N})$ *be the pseudorandom generator and let* $A$ *be the adversary. The advantage of* $A$ *for the generator* $\mathcal{G}$ *in the PRG model (PRG-advantage) is defined as*

$$\mathsf{Adv}^{PRG}_{\mathcal{G}}(A) = \Pr\left[\mathbf{Exp}^{PRG-1}_{\mathcal{G}}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{PRG-0}_{\mathcal{G}}(A) \Rightarrow 1\right],$$

*where experiments* $\mathbf{Exp}^{PRG-1}_{\mathcal{G}}(A)$ *and* $\mathbf{Exp}^{PRG-0}_{\mathcal{G}}(A)$ *are defined as follows*

| $\underline{\mathbf{Exp}^{PRG-1}_{\mathcal{G}}(A)}$ | $\underline{\mathbf{Exp}^{PRG-0}_{\mathcal{G}}(A)}$ |
|---|---|
| $h \xleftarrow{\$} A,\ h \in \mathbb{N}$ | $h \xleftarrow{\$} A,\ h \in \mathbb{N}$ |
| $K^* \xleftarrow{\$} \mathcal{G}.\mathcal{K}()$ | $s \xleftarrow{\$} \{0,1\}^{nh}$ |
| $s \leftarrow \mathcal{G}.\mathcal{N}(K^*, h)$ | $b' \xleftarrow{\$} A(s)$ |
| $b' \xleftarrow{\$} A(s)$ | ***return*** $b'$ |
| ***return*** $b'$ | |

The key generation algorithm ACPKM-Master$_{T^*}$ with the master key change frequency $T^*$ can be considered as the pseudorandom generator. Here algorithms ACPKM-Master$_{T^*}.\mathcal{K}$ and ACPKM-Master$_{T^*}.\mathcal{N}$ are defined as

– $\text{ACPKM-Master}_{T^*}.\mathcal{K}() \xrightarrow{\$} K^* :\ K^* \xleftarrow{\mathcal{U}} \{0,1\}^k$;
– $\text{ACPKM-Master}_{T^*}.\mathcal{N}(K^*, h) \to s :\ s = \text{CTR-ACPKM}_{T^*}(K^*, 1^{n/2}, 0^{nh})$.

**Theorem 3.** *Let $A$ be an adversary in the PRG model for the generator ACPKM-Master$_{T^*}$, that makes query up to $h$ blocks. Then there exists adversary $B$ in the IND-CPNA model for the CTR-ACPKM$_{T^*}$ encryption mode that makes at most one query length up to $h$ blocks. For all $T^* \in \mathbb{N} : n \mid T^*$ the following inequation holds*

$$\mathsf{Adv}^{\textit{IND-CPNA}}_{\textit{CTR-ACPKM}_{T^*}}(B) \geqslant \mathsf{Adv}^{\textit{PRG}}_{\textit{ACPKM-Master}_{T^*}}(A).$$

*Proof.* Construct the adversary $B$ in the IND-CPNA model for the CTR-ACPKM$_{T^*}$ encryption mode which uses $A$ as a black box. Adversary $B$ has access to oracle $\mathsf{Encrypt}^b$ and simulates the experiment $\mathbf{Exp}^{\text{PRG}-b}_{\text{ACPKM-Master}_{T^*}}$ for the adversary $A$ as follows

$\underline{B^{\mathsf{Encrypt}^b}(A)}$
$h \xleftarrow{\$} A$
$P \leftarrow 0^{nh},\ IV \leftarrow 1^{n/2}$
$s \leftarrow \mathsf{Encrypt}^b(IV, P)$
$b' \xleftarrow{\$} A(s)$
**return** $b'$

Note that if $b = 0$ then $B$ simulates the experiment $\mathbf{Exp}^{\text{PRG}-0}_{\text{ACPKM-Master}_{T^*}}(A)$ and if $b = 1$ then it simulates the experiment $\mathbf{Exp}^{\text{PRG}-1}_{\text{ACPKM-Master}_{T^*}}(A)$.

Hence,

$$\mathsf{Adv}^{\text{PRG}}_{\text{ACPKM-Master}_{T^*}}(A) = \mathsf{Adv}^{\text{IND-CPNA}}_{\text{CTR-ACPKM}_{T^*}}(B).$$

□

**Definition 7.** *For a permutation $P \in Perm(\{0,1\}^n)$ by $CBCMAC_P$ we define the function that takes message $M = M[1]||\ldots||M[r]$, $r \in \mathbb{N}$, $M[1], \ldots, M[r] \in \{0,1\}^n$ and returns value*

$$CBCMAC_P(M) = P(P(\ldots(P(M[1]) \oplus M[2])\ldots) \oplus M[r]).$$

**Definition 8.** *For a permutation $P \in Perm(\{0,1\}^n)$ by CBC-$E_P$ we define the function that takes message $M \in \{0,1\}^*_n$ and returns value*

$$CBC\text{-}E_P(M) = P^{-1}(CBCMAC_P(M)).$$

**Definition 9.** *For a permutation $P \in Perm(\{0,1\}^n)$ and block $K_1 \in \{0,1\}^n$ by $\mathsf{MAC}_{P,K_1}$ we define the function that takes message $M \in \{0,1\}^*_n$*

*and returns value*

$$\mathsf{MAC}_{P,K_1}(M) = P(\mathit{CBC\text{-}E}_P(M) \oplus K_1).$$

**Definition 10.** *For permutations $P_1, P_2 \in \mathit{Perm}\,(\{0,1\}^n)$ by $\mathsf{MAC}_{P_1,P_2}$ we define the function that takes message $M \in \{0,1\}_n^*$ and returns value*

$$\mathsf{MAC}_{P_1,P_2}(M) = P_2(\mathit{CBC\text{-}E}_{P_1}(M)).$$

**Definition 11.** *For the blockcipher*

$$E = \left\{ E_K \in \mathit{Perm}\,(\{0,1\}^n) \mid K \in \{0,1\}^k \right\}$$

*define the family $\mathcal{F}_E$ as follows*

$$\mathcal{F}_E =$$

$$= \left\{ \underbrace{(\mathit{CBCMAC}_{E_K}, \mathsf{MAC}_{E_K,K_1}, \mathsf{MAC}_{E_K,K_1\cdot\alpha})}_{(F_1,F_2,F_3)} \mid K \in \{0,1\}^k,\ K_1 \in \{0,1\}^n \right\},$$

*where $\alpha$ is a primitive element of field $GF(2^n)$, $\cdot$ is multiplication in field $GF(2^n)$.*

*The uniform distribution $\mathcal{U}$ on $\mathcal{F}_E$ is defined as follows*

$$\Pr_{(F_1,F_2,F_3)\xleftarrow{\mathcal{U}}\mathcal{F}_E}[(F_1,F_2,F_3)] =$$
$$= \Pr_{K\xleftarrow{\mathcal{U}}\{0,1\}^k}[K] \cdot \Pr_{K_1\xleftarrow{\mathcal{U}}\{0,1\}^n}[K_1], \quad \forall\ (F_1,F_2,F_3) \in \mathcal{F}_E.$$

**Definition 12.** *For the family of all permutations $\mathit{Perm}\,(\{0,1\}^n)$ define the family $\mathcal{F}_{Perm}$ as follows*

$$\mathcal{F}_{Perm} =$$

$$= \left\{ \underbrace{(\mathit{CBCMAC}_P, \mathsf{MAC}_{P,K_1}, \mathsf{MAC}_{P,K_1\cdot\alpha})}_{(F_1,F_2,F_3)} \mid P \in \mathit{Perm}\,(\{0,1\}^n), K_1 \in \{0,1\}^n \right\},$$

*where $\alpha$ is a prime element of field $GF(2^n)$, $\cdot$ is multiplication in field $GF(2^n)$.*

*The uniform distribution* $\mathcal{U}$ *on* $\mathcal{F}_{Perm}$ *is defined as follows*

$$\Pr_{(F_1,F_2,F_3)\xleftarrow{\mathcal{U}}\mathcal{F}_{Perm}}[(F_1,F_2,F_3)] =$$
$$= \Pr_{P\xleftarrow{\mathcal{U}}Perm(\{0,1\}^n)}[P]\cdot\Pr_{K_1\xleftarrow{\mathcal{U}}\{0,1\}^n}[K_1], \quad \forall\ (F_1,F_2,F_3)\in\mathcal{F}_{Perm}.$$

**Definition 13.** *For the family of all permutations* $Perm(\{0,1\}^n)$ *define the family* $\mathcal{F}_{3Perm}$ *as follows*

$$\mathcal{F}_{3Perm} = \left\{ \underbrace{(CBCMAC_{P_1}, \mathsf{MAC}_{P_1,P_2}, \mathsf{MAC}_{P_1,P_3})}_{(F_1,F_2,F_3)} \mid P_1,P_2,P_3\in Perm(n) \right\}.$$

*The uniform distribution* $\mathcal{U}$ *on* $\mathcal{F}_{3Perm}$ *is defined as follows*

$$\Pr_{(F_1,F_2,F_3)\xleftarrow{\mathcal{U}}\mathcal{F}_{3Perm}}[(F_1,F_2,F_3)] =$$
$$= \Pr_{P_1,P_2,P_3\xleftarrow{\mathcal{U}}Perm(\{0,1\}^n)}[P_1,P_2,P_3], \quad \forall\ (F_1,F_2,F_3)\in\mathcal{F}_{Perm}$$

**Definition 14.** *For the section block size* $N$ *define the message-authentication scheme OMAC-RK*$_N$ *(OMAC with Random Keys). This scheme processes messages* $M$, $|M|\leqslant m$ *as it is described in section 3.2. The keys* $K^1,\ldots,K^l \xleftarrow{\mathcal{U}} \{0,1\}^k$, $K_1^1,\ldots,K_1^l \xleftarrow{\mathcal{U}} \{0,1\}^n$, *where* $l=\lceil m/N\rceil$, *are generated at random independently and uniformly.*

Consider the additional adversary model 3PRF$_N$.

**Definition 15.** *Let* $A$ *be an adversary. The advantage of* $A$ *in the 3PRF*$_N$ *model for the finite family*

$$\mathcal{F} = \left\{(F_1,F_2,F_3),\ F_1\in Func\left(\{0,1\}^{nN},\{0,1\}^n\right),\ F_2,F_3\in Func\left(\{0,1\}_n^{\leqslant nN},\{0,1\}^n\right)\right\},$$

*with the certain distribution* $\mathfrak{D}$ *on it is defined as*

$$\mathsf{Adv}_{\mathcal{F}}^{3PRF_N}(A) = \Pr\left[\mathbf{Exp}_{\mathcal{F}}^{3PRF_N-1}(A)\Rightarrow 1\right] - \Pr\left[\mathbf{Exp}_{\mathcal{F}}^{3PRF_N-0}(A)\Rightarrow 1\right],$$

*where experiments* $\mathbf{Exp}_{\mathcal{F}}^{3PRF_N-1}(A)$ *and* $\mathbf{Exp}_{\mathcal{F}}^{3PRF_N-0}(A)$ *are defined as follows*

$\underline{\mathbf{Exp}_{\mathcal{F}}^{3PRF_N-b}(A)}$
***if*** $b = 0$ ***then***
  $F_1 \xleftarrow{\mathcal{U}} Func\left(\{0,1\}^{nN}, \{0,1\}^n\right)$
  $F_2, F_3 \xleftarrow{\mathcal{U}} Func\left(\{0,1\}_n^{\leqslant nN}, \{0,1\}^n\right)$
***else***
  $(F_1, F_2, F_3) \xleftarrow{\mathfrak{D}} \mathcal{F}$
***end if***
$b' \xleftarrow{\$} A^{\mathsf{F}_1^b,\ \mathsf{F}_2^b,\ \mathsf{F}_3^b}$
***return*** $b'$

$\underline{\text{Oracle } \mathsf{F}_1^b,\ M \in \{0,1\}^{nN}}$
***return*** $F_1(M)$

$\underline{\text{Oracle } \mathsf{F}_2^b,\ M \in \{0,1\}_n^{\leqslant nN}}$
***return*** $F_2(M)$

$\underline{\text{Oracle } \mathsf{F}_3^b,\ M \in \{0,1\}_n^{\leqslant nN}}$
***return*** $F_3(M)$

**Lemma 1.** *Let $A$ be an adversary in the $3PRF_N$ model for the $\mathcal{F}_E$ family and the total length of his queries is at most $\sigma$ blocks. Then exists adversary $B$ in the PRP-CPA model for the $E$ blockcipher and exists adversary $C$ in model $3PRF_N$ for the $\mathcal{F}_{Perm}$ family such as*

$$\mathsf{Adv}_E^{PRP\text{-}CPA}(B) \geqslant \mathsf{Adv}_{\mathcal{F}_E}^{3PRF_N}(A) - \mathsf{Adv}_{\mathcal{F}_{Perm}}^{3PRF_N}(C),$$

*where $B$ makes at most $\sigma$ queries to his oracle, $C$ queries are of the total block length at most $\sigma$.*

*Proof.* Construct the adversary $C$ in the $3\mathrm{PRF}_N$ model for the $\mathcal{F}_{Perm}$ family which uses $A$ as a black box.

$C$ has access to the oracles $\mathsf{F}_1^b,\ \mathsf{F}_2^b,\ \mathsf{F}_3^b$ and simulates the experiment $\mathbf{Exp}_{\mathcal{F}_E}^{3\mathrm{PRF}_N-b}$ for the adversary $A$ by simply translating all queries of adversary $A$ to his oracles. At the end of simulation it returns the same bit as $A$

Note that if $b = 0$ then $C$ simulates exactly the $\mathbf{Exp}_{\mathcal{F}_E}^{3\mathrm{PRF}_N-0}$ experiment. Hence,

$$\Pr\left[\mathbf{Exp}_{\mathcal{F}_{Perm}}^{3\mathrm{PRF}_N-0}(C) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}_{\mathcal{F}_E}^{3\mathrm{PRF}_N-0}(A) \Rightarrow 1\right],$$

$$\Pr\left[\mathbf{Exp}_{\mathcal{F}_{Perm}}^{3\mathrm{PRF}_N-1}(C) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}_{\mathcal{F}_{Perm}}^{3\mathrm{PRF}_N-1}(A) \Rightarrow 1\right]$$

Now construct the adversary $B$ in the PRP-CPA model for the $E$ blockcipher which uses the same adversary $A$ as a black box.

$B$ has access to the oracle $\mathsf{P}^b$ and simulates the $\mathbf{Exp}_{\mathcal{F}_E}^{3\mathrm{PRF}_N-b}$ experiment for the adversary $A$ as follows.

$\underline{B^{\mathsf{P}^b}(A)}$
$K_1 \xleftarrow{\mathcal{U}} \{0,1\}^n$
$b' \xleftarrow{\$} A^{\mathsf{SimF}_1^b,\ \mathsf{SimF}_2^b,\ \mathsf{SimF}_3^b}$
**return** $b'$

$\underline{\mathsf{Subfunc}\ \mathsf{SimF}_1^b}$
**return** $CBCMAC_{\mathsf{P}^b}(M)$
$\underline{\mathsf{Subfunc}\ \mathsf{SimF}_2^b}$
**return** $MAC_{\mathsf{P}^b,K_1}(M)$
$\underline{\mathsf{Subfunc}\ \mathsf{SimF}_3^b}$
**return** $MAC_{\mathsf{P}^b,K_1\cdot\alpha}(M)$

Note that if $b = 1$ then $B$ simulates exactly the $\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_E}$ experiment for $A$ and if $b = 0$ the adversary $B$ simulates $\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{Perm}}$. Thus,

$$\Pr\left[\mathbf{Exp}^{\text{PRP-CPA}-1}_E(B) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_E}(A) \Rightarrow 1\right]$$

$$\Pr\left[\mathbf{Exp}^{\text{PRP-CPA}-0}_E(B) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{Perm}}(A) \Rightarrow 1\right]$$

By definition,

$$\begin{aligned}
&\mathsf{Adv}^{\text{PRP-CPA}}_E(B) = \\
&\quad = \Pr\left[\mathbf{Exp}^{\text{PRP-CPA}-1}_E(B) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{\text{PRP-CPA}-0}_E(B) \Rightarrow 1\right] = \\
&\quad = \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_E}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{Perm}}(A) \Rightarrow 1\right] = \\
&\quad = \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_E}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_E}(A) \Rightarrow 1\right] - \\
&\quad - \left( \underbrace{\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{Perm}}(A) \Rightarrow 1\right]}_{=\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{Perm}}(C) \Rightarrow 1\right]} - \underbrace{\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_E}(A) \Rightarrow 1\right]}_{=\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_{Perm}}(C) \Rightarrow 1\right]} \right) = \\
&\qquad\qquad = \mathsf{Adv}^{3\mathrm{PRF}_N}_{\mathcal{F}_E}(A) - \mathsf{Adv}^{3\mathrm{PRF}_N}_{\mathcal{F}_{Perm}}(C).
\end{aligned}$$

□

**Definition 16.** *Let $A$ be an adversary. The advantage of $A$ in the model 3PRP-CPA for the family of all permutations $Perm(\{0,1\}^n)$ is defined as*

$$\begin{aligned}
&\mathsf{Adv}^{\textit{3PRP-CPA}}_{Perm(\{0,1\}^n)}(A) = \\
&= \Pr\left[\mathbf{Exp}^{\textit{3PRP-CPA}-1}_{Perm(\{0,1\}^n)}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{\textit{3PRP-CPA}-0}_{Perm(\{0,1\}^n)}(A) \Rightarrow 1\right],
\end{aligned}$$

*where the experiments $\mathbf{Exp}^{\textit{3PRP-CPA}-1}_{Perm(\{0,1\}^n)}(A)$ and $\mathbf{Exp}^{\textit{3PRP-CPA}-0}_{Perm(\{0,1\}^n)}(A)$ are defined as follows:*

$\underline{\mathbf{Exp}^{3PRP\text{-}CPA-b}_{Perm(\{0,1\}^n)}(A)}$
***if*** $b = 0$ ***then***
  $P_1, P_2, P_3 \xleftarrow{\mathcal{U}} Perm(\{0,1\}^n)$
***else***
  $P \xleftarrow{\mathcal{U}} Perm(\{0,1\}^n),\ K_1 \xleftarrow{\mathcal{U}} \{0,1\}^n$
***end if***
$b' \xleftarrow{\$} A^{\mathsf{P}_1^b, \mathsf{P}_2^b, \mathsf{P}_3^b}$
***return*** $b'$

$\underline{\text{Oracle } \mathsf{P}_1{}^b(M), M \in \{0,1\}^n}$
***if*** $b = 0$ ***then***
  ***return*** $P_1(M)$
***else***
  ***return*** $P(M)$
***end if***
$\underline{\text{Oracle } \mathsf{P}_2{}^b(M), M \in \{0,1\}^n}$
***if*** $b = 0$ ***then***
  ***return*** $P_2(M)$
***else***
  ***return*** $P(M \oplus K_1)$
***end if***
$\underline{\text{Oracle } \mathsf{P}_2{}^b(M), M \in \{0,1\}^n}$
***if*** $b = 0$ ***then***
  ***return*** $P_3(M)$
***else***
  ***return*** $P(M \oplus K_1 \cdot \alpha)$
***end if***

**Lemma 2.** *[18] For any $A$ in the 3PRP-CPA model that makes at most $q$ queries to his oracle*

$$\mathsf{Adv}^{3PRP\text{-}CPA}_{Perm(\{0,1\}^n)}(A) \leqslant \frac{q^2}{2^n}.$$

**Lemma 3.** *Let $A$ be an adversary in the $3PRF_N$ model for the $\mathcal{F}_{Perm}$ family and the total length of his queries is at most $\sigma$ blocks. Then exists adversary $B$ in the 3PRP-CPA model and exists adversary $C$ in model $3PRF_N$ for the $\mathcal{F}_{3Perm}$ family such as*

$$\mathsf{Adv}^{PRP\text{-}CPA}_{E}(B) \geqslant \mathsf{Adv}^{3PRF_N}_{\mathcal{F}_E}(A) - \mathsf{Adv}^{3PRF_N}_{\mathcal{F}_{Perm}}(C),$$

*where $B$ makes at most $\sigma$ queries to his oracle, $C$ queries are of the total block length at most $\sigma$.*

*Proof.* Construct the adversary $C$ in $3\mathrm{PRF}_N$ model for the $\mathcal{F}_{3Perm}$ family which uses $A$ as a black box.

$C$ has access to the oracles $\mathsf{F}_1^b,\ \mathsf{F}_2^b,\ \mathsf{F}_3^b$ and simulates the experiment $\mathbf{Exp}^{3\mathrm{PRF}_N-b}_{\mathcal{F}_{Perm}}$ for the adversary $A$ by simply translating all queries of adversary $A$ to his oracles. At the end of simulation it returns the same bit as $A$

Note that if $b = 0$ then $C$ simulates exactly the $\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_{Perm}}$ experiment. Hence,

$$\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_{3Perm}}(C) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_{Perm}}(A) \Rightarrow 1\right],$$

$$\Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{3Perm}}(C) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right].$$

Now construct the adversary $B$ in the 3PRP-CPA model which uses the same adversary $A$ as a black box.

$B$ has access to the oracle $\mathsf{P}^b$ and simulates the $\mathbf{Exp}^{\mathrm{3PRF}_N-b}_{\mathcal{F}_{Perm}}$ experiment for the adversary $A$ as follows.

$\underline{B^{\mathsf{P}^b}(A)}$
$b' \xleftarrow{\$} A^{\mathsf{SimF}^b_1,\ \mathsf{SimF}^b_2,\ \mathsf{SimF}^b_3}$
**return** $b'$

$\underline{\mathsf{Subfunc}\ \mathsf{SimF}^b_1}$
**return** $\mathrm{CBCMAC}_{\mathsf{P}^b_1}(M)$

$\underline{\mathsf{Subfunc}\ \mathsf{SimF}^b_2}$
**return** $\mathsf{MAC}_{\mathsf{P}^b_1,\mathsf{P}^b_2}(M)$

$\underline{\mathsf{Subfunc}\ \mathsf{SimF}^b_3}$
**return** $\mathsf{MAC}_{\mathsf{P}^b_1,\mathsf{P}^b_3}(M)$

Note that if $b = 1$ then $B$ simulates exactly the experiment $\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{Perm}}$ for $A$ and if $b = 0$ adversary $B$ simulates $\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{3Perm}}$. Thus,

$$\Pr\left[\mathbf{Exp}^{\text{3PRP-CPA}-1}_{Perm(\{0,1\}^n)}(B) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{Perm}}(A) \Rightarrow 1\right],$$
$$\Pr\left[\mathbf{Exp}^{\text{3PRP-CPA}-0}_{Perm(\{0,1\}^n)}(B) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right].$$

By definition,

$$\begin{aligned}
&\mathsf{Adv}^{\text{3PRP-CPA}}_{Perm(\{0,1\}^n)}(B) = \\
&\quad = \Pr\left[\mathbf{Exp}^{\text{3PRP-CPA}-1}_{Perm(\{0,1\}^n)}(B) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{\text{3PRP-CPA}-0}_{Perm(\{0,1\}^n)}(B) \Rightarrow 1\right] = \\
&\quad = \Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{Perm}}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right] = \\
&\quad = \Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{Perm}}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-0}_{\mathcal{F}_{Perm}}(A) \Rightarrow 1\right] - \\
&\quad - \left( \underbrace{\Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right]}_{=\Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-1}_{\mathcal{F}_{3Perm}}(C)\Rightarrow 1\right]} - \underbrace{\Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-0}_{\mathcal{F}_{Perm}}(A) \Rightarrow 1\right]}_{=\Pr\left[\mathbf{Exp}^{\mathrm{3PRF}_N-0}_{\mathcal{F}_{3Perm}}(C)\Rightarrow 1\right]} \right) = \\
&\qquad\qquad = \mathsf{Adv}^{\mathrm{3PRF}_N}_{\mathcal{F}_{Perm}}(A) - \mathsf{Adv}^{\mathrm{3PRF}_N}_{\mathcal{F}_{3Perm}}(C).
\end{aligned}$$

□

**Lemma 4.** *Let $n, N, q_1, q_2, m_1, \ldots, m_{q_2}$ be positive integers such as $m_j \leqslant N$ and $q_1 N + m_1 + \ldots + m_{q_2} \leqslant 2^{n-1}$. Let also $T_1, \ldots, T_{q_1} \in \{0,1\}^n$ be different and for two sets of strings $M_1^{(1)}, \ldots, M_{q_1}^{(1)}$ and $M_1^{(2)}, \ldots, M_{q_2}^{(2)}$*

– *Let $M_1^{(1)}, \ldots, M_{q_1}^{(1)} \in \{0,1\}^{n \cdot N}$ be different;*

– *Let $M_1^{(2)}, \ldots, M_{q_2}^{(2)}$ be different and $M_j^{(2)} \in \{0,1\}^{n \cdot m_j}$, $j = 1, \ldots, q_2$.*

*Let us consider the event Event according to uniformly random choice of $P$ from the set $Perm(\{0,1\}^n)$ if both following conditions hold*

– *$CBCMAC_P(M_j^{(1)}) = T_j$, $j = 1, \ldots, q_1$;*

– *CBC-$E_P(M_1^{(2)}), \ldots,$ CBC-$E_P(M_{q_2}^{(2)})$ are different.*

*Thus*

$$\Pr[Event] \geqslant \frac{1}{2^{q_1 n}} \left(1 - \frac{2(q_1 N + m_1 + \ldots + m_{q_2})^2}{2^n}\right).$$

*Proof.* We start the proof by describing the randomized algorithm that constructs permutation $P$ from $Perm(\{0,1\}^n)$ such that $Event$ occurs for $P$. Then we lower estimate the cardinality of set of all such permutations. Then the ratio of this estimate to the cardinality of set $Perm(\{0,1\}^n)$ can be used as the estimation for the desired probability $\Pr[Event]$.

Describe the randomized algorithm that constructs $P \in Perm(\{0,1\}^n)$ such that for $P$ the $Event$ conditions hold. This algorithm constructs $P$ by defining the images $P(x)$ of strings $x \in \{0,1\}^n$. We assume that no one image $P(x)$ is defined at the start of the algorithm. By $Domain(P)$ denote the set of strings $x$ such that $P(x)$ is just defined and by $Free(P)$ denote the set of strings from $\{0,1\}^n$ those are not images for any strings $x$. Thus, $Domain(P) = \emptyset$ and $Free(P) = \{0,1\}^n$ at the start of the algorithm. By $\mathcal{T}$ we denote the set $\{T_1, \ldots, T_{q_1}\}$. For the sets $A, B \subseteq \{0,1\}^n$ by $A \oplus B$ we denote the set $\{a \oplus b | a \in A, b \in B\}$. Note that the inequation $|A \oplus B| \leqslant |A| \cdot |B|$ holds for all $A$ and $B$.

Recall that on values calculating by CBCMAC$_P$ or CBC-E$_P$ the input sequence blocks are used consequently. By $X_i^{(k)}[j]$ in algorithm specification we denote the string that $P$ takes as input after xoring the $j$-th message block $M_i^{(k)}$ and the current internal state. In this notation the $Event$ conditions can be rewritten as $P(X_i^{(1)}[N]) = T_i$, $i = 1, \ldots, q_1$, and $X_1^{(2)}[m_1], \ldots, X_{q_2}^{(2)}[m_{q_2}]$ are different. All $X_i^{(k)}[j]$ except the first ones depend on permutation $P$. In the main part of the algorithm the images $P(X_i^{(k)}[j])$ are defined consequently for all $j$ except the last ones (the $N$-th if $k = 1$ and $m_i$-th if $k = 2$). This is made so as the next conditios hold:

1. strings $X_i^{(k)}[j]$ for different tuples $i, k, j$ are equal only if there is no such permutation $P$ that these blocks are different (this is true when all

previous blocks of message were equal);

2. the images $P(X_i^{(k)}[j])$ are chosen not equal to the strings $T_1, \ldots, T_{q_1}$ because that strings are used to define the images $P(X_1^{(1)}[N]), \ldots, P(X_{q_1}^{(1)}[N])$ and those images match the function $\mathrm{CBCMAC}_P$ outputs when taking the first set strings.

It is easy to see that permutation $P$ defined in such way satisfies the *Event* conditions.

The algorithm formal description

1. $X_1^{(1)}[1] \leftarrow M_1^{(1)}[1], \ldots, X_{q_1}^{(1)}[1] \leftarrow M_{q_1}^{(1)}[1]$
2. $X_1^{(2)}[1] \leftarrow M_1^{(2)}[1], \ldots, X_{q_2}^{(2)}[1] \leftarrow M_{q_2}^{(2)}[1]$
3. $Defined \leftarrow \{X_1^{(1)}[1], \ldots, X_{q_1}^{(1)}[1], X_1^{(2)}[1], \ldots, X_{q_2}^{(2)}[1]\}$
4. for $j \leftarrow 1$ to $N-1$ do
5. for $k \leftarrow 1$ to 2 do
6. for $i \leftarrow 1$ to $q_k$ do
7. if $((k = 1)$ or $(k = 2$ and $j \leqslant m_i - 1))$ and $(X_i^{(k)}[j] \notin Domain(P))$ then
8. $Equal \leftarrow \{(k', i') \mid X_i^{(k)}[j] = X_{i'}^{(k')}[j],\ k' = 1, 2,\ i' = 1, \ldots, q_{k'}\}$;
9. $Bad \leftarrow Defined \oplus \{M_{i'}^{(k')}[j+1] \mid (k', i') \in Equal\}$;
10. $P(X_i^{(k)}[j]) \xleftarrow{\mathcal{U}} Free(P) \setminus (Bad \cup \mathcal{T})$;
11. for all $(k', i') \in Equal$ do
12. $X_{i'}^{(k')}[j+1] \leftarrow P(X_i^{(k)}[j]) \oplus M_{i'}^{(k')}[j+1]$;
13. $Add \leftarrow \{X_{i'}^{(k')}[j+1] \mid (k', i') \in Equal\}$;
14. $Defined \leftarrow Defined \cup Add$;
15. for $i \leftarrow 1$ to $q_1$ do $P(X_{(i)}^1[N]) \leftarrow T_i$;
16. Complete the definition of $P$ images by random strings from $Free(P)$.

Note that at steps 8 to 14 we define images for equal strings $X_i^{(k)}[j]$ (it can be only if the prefixes of $M_i^{(k)}$ are equal). Note also that the image $P(X_i^{(k)}[j])$ on the step 10 is defined such that $P(X_i^{(k)}[j])$ when xoring with any possible next block $M_{i'}^{(k')}[j+1]$ was not in the set $Defined$. The set $Bad$ is used for this purpose. The second *Event* condition fulfills because of taking new values from set $Free(P)$. The first condition is fulfilled due to the step 15.

Introduce some additional notation. By $l_0$ denote the $Defined$ set cardinality after fulfilling the step 3 and by $l_t$, $t = 1, 2, \ldots$ denote the cardinalities of sets $Add$ at the step 13 after defining the $t$-th image in $P$. Note that $Add \cap Defined = \emptyset$, thus, after defining the $t$-th image in $P$ cardinality of $Defined$ is equal to $l_0 + l_1 + \ldots + l_t$. Note also that the cardinality of the set $\{M_{i'}^{(k')}[j+1] \mid (k', i') \in Equal\}$ used in the set $Bad$ definition is equal to

the cardinality of the set $Add$, that is obtained on this iteration on step 13 because of its construction ($Add$ is defined as xoring all elements of the set $\{M_{i'}^{(k')}[j+1] \mid (k',i') \in Equal\}$ with the single string).

Now we can estimate the cardinality of the algorithm possible results set. The desired cardinality is a multiple of cardinalities of sets from those the image $P(X_i^{(k)}[j])$ is chosen at step 10 and the cardinality of the set $Free(P)$ at step 16. When defining the $t$-th image the cardinality of set $Free(P) \setminus (Bad \cup \mathcal{T})$ is at most

$$2^n - (t-1) - q_1 - ((l_0 + \ldots + l_{t-1}) \cdot l_t).$$

Actually,

$$|Bad| = |Defined \oplus \{M_{i'}^{(k')}[j+1] \mid (k',i') \in Equal\}| \leqslant (l_0 + \ldots + l_{t-1}) \cdot l_t,$$

and $|Free(P)| = 2^n - (t-1)$, $|\mathcal{T}| = q_1$.

By $s$ denote the number of images in the permutation $P$ that are defined before fulfilling the step 15. Note that $s \leqslant q_1(N-1)+(m_1-1)+\ldots+(m_{q_2}-1)$. Thus for the probability of $Event$ holds the following relation

$$\begin{aligned}
\Pr[Event] &\geqslant \frac{(2^n - s - q_1)! \cdot \prod_{t=1}^{s}(2^n - (t-1) - (q_1 + (l_0 + \ldots + l_{t-1})l_t))}{(2^n)!} = \\
&= \frac{1}{(2^n - s) \cdot \ldots \cdot (2^n - s - q_1 + 1)} \cdot \prod_{t=1}^{s} \frac{2^n - (t-1) - (q_1 + (l_0 + \ldots + l_{t-1})l_t)}{2^n - (t-1)} \geqslant \\
&\geqslant \frac{1}{2^{nq_1}} \cdot \prod_{t=1}^{s} \left(1 - \frac{q_1 + (l_0 + \ldots + l_{t-1})l_t}{2^n - (t-1)}\right) \geqslant \\
&\geqslant \frac{1}{2^{nq_1}} \cdot \left(1 - \sum_{t=1}^{s} \frac{q_1 + (l_0 + \ldots + l_{t-1})l_t}{2^n - (t-1)}\right).
\end{aligned}$$

Note that the $Defined$ set cardinality can not be greater than the number of blocks in the strings of the first and the second groups together. Thus $l_0 + \ldots + l_s \leqslant q_1 N + m_1 + \ldots + m_{q_2}$. By the conditions of lemma we get $s \leqslant q_1(N-1) + (m_1 - 1) + \ldots + (m_{q_2} - 1) \leqslant 2^{n-1}$. The following estimation ends the proof.

$$\sum_{t=1}^{s} \frac{q_1 + (l_0 + \ldots + l_{t-1})l_t}{2^n - (t-1)} \leqslant \frac{1}{2^{n-1}} \sum_{t=1}^{s} (q_1 + (l_0 + \ldots + l_{t-1})l_t) =$$

$$= \frac{1}{2^{n-1}} \sum_{t=1}^{s} q_1 + \frac{1}{2^{n-1}} \sum_{t=1}^{s} (l_0 + \ldots + l_{t-1})l_t = \frac{2q_1 \cdot s}{2^n} + \frac{(l_0 + \ldots + l_s)^2 - l_0^2 - \ldots - l_s^2}{2^n} \leqslant$$

$$\leqslant \frac{2q_1(q_1(N-1) + (m_1 - 1) + \ldots + (m_{q_2} - 1))}{2^n} + \frac{(q_1 N + m_1 + \ldots + m_{q_2})^2}{2^n} \leqslant$$

$$\leqslant \frac{2(q_1 N + m_1 + \ldots + m_{q_2})^2}{2^n}.$$

□

**Lemma 5.** *Let $n, N, q_1, q_2, q_3, m_1, \ldots, m_{q_2}, m_1, \ldots, m_{q_3}$ be positive integers such that $\sigma = q_1 N + m_1 + \ldots + m_{q_2} + m_1 + \ldots + m_{q_3} \leqslant 2^{n-1}$. Fix $q = q_1 + q_2 + q_3$ bit strings $M_1^{(i)}, \ldots, M_{q_i}^{(i)}$, $1 \leqslant i \leqslant 3$ such as*

- *$M_1^{(1)}, \ldots, M_{q_1}^{(1)} \in \{0,1\}^{n \cdot N}$ are different;*
- *$M_1^{(2)}, \ldots, M_{q_2}^{(2)}$ are different and $M_j^{(2)} \in \{0,1\}^{n \cdot m_j}$, $j = 1, \ldots, q_2$;*
- *$M_1^{(3)}, \ldots, M_{q_3}^{(3)}$ are different and $M_j^{(3)} \in \{0,1\}^{n \cdot m_j}$, $j = 1, \ldots, q_3$*

*and $q$ different bit strings $T_1^{(i)}, \ldots, T_{q_i}^{(i)} \in \{0,1\}^n$, $1 \leqslant i \leqslant 3$ such as $\{T_1^{(i)}, \ldots, T_{q_i}^{(i)}\}$ are different. Then the number of permutations $P_1, P_2, P_3 \in Perm(\{0,1\}^n)$ that satisfy conditions **

- *$CBCMAC_{P_1}(M_j^{(1)}) = T_j^{(1)}$, $j = 1, \ldots, q_1$;*
- *$\mathsf{MAC}_{P_1,P_2}(M_j^{(2)}) = T_j^{(2)}$, $j = 1, \ldots, q_2$;*
- *$\mathsf{MAC}_{P_1,P_3}(M_j^{(3)}) = T_j^{(3)}$, $j = 1, \ldots, q_3$;*

*is at least $((2^n)!)^3 \left(1 - \frac{2\sigma^2}{2^n}\right) \frac{1}{2^{nq}}$.*

*Proof.* The proof follows from Lemma 4 and it is similar to the proof of Lemma 5.3 from [18].

From Lemma 4 the number of permutations $P_1$ that fulfill the conditions

- $\mathrm{CBCMAC}_{P_1}(M_j^{(1)}) = T_j^{(1)}$, $j = 1, \ldots, q_1$;
- $\text{CBC-E}_{P_1}(M_j^{(2)}) \neq \text{CBC-E}_{P_1}(M_k^{(2)})$, $j \neq k$, $j, k = 1, \ldots, q_2$.
- $\text{CBC-E}_{P_1}(M_j^{(3)}) \neq \text{CBC-E}_{P_1}(M_k^{(3)})$, $j \neq k$, $j, k = 1, \ldots, q_3$.

is at least $(2^n)! \cdot \frac{1}{2^{q_1 n}} \left(1 - \frac{2\sigma^2}{2^n}\right)$.

Note that after fixing such permutation $P_1$ the arguments for both permutations those are defined while processing messages $M_j^{(2)}$ and $M_j^{(3)}$ are different. Thus for $P_2$ and $P_3$ it is sufficient to fix $q_2$ and $q_3$ images with values from the sets $\{T_1^{(2)}, \ldots, T_{q_2}^{(2)}\}$ and $\{T_1^{(3)}, \ldots, T_{q_3}^{(3)}\}$.

Hence the number of tuples $P_1, P_2, P_3 \in Perm(\{0,1\}^n)$ such that conditions

– $\mathrm{CBCMAC}_{P_1}(M_j^{(1)}) = T_j^{(1)},\ j = 1, \ldots, q_1$;
– $\mathsf{MAC}_{P_1,P_2}(M_j^{(2)}) = T_j^{(2)},\ j = 1, \ldots, q_2$;
– $\mathsf{MAC}_{P_1,P_3}(M_j^{(3)}) = T_j^{(3)},\ j = 1, \ldots, q_3$;

hold is at least

$$(2^n)! \cdot \frac{1}{2^{q_1 n}} \left(1 - \frac{2\sigma^2}{2^n}\right) \cdot (2^n - q_2)! \cdot (2^n - q_3)! \geqslant$$
$$\geqslant (2^n)! \cdot \frac{1}{2^{q_1 n}} \left(1 - \frac{2\sigma^2}{2^n}\right) \cdot \frac{(2^n)!}{2^{q_2 n}} \cdot \frac{(2^n)!}{2^{q_3 n}} = ((2^n)!)^3 \left(1 - \frac{2\sigma^2}{2^n}\right) \frac{1}{2^{qn}}.$$

□

**Lemma 6.** *Let $A$ be an adversary in the $3PRF_N$ model for three permutations. Let the total block length of his queries be $\sigma$ where $\sigma \leqslant 2^{n-1}$. Thus,*

$$\mathsf{Adv}^{3PRF_N}_{\mathcal{F}_{3Perm}}(A) \leqslant \frac{2.5\sigma^2}{2^n}.$$

*Proof.* The proof follows from Lemma 5 and it is similar to the proof of Lemma 5.1 from [18].

We can assume without loss of generality that all adversaries are deterministic due to the fact that the adversary advantage does not depend on his time complexity (only on queries complexity).

Assume that the adversary $A$ made $q = q_1 + q_2 + q_3$ queries

– $q_1$ queries $M_j^{(1)} \in \{0,1\}^{nN}, j = 1, \ldots, q_1$, to the oracle $\mathsf{F}_1^b$ and received values
$T_j^{(1)} \in \{0,1\}^n, j = 1, \ldots, q_1$ as an answer;
– $q_2$ queries $M_j^{(2)} \in \{0,1\}_n^{\leqslant nN}, j = 1, \ldots, q_2$, to the oracle $\mathsf{F}_2^b$ and received values
$T_j^{(2)} \in \{0,1\}^n, j = 1, \ldots, q_2$ as an answer;
– $q_3$ queries $M_j^{(3)} \in \{0,1\}_n^{\leqslant nN}, j = 1, \ldots, q_3$, to the oracle $\mathsf{F}_3^b$ and received values
$T_j^{(3)} \in \{0,1\}^n, j = 1, \ldots, q_3$ as an answer.

Since the adversary is deterministic the number of queries, the values of the queries, and the adversary's execution result are uniquely defined by the returning values $T_j^{(1)}, T_j^{(2)}, T_j^{(3)}$

By $\mathcal{T}_{one}$ we denote the set of tuples $T$ those consist of $q$ elements from $\{0,1\}^n$

$$T = \{T_1^{(1)}, \ldots, T_{q_1}^{(1)}, T_1^{(2)}, \ldots, T_{q_2}^{(2)}, T_1^{(3)}, \ldots, T_{q_3}^{(3)}\},$$

such that the adversary returns 1 as result of his execution. By $\mathcal{T}_{good}$ we denote the set of tuples $T$ such that all elements in tuple are different. Note that the cardinality of the complement set is at most $\binom{q}{2}\frac{2^{qn}}{2^n}$. Thus,

$$|\{T : T \in \mathcal{T}_{good} \cap \mathcal{T}_{one}\}| \geqslant |\mathcal{T}_{one}| - \binom{q}{2}\frac{2^{qn}}{2^n}.$$

By definition,

$$\mathsf{Adv}^{3\mathrm{PRF}_N}_{\mathcal{F}_{3Perm}}(A) = \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right].$$

Obviously,

$$\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right] = \Pr_{F_1,F_2,F_3}\left[1 \leftarrow A^{\mathsf{F}_1^0,\mathsf{F}_2^0,\mathsf{F}_3^0}\right] = \sum_{T\in\mathcal{T}_{one}} \frac{1}{2^{qn}} = \frac{|\mathcal{T}_{one}|}{2^{qn}}.$$

Now lower estimate $\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right]$.

$$\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right] =$$

$$= \Pr_{P_1,P_2,P_3}\left[1 \leftarrow A^{\mathsf{F}_1^1,\mathsf{F}_2^1,\mathsf{F}_3^1}\right] = \frac{|\{(P_1,P_2,P_3) : 1 \leftarrow A^{\mathsf{F}_1^1,\mathsf{F}_2^1,\mathsf{F}_3^1}\}|}{((2^n)!)^3} \geqslant$$

$$\geqslant \sum_{T\in\mathcal{T}_{good}\cap\mathcal{T}_{one}} \frac{|\{(P_1,P_2,P_3)\ satisfying\ *5\}|}{((2^n)!)^3} \geqslant \left(|\mathcal{T}_{one}| - \binom{q}{2}\frac{2^{qn}}{2^n}\right)\cdot\left(1 - \frac{2\sigma^2}{2^n}\right)\frac{1}{2^{nq}} =$$

$$= \left(\frac{|\mathcal{T}_{one}|}{2^{qn}} - \binom{q}{2}\frac{1}{2^n}\right)\cdot\left(1 - \frac{2\sigma^2}{2^n}\right) \geqslant \frac{|\mathcal{T}_{one}|}{2^{qn}} - \binom{q}{2}\frac{1}{2^n} - \frac{2\sigma^2}{2^n} \geqslant \frac{|\mathcal{T}_{one}|}{2^{qn}} - \frac{2.5\sigma^2}{2^n}.$$

Therefore,

$$\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right] \leqslant \frac{2.5\sigma^2}{2^n}.$$

Applying similar arguments to $\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_{3Perm}}(A) = 0\right]$ and $\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) = 0\right]$ lead to the final estimation

$$\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_{3Perm}}(A) \Rightarrow 1\right] \leqslant \frac{2.5\sigma^2}{2^n}.$$

□

**Lemma 7.** *Let $A$ be an adversary in the $3PRF_N$ model for the $\mathcal{F}_E$ family. Let the total block length of his queries be $\sigma$ where $\sigma \leqslant 2^{n-1}$. Then exists the adversary $B$ in the PRP-CPA model for the blockcipher $E$ making at most $\sigma$ queries to his oracle such that*

$$\mathsf{Adv}_E^{PRP\text{-}CPA}(B) \geqslant \mathsf{Adv}_{\mathcal{F}_E}^{3PRF_N}(A) - \frac{3.5\sigma^2}{2^n}.$$

*Proof.* The proof directly follows from Lemmas 1, 2, 3, 6. □

**Definition 17.** *Let us define the family of random functions*

$$OMAC\text{-}RSF \subseteq Func(\{0,1\}_n^*, \{0,1\}^n).$$

*Each element $F$ of this family matches functions $G_1, G_2, G_3$ where*

1. $G_1 \in Func(\{0,1\}^{nN}, \{0,1\}^n)$;
2. $G_2 \in Func(\{0,1\}_n^*, \{0,1\}^n)$;
3. $G_3 \in Func(\{0,1\}_n^{\leqslant nN}, \{0,1\}^n)$.

*For any message $M \in \{0,1\}_n^*$, $M = M^1||\ldots||M^l$, $l \in \mathbb{N}$ where $M^1, \ldots, M^{l-1} \in \{0,1\}^{nN}$, $M^l \in \{0,1\}^{nR}$, $R \leqslant N$, each element $F$ of the family is defined by the following way*

1. *If $|M| \leqslant nN$, then $F(M) = G_3(M)$;*
2. *if $|M| > nN$, then*

$$F(M) = G_2\left((G_1(M^1) \oplus M^2[1]) \| M^2[2] \| \ldots \| M^2[N] \| M^3 \| \ldots \| M^l\right);$$

*We define the uniform distribution $\mathcal{U}$ on OMAC-RSF as follows:*

$$\begin{aligned}\Pr_{F \xleftarrow{\mathcal{U}} OMAC\text{-}RSF}[F] = \\ = \Pr_{G_3 \xleftarrow{\mathcal{U}} Func(\{0,1\}_n^*, \{0,1\}^n)}[G_3] \cdot \Pr_{G_1 \xleftarrow{\mathcal{U}} Func(\{0,1\}^{nN}, \{0,1\}^n)}[G_1] \cdot \\ \cdot \Pr_{G_2 \xleftarrow{\mathcal{U}} Func(\{0,1\}_n^{\leqslant nN}, \{0,1\}^n)}[G_2], \ \forall\ F \in OMAC\text{-}RSF\end{aligned}$$

Let us prove that OMAC-RSF is indistinguishable from the family $Func(\{0,1\}_n^*, \{0,1\}^n)$ with uniform distribution defined on it.

**Lemma 8.** *For any adversary $A$ making at most $q$ queries*

$$\mathsf{Adv}_{OMAC\text{-}RSF}^{PRF}(A) \leqslant \frac{q^2}{2^{n+1}};$$

*Proof.* Suppose the adversary making $q$ different queries $M_1, \ldots, M_q$. Consider the following cases:

1. $|M_i| \leqslant nN$ for all $i \in \{1, \ldots, q\}$;

2. $M_i^1 = M_j^1$ for all $i, j \in \{1, \ldots, q\}$, such that $|M_i| > nN$, $|M_j| > nN$;
3. $\exists\, i, j \in \{1, \ldots, q\}$, such that $|M_i| > nN$, $|M_j| > nN$, where $M_i^1 \neq M_j^1$;

Let the oracle return a value $T_i \in \{0,1\}^n$ on receiving query $M_i$, $1 \leqslant i \leqslant q$. Denote by $T \in \{0,1\}^{n \times q}$ the string $T_1 \| \ldots \| T_q$.

By definition, for any $M_1, \ldots, M_q$

$$
\begin{aligned}
&- \mathsf{Adv}^{\mathrm{PRF}}_{\text{OMAC-RSF}}(A) = \\
&= \sum_{T \in \{0,1\}^{n \times q}} \Pr\left[A \Rightarrow 1 | T\right] \cdot \left(\Pr_{F \xleftarrow{\mathcal{U}} Func(\{0,1\}^*, \{0,1\}^n)}[T] - \Pr_{F \xleftarrow{\mathcal{U}} \mathcal{G}}[T]\right) \leqslant \\
&\qquad \leqslant \sum_{T \in \{0,1\}^{n \times q} : \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}[T] < \frac{1}{2^{qn}}} \left(\frac{1}{2^{qn}} - \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}[T]\right).
\end{aligned}
$$

Obviously, $\mathsf{Adv}^{\mathrm{PRF}}_{\text{OMAC-RSF}}(A) = 0$ for the first two cases.

Consider the third case.

Denote by $Coll$ the following event: $\exists\, i, j \in \{1, \ldots, q\}$, such that $|M_i| > nN$, $|M_j| > nN$, where $F_3(M_i^1) \oplus M_i^2[1] = F_3(M_j^1) \oplus M_j^2[1]$.

Let us consider the value $\Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}[T]$ in more detail.

$$
\begin{aligned}
\Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}[T] &= \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}[T \mid Coll] \cdot \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}[Coll] + \\
&\quad + \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}\left[T \mid \overline{Coll}\right] \cdot \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}\left[\overline{Coll}\right] \geqslant \\
&\geqslant \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}\left[T \mid \overline{Coll}\right] \cdot \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}\left[\overline{Coll}\right] = \frac{1}{2^{qn}} \cdot \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}\left[\overline{Coll}\right].
\end{aligned}
$$

The probability of $Coll$ is at most $\binom{q}{2}\frac{1}{2^n}$. Thus,

$$
\Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}[T] \geqslant \frac{1}{2^{qn}}\left(1 - \binom{q}{2}\frac{1}{2^n}\right) \geqslant \frac{1}{2^{qn}}\left(1 - \frac{q^2}{2^{n+1}}\right).
$$

Thus, for any $A$

$$
\begin{aligned}
- \mathsf{Adv}^{\mathrm{PRF}}_{\text{OMAC-RSF}}(A) &\leqslant \sum_{T \in \{0,1\}^{n \times q} : \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}[T] < \frac{1}{2^{qn}}} \left(\frac{1}{2^{qn}} - \Pr_{F \xleftarrow{\mathcal{U}} \text{OMAC-RSF}}[T]\right) \leqslant \\
&\leqslant 2^{nq} \cdot \frac{q^2}{2^{n+1}} \cdot \frac{1}{2^{qn}} \leqslant \frac{q^2}{2^{n+1}}.
\end{aligned}
$$

Using the same arguments it can be shown that

$$\begin{aligned}\mathsf{Adv}^{\mathrm{PRF}}_{\text{OMAC-RSF}}(A) &= \Pr\left[\mathbf{Exp}^{\mathrm{PRF}-1}_{\text{OMAC-RSF}}(A) \Rightarrow 1\right] - \Pr\left[\mathbf{Exp}^{\mathrm{PRF}-0}_{\text{OMAC-RSF}}(A) \Rightarrow 1\right] = \\ &= (1 - \Pr\left[\mathbf{Exp}^{\mathrm{PRF}-0}_{\text{OMAC-RSF}}(A) \Rightarrow 1\right]) - (1 - \Pr\left[\mathbf{Exp}^{\mathrm{PRF}-1}_{\text{OMAC-RSF}}(A) \Rightarrow 1\right]) = \\ &= \Pr\left[\mathbf{Exp}^{\mathrm{PRF}-0}_{\text{OMAC-RSF}}(A) \Rightarrow 0\right] - \Pr\left[\mathbf{Exp}^{\mathrm{PRF}-1}_{\text{OMAC-RSF}}(A) \Rightarrow 0\right] = \\ &= -\mathsf{Adv}^{\mathrm{PRF}}_{\text{OMAC-RSF}}(A') \leqslant \frac{q^2}{2^{n+1}}.\end{aligned}$$

□

**Theorem 4.** *Let $N$ be the parameter of the OMAC-RK mode. Then for any adversary $A$ which makes queries, where the maximal message length is at most $m$ blocks and the total message length is at most $\sigma$ blocks, there exists a set of adversaries $B_j$, $j = 1, \ldots, \lceil m/N \rceil$, such that*

$$\sum_{j=1}^{\lceil m/N \rceil} \mathsf{Adv}^{3PRF_N}_{\mathcal{F}_E}(B_j) \geqslant \mathsf{Adv}^{PRF}_{OMAC\text{-}RK_N}(A) - \frac{\left(\sigma_1^2 + \ldots + \sigma^2_{\lceil m/N \rceil}\right)}{2^{n+1}}.$$

*where $B_j$ makes queries where the total message length is at most $\sigma_j$ blocks, $\sigma_j$ is the total block length of data processed under the $K^j$ section key, $\sigma_j \leqslant 2^{n-1}$, $\sigma_1 + \ldots + \sigma_{\lceil m/N \rceil} = \sigma$.*

*Proof.* Define a set of hybrid experiment $Hybrid_j(A)$, $j \in \{0, 1, \ldots, \lceil m/N \rceil\}$. In the $Hybrid_j(A)$ experiment the $\mathsf{F}$ oracle is replaced by the $\mathsf{F}_j$ oracle that processes input queries as follows

- The $\mathsf{F}$ oracle chooses $j$ keys $K^1, \ldots, K^j \xleftarrow{\mathcal{U}} \{0,1\}^k$ and $j$ blocks $K^1_1, \ldots, K^j_1 \xleftarrow{\mathcal{U}} \{0,1\}^n$. Also the oracle chooses the functions $F_1, F_2 \xleftarrow{\mathcal{U}} Func(\{0,1\}^*_n, \{0,1\}^n)$ at random.
- On the query $M$, $|M|_n \leqslant m$, he returns a value $T \in \{0,1\}^n$, which is formed as follows. The first $j$ sections are processed under the first $j$ keys and $j$ blocks. If $|M|_n > jN$, i.e. $M = M^1 || \ldots || M^j || M'$, $|M'|_n \leqslant (\lceil m/N \rceil - j)N$, then the oracle computes a value $F_1(M'')$, if $n \mid |M'|$, or $F_2(M'')$, otherwise, where the message $M''$ is formed as follows. The first block $M''[1]$ of the message $M''$ is computed by xoring to the first block of the message $M' = \mathrm{pad}_n(M')$ the result of the first $j$ section processing:

$$M''[1] = M'[1] \oplus \mathrm{CBCMAC}_{E_{K^1}, \ldots E_{K^j}}(M^1 || \ldots || M^j).$$

  The blocks $M''[i]$, $i \geqslant 2$, are equal to the blocks $M'[i]$.

The result of the any hybrid experiment is what the adversary $A$ returns.

Note that the $Hybrid_{\lceil m/N \rceil}(A)$ experiment totally coincidences with the $\mathbf{Exp}^{\mathrm{PRF}-1}_{\text{OMAC-RK}_N}(A)$ experiment, and $Hybrid_0(A)$ — with the $\mathbf{Exp}^{\mathrm{PRF}-0}_{\text{OMAC-RK}_N}(A)$ experiment. Thus,

$$\Pr\left[Hybrid_{\lceil m/N \rceil}(A) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\mathrm{PRF}-1}_{\text{OMAC-RK}_N}(A) \Rightarrow 1\right],$$

$$\Pr\left[Hybrid_0(A) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\mathrm{PRF}-0}_{\text{OMAC-RK}_N}(A) \Rightarrow 1\right].$$

The last equality holds due to that the family of random functions, each of which is realized using a couple of random functions $F_1, F_2 \xleftarrow{\mathcal{U}} Func(\{0,1\}^*_n, \{0,1\}^n)$, where $F_1$ processes message of complete length and $F_2$ processes incomplete messages modified by the function $\mathrm{pad}_n$, is statistically indistinguishable from the standard family of random functions $F \xleftarrow{\mathcal{U}} Func(\{0,1\}^*, \{0,1\}^n)$.

Construct a set of adversaries $B_j$ in the $3\mathrm{PRF}_N$ model for the used block cipher $E$, using $A$ as a black box.

The adversary $B_j$ chooses the following random values:

1. $j-1$ keys $K^1, K^2, \ldots, K^{j-1} \xleftarrow{\mathcal{U}} \{0,1\}^k$;
2. $j-1$ blocks $K_1^1, K_1^2, \ldots, K_1^{j-1} \xleftarrow{\mathcal{U}} \{0,1\}^n$;
3. random functions $F_1, F_2 \xleftarrow{\mathcal{U}} Func(\{0,1\}^*_n, \{0,1\}^n)$.

The adversary $B_j$ model the experiment $Hybrid_j(A)$ except for the $j$-th section process. While processing the $j$-th section the adversary makes queries to the oracles $\mathsf{F}_1^b,\ \mathsf{F}_2^b,\ \mathsf{F}_3^b$:

– query $(M')^j$ to the $\mathsf{F}_1^b$ oracle if the $j$-th section is intermediate;
– query $(M')^j$ to the $\mathsf{F}_2^b$ oracle if the $j$-th section is final and complete;
– query $(M')^j$ to the $\mathsf{F}_3^b$ oracle if the $j$-th section is final and incomplete;

where $(M')^j$ is formed as follows. The first block $(M')^j[1]$ of the message $(M')^j$ is computed by xoring to the first block of the message $M^j = \mathrm{pad}_n(M^j))$the result of the first $j$ section processing:

$$(M')^j[1] = M^j[1] \oplus \mathrm{CBCMAC}_{K^1,\ldots K^{j-1}}(M_1 || \ldots || M_{j-1}).$$

The next blocks $(M')^j[i]$, $i \geqslant 2$, are equal to blocks $M^j[i]$. The adversary $B_j$ returns as a result what $A$ returns.

The following equalities hold

$$\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-1}_{\mathcal{F}_E}(B_j) \Rightarrow 1\right] = \Pr\left[Hybrid_j(A) \Rightarrow 1\right],$$

$$\Pr\left[\mathbf{Exp}^{3\mathrm{PRF}_N-0}_{\mathcal{F}_E}(B_j) \Rightarrow 1\right] = \Pr\left[Hybrid'_{j-1}(A) \Rightarrow 1\right],$$

where $Hybrid'_j(A)$ is a modification of the $Hybrid_j(A)$ experiment: the $j+1$-th section is processed using independent random functions $G_1, G_2, G_3$:

– using $G_1 \xleftarrow{\mathcal{U}} Func(\{0,1\}^{nN}, \{0,1\}^n)$ if the $j+1$-th section is intermediate;

– using $G_2 \xleftarrow{\mathcal{U}} Func(\{0,1\}_n^{\leqslant nN}, \{0,1\}^n)$ if the $j+1$-th section is final and complete;

– using $G_3 \xleftarrow{\mathcal{U}} Func(\{0,1\}_n^{\leqslant nN}, \{0,1\}^n)$ if the $j+1$-th section is final and incomplete;

Then, for advantages of $B_j$ the following estimations hold

$$\sum_{j=1}^{\lceil m/N \rceil} \mathsf{Adv}^{3\mathrm{PRF}_N}_{\mathcal{F}_E}(B_j) = \sum_{j=1}^{\lceil m/N \rceil} \Pr\left[Hybrid_j(A) \Rightarrow 1\right] - \sum_{j=1}^{\lceil m/N \rceil} \Pr\left[Hybrid'_{j-1}(A) \Rightarrow 1\right] =$$

$$= \Pr\left[Hybrid_{\lceil m/N \rceil}(A) \Rightarrow 1\right] - \Pr\left[Hybrid_0(A) \Rightarrow 1\right] +$$

$$+ \sum_{j=0}^{\lceil m/N \rceil - 1} \left(\Pr\left[Hybrid_j(A) \Rightarrow 1\right] - \Pr\left[Hybrid'_j(A) \Rightarrow 1\right]\right) =$$

$$= \mathsf{Adv}^{\mathrm{PRF}}_{\mathrm{OMAC\text{-}RK}_N}(A) + \sum_{j=0}^{\lceil m/N \rceil - 1} \left(\Pr\left[Hybrid_j(A) \Rightarrow 1\right] - \Pr\left[Hybrid'_j(A) \Rightarrow 1\right]\right).$$

Using Lemma 8 for all $j = 0, \ldots, \lceil m/N \rceil - 1$, we obtain

$$\Pr\left[Hybrid_j(A) \Rightarrow 1\right] - \Pr\left[Hybrid'_j(A) \Rightarrow 1\right] = -\mathsf{Adv}^{\mathrm{PRF}}_{\mathrm{OMAC\text{-}RSF}}(A'_j) -$$

$$- \mathsf{Adv}^{\mathrm{PRF}}_{\mathrm{OMAC\text{-}RSF}}(A''_j) \geqslant -\frac{(q'_{j+1})^2}{2^{n+1}} - \frac{(q''_{j+1})^2}{2^{n+1}} \geqslant -\frac{\sigma^2_{j+1}}{2^{n+1}},$$

where $A'_j$ and $A''_j$ are the adversaries for OMAC-RSF in the PRF model, those simulate the $Hybrid_j(A)$ experiment except for functions $F_1$ and $F_2$ using way: the adversary $A'_j$ makes $q'_{j+1}$ queries in order to process complete pieces $M'$ of messages and the adversary $A''_j$ makes $q''_{j+1}$ queries in order to process incomplete pieces $M'$ of messages.

Thus,

$$\sum_{j=1}^{\lceil m/N \rceil} \mathsf{Adv}^{3\mathrm{PRF}_N}_{\mathcal{F}_E}(B_j) \geqslant \mathsf{Adv}^{\mathrm{PRF}}_{\mathrm{OMAC\text{-}RK}_N}(A) - \frac{\left(\sigma_1^2 + \ldots + \sigma^2_{\lceil m/N \rceil}\right)}{2^{n+1}}.$$

□

**Lemma 9.** *Let $N$ be the parameter of the OMAC-RK mode. Then for any adversary $A$ which makes queries, where the maximal message length is at most $m$ blocks and the total message length is at most $\sigma$ blocks, there exists*

*adversary $B$ such that*

$$\mathsf{Adv}^{PRF}_{OMAC\text{-}RK_N}(A) \leqslant \frac{4\left(\sigma_1^2 + \ldots + \sigma^2_{\lceil m/N \rceil}\right)}{2^n} + \left\lceil \frac{m}{N} \right\rceil \cdot \mathsf{Adv}^{PRP\text{-}CPA}_{E}(B)$$

*where $B$ makes at most $\sigma_1$ queries, $s = \lceil k/n \rceil$, $\sigma_j$ is the total block length of data processed under the $K^j$ section key, $\sigma_j \leqslant 2^{n-1}$, $\sigma_1 + \ldots + \sigma_{\lceil m/N \rceil} = \sigma$.*

*Proof.* The proof follows from Lemma 7, 4. □

**Lemma 10.** *Let $N$ and $T^*$ be the parameters of the OMAC-ACPKM-Master mode. Then for any adversary $A$ which makes queries, where the maximal message length is at most $m$ blocks and the total message length is at most $\sigma$ blocks, there exists adversaries $B$ and $C$ such that*

$$\mathsf{Adv}^{PRG}_{ACPKM\text{-}Master_{T^*}}(B) + \mathsf{Adv}^{PRF}_{OMAC\text{-}RK_N}(C) \geqslant \mathsf{Adv}^{PRF}_{OMAC\text{-}ACPKM\text{-}Master_{N,T^*}}(A).$$

*where $B$ makes query of length at most $\lceil m/N \rceil \cdot d$ blocks, $d = \lceil k/n \rceil + 1$, and $C$ makes queries, where the maximal message length is at most $m$ blocks and the total message length is at most $\sigma$ blocks.*

*Proof.* Construct the adversary $C$ in the PRF model for the OMAC-RK$_N$ mode, using $A$ as a black box. The adversary $C$ having access to the $\mathsf{F}^b$ oracle simulates the $\mathbf{Exp}^{\text{PRF}-b'}_{\mathcal{F}_{Perm}}$ experiment for $A$ by transmitting queries of $A$ to its own oracle and vice versa. The adversary $C$ returns as a result what $A$ returns.

Note that in the case $b = 0$, $C$ simulates exactly the experiment $\mathbf{Exp}^{\text{PRF}-0}_{\text{OMAC-ACPKM-Master}_{N,T^*}}$. Thus,

$$\Pr\left[\mathbf{Exp}^{\text{PRF}-0}_{\text{OMAC-RK}_N}(C) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\text{PRF}-0}_{\text{OMAC-ACPKM-Master}_{N,T^*}}(A) \Rightarrow 1\right],$$

$$\Pr\left[\mathbf{Exp}^{\text{PRF}-1}_{\text{OMAC-RK}_N}(C) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\text{PRF}-1}_{\text{OMAC-RK}_N}(A) \Rightarrow 1\right].$$

Now construct the adversary $B$ in the PRG model for ACPKM-Master$_{T^*}$ mechanism, using the same adversary $A$ as a black box.

The adversary $B$ simulates the experiment $\mathbf{Exp}^{\text{PRF}-b'}_{\text{OMAC-ACPKM-Master}_{N,T^*}}$ for $A$ in the following way. Obtaining keys $K^1, \ldots, K^l \in \{0,1\}^k$ and $K^1_1, \ldots, K^l_1 \in \{0,1\}^n$, the adversary $B$ in the case $b = 1$ simulates exactly the experiment $\mathbf{Exp}^{\text{PRF}-1}_{\text{OMAC-ACPKM-Master}_{N,T^*}}(A)$, and in the case $b = 0$ it simulates the experiment $\mathbf{Exp}^{\text{PRF}-1}_{\text{OMAC-RK}_N}(A)$. Thus,

$$\Pr\left[\mathbf{Exp}^{\text{PRG}-1}_{\text{ACPKM-Master}_{T^*}}(B) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\text{PRF}-1}_{\text{OMAC-ACPKM-Master}_{N,T^*}}(A) \Rightarrow 1\right]$$

$$\Pr\left[\mathbf{Exp}^{\text{PRG}-0}_{\text{ACPKM-Master}_{T^*}}(B) \Rightarrow 1\right] = \Pr\left[\mathbf{Exp}^{\text{PRF}-1}_{\text{OMAC-RK}_N}(A) \Rightarrow 1\right]$$

Using similar to Lemma 7 arguments, we obtain

$$\mathsf{Adv}^{\text{PRG}}_{\text{ACPKM-Master}_{T^*}}(B) = \mathsf{Adv}^{\text{PRF}}_{\text{OMAC-ACPKM-Master}_{N,T^*}}(A) - \mathsf{Adv}^{\text{PRF}}_{\text{OMAC-RK}_N}(C).$$

□

The proof of Theorem 1 follows from Lemma 3, Lemma 9, Lemma 10.

## E Bounds comparison

### E.1 CTR and CTR-ACPKM

In all cases we analyze the parameters for which the following inequality holds:

$$\sum_{1 \leqslant i,j \leqslant l} \sigma_i \sigma_j \geqslant 2s(\sigma_1 + \ldots + \sigma_{l-1}) + s^2(l-1).$$

**The case $l = 2$.** Using the restriction $s \leqslant \min(\sqrt{2N}, \sigma_2)$ we obtain

$$2\sigma_1(\sigma_2 - s) \geqslant 2N(\sigma_2 - s) \underbrace{\geqslant}_{\sigma_2 \geqslant s} 2N \underbrace{\geqslant}_{\sqrt{2N} \geqslant s} s^2.$$

Thus, $2\sigma_1(\sigma_2 - s) \geqslant s^2 \quad \Rightarrow \quad 2\sigma_1\sigma_2 \geqslant 2s\sigma_1 + s^2$.

**The case $l = 3$.** Using the restriction $s \leqslant \min(\sqrt{2N}, N/2)$ we obtain

$$2\sigma_1\sigma_3 + 2\sigma_2\sigma_3 \underbrace{\geqslant}_{\sigma_3 \geqslant 1} 2\sigma_1 + 2\sigma_2 \geqslant 4N \underbrace{\geqslant}_{\sqrt{2N} \geqslant s} 2s^2;$$

$$\frac{\sigma_1\sigma_2}{\sigma_1 + \sigma_2} \underbrace{\geqslant}_{\sigma_2 \leqslant \sigma_3} \frac{\sigma_1\sigma_2}{2\sigma_1} \geqslant \frac{\sigma_2}{2} \underbrace{\geqslant}_{\sigma_2 \geqslant N} \frac{N}{2} \geqslant s.$$

Thus, $\frac{\sigma_1\sigma_2}{\sigma_1+\sigma_2} \geqslant s \quad \Rightarrow \quad \sigma_1\sigma_2 \geqslant 2s(\sigma_1 + \sigma_2)$. Summing up the inequalities we obtain

$$2\sigma_1\sigma_2 + 2\sigma_1\sigma_3 + 2\sigma_2\sigma_3 \geqslant 2s(\sigma_1 + \sigma_2) + 2s^2.$$

**The case $l \geqslant 4$.** Using the restriction $s \leqslant \sqrt{2N}$ we obtain

$$\sum_{1 \leqslant i,j \leqslant l} \sigma_i \sigma_j \underbrace{\geqslant}_{\sigma \geqslant 1} 2\sigma_1\sigma_2 + 2\sigma_2\sigma_3 + \ldots + 2\sigma_{l-1}\sigma_1 + (2\sigma_1 + \ldots + 2\sigma_{l-1}) \underbrace{\geqslant}_{\sigma_i > N}$$
$$\geqslant 2N(\sigma_1 + \ldots + \sigma_{l-1}) + 2N(l-1) \underbrace{\geqslant}_{2N > s^2} 2s(\sigma_1 + \ldots + \sigma_{l-1}) + s^2(l-1).$$

## E.2 OMAC and OMAC-ACPKM-Master

In all cases we analyze the parameters for which the following inequality holds:

$$4 \cdot \sum_{1 \leqslant i,j \leqslant l} \sigma_i \sigma_j \geqslant (dl)^2.$$

**The case $l = 2$.** Using the restriction $d \leqslant \min(N, 2\sigma_2)$ we obtain

$$8\sigma_1\sigma_2 \underbrace{\geqslant}_{\sigma_1 \geqslant N} 8N\sigma_2 \geqslant 4d^2.$$

**The case $l = 3$.** Using the restriction $d \leqslant \min(N, 16)$ we obtain

$$8\sigma_1\sigma_2 + 8\sigma_1\sigma_3 + 8\sigma_3\sigma_2 \underbrace{\geqslant}_{\sigma_3 \geqslant 1} 8\sigma_1\sigma_2 + 8\sigma_1 + 8\sigma_2 \underbrace{\geqslant}_{\sigma_1, \sigma_2 \geqslant N} 8N^2 + 16N \geqslant 9d^2;$$

**The case $l \geqslant 4$.** Using the restriction $d \leqslant N$ we obtain

$$4 \cdot \sum_{1 \leqslant i,j \leqslant l} \sigma_i \sigma_j \geqslant 4 \cdot \sum_{1 \leqslant i,j \leqslant l-1} \sigma_i \sigma_j \underbrace{\geqslant}_{\sigma_i \geqslant N} 4(l-1)(l-2)N^2 \underbrace{\geqslant}_{\boldsymbol{N} \geqslant d,\ l \geqslant 4} (dl)^2$$

# Provably Secure Counter Mode with Related Key-based Internal Re-keying

Evgeny Alekseev[1], Kirill Goncharenko[2], and Grigory Marshalko[3]

[1]Crypto-Pro LLC, Moscow, Russia
alekseev@cryptopro.ru
[2]Moscow State University, Moscow, Russia
al_tair94@mail.ru
[3]Technical Committee for Standardization
«Cryptography and security mechanisms» (TC 26), Moscow, Russia
marshalko_gb@tc26.ru

**Abstract**

Block cipher cryptanalysis in the related-key adversary model is usually underestimated, since the conditions of this model could be hardly obtained in practice. Nevertheless, the existence of a block cipher, which is investigated and proved to be secure in the related-key model, allows to create more efficient cryptographic protocols without significant performance loss. In this article we propose a key recovery attack on a reduced Kuznyechik block cipher in the related-key model, and discuss why it could be hard to extend such approach to the full cipher. We also propose a new internally re-keyed block cipher mode of operation called CTRR («CounTer with Related-key Re-keying»), and prove its security under the assumption of the underlying cipher (e.g. Kuznyechik) security in the related-key adversary model.

**Keywords:** related-key attack, provable security, internal re-keying, block cipher.

## 1 Introduction

The notion of a related-key attack (RKA) was formally introduced in [27] but similar ideas can be found in [28, 26]. This model assumes that the cryptosystem uses several keys, and the adversary knows a certain simple relationship between them (for example, a bitwise sum). Such conditions are less likely to be achievable in practice than the conditions of classical models suggesting the existence of only one unknown key. Actually there are a number of protocols and schemes, in which such conditions can arise (e.g. [29]). However, researchers pay less attention to cryptographic analysis in the related-key adversary model than in the classical models. In doing so, the presence of block cipher which is in-depth investigated and reasonably secure in a related-key model can help to solve a lot of problems of synthesis of

cryptographic protocols. Thus, it will allow creating more efficient cryptosystems due to the possibility of safe use of several keys connected by a simple fixed ratio. In a number of cases, this would allow not to use diversification functions for obtaining a sufficient number of keys. Problems of that nature are relevant and arise, for example, in the work of Russian Technical committee for standardization (TC26): the use of related keys would simplify the construction of some internally re-keyed block cipher modes of operation [15] and authenticated encryption modes.

In this paper, we present an effective method for recovering the key of a truncated version of the Kuznyechik block cipher [18] in a model with a known bitwise sum of key pairs. Also we give arguments in favour of the inability to improve this method for recovering the key of the full version of the cipher. To the best of our knowledge, the Kuznyechik was not investigated in this model before.

Also we propose internally re-keyed block cipher mode of operation called CTRR («CounTer with Related-key Re-keying») that use several keys with fixed bitwise or modulo $2^k$ sum. We obtain a lower security bound for this mode which depends on the security of the used cipher in the adversary model with two related keys. The use of related keys allows to solve certain problems inherent to classic single-key modes (for more details see Section 5.1). However, the degree of investigation of Kuznyechik in related-key model is currently too small to say confidently that the cipher is secure in this context and, as a consequence, to use CTRR in practice. The authors hope that this example will serve as a convincing illustration of the importance of carrying out further large-scale investigations of the Kuznyechik cipher in related-key model.

## 2 Basic notations and security notions

By $V_u$ we denote the set of $u$-component bit strings. Let $0^u$ be the string, consisting of $u$ zeros. For bit strings $U$ and $V$ we denote by $U\|V$ their concatenation. Let $|U|$ be the bit length of the string $U$.

For a bit string $U$ and a positive integer $l \leqslant |U|$ let $\mathrm{msb}_l(U)$ ($\mathrm{lsb}_l(U)$) be the string, consisting of the leftmost (rightmost) $l$ bits of $U$. For nonnegative integers $l$ and $i$ let $\mathrm{str}_l(i)$ be $l$-bit representation of $i$ with the least significant bit on the right. For a nonnegative integer $l$ and a bit string $U \in V_l$ let $\mathrm{int}(U)$ be an integer $i$ such that $\mathrm{str}_l(i) = U$. Let $\mathrm{Inc}(U)$ be the function, which takes the input $U \in V_u$ and outputs the string $\mathrm{msb}_{u/2}(U)\|\mathrm{str}_{u/2}(\mathrm{int}(\mathrm{lsb}_{u/2}(U)) + 1 \bmod 2^{u/2})$.

For any finite set $S$, define $Perm(S)$ as the set of all bijective mappings on $S$ (permutations on $S$). A block cipher $E$ (or just a cipher) with block size $n$ and key size $k$ is a family of permutations $E_K \in Perm(V_n)$, where $K \in V_k$ is a key. We let $Perm(V_k, V_n)$ denote the set of all blockciphers with domain $V_n$ and key-space $V_k$. If the value $s$ is chosen in a set $S$ uniformly at random, then we denote $s \in_{\mathcal{U}} S$. The notation $G \in_{\mathcal{U}} Perm(V_k, V_n)$ corresponds to selecting a random blockcipher. In more detail, it comes down to defining $G$ via $\forall K \in V_k : G_K(\cdot) \in_{\mathcal{U}} Perm(V_n)$.

For a bit string $U$, $|U| = l$, we denote by $U[i] \in V_n$, $1 \leqslant i \leqslant \lceil l/n \rceil - 1$, and $U[\lceil l/n \rceil] \in V_h$, $h \leqslant n$, such strings that $U = U[1] \| U[2] \| \ldots \| U[\lceil l/n \rceil]$ and call them «blocks» of the string $U$. We denote by the block-length $|U|_n = \lceil |U|/n \rceil$ the length of the string $U$ in blocks.

We model an adversary using an interactive probabilistic algorithm that has access to one or more oracles. Denote by $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \cdots}$ an adversary $\mathcal{A}$ that interacts with oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$ by making queries. Notation $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \cdots} \to b$ means that the algorithm $\mathcal{A}$, after interacting with oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$, outputs bit $b \in \{0, 1\}$. The resources of $\mathcal{A}$ are measured in terms of time and query complexities. For a fixed model of computation and a method of encoding the time complexity includes the description size of $\mathcal{A}$. The query complexity usually includes the number of queries and the maximal length of queries. By $\mathsf{Adv}^{\mathrm{M}}_{\mathrm{P}}(\mathcal{A})$ we denote the measure of the adversary $A$ success in realizing a certain threat, defined by the model M, for the cryptographic scheme P. The formal definition of this measure will be given in each specific case. By $\mathsf{InSec}^{\mathrm{M}}_{\mathrm{P}}(t, a, b, \ldots)$ we denote $\max\limits_{\mathcal{A} \in \mathcal{A}(t,a,b,\ldots)} \mathsf{Adv}^{\mathrm{M}}_{\mathrm{P}}(\mathcal{A})$, where $\mathcal{A}(t, a, b, \ldots)$ is the set of the adversaries, satisfying the limitations, which are defined by the values $t, a, b, \ldots$. Usually $t$ denotes the computational resources limitations of $\mathcal{A}$, and $a, b, \ldots$ denote its oracles queries limitations.

## 3 Related-key security model and provably secure protocols

In a related-key model an adversary has an opportunity to obtain the results of encryption not with one secret key used in the scheme or the protocol, but with several keys that satisfy some known relation. Note that the adversary does not know any of the used keys. In practice, these relations can be different, but the modulo $2^k$ addition or bit-wise addition with some constant value are the most often considered. The cases when the adversary is passive, that is, he only knows the relation between the keys on which the data was processed, or active, that is, he is able to choose this relation on his own, are also considered. Such conditions may seem unrealistic, but,

nevertheless, they can arise, for example, in the following cases:

– the adversary can mount a «man in the middle» attack on the key transfer protocol that does not provide integrity (e.g. [8, 9]);
– in some systems keys can be produced not equiprobable or be selected from a small set; thus, with a sufficient amount of data, the adversary can find the ciphertexts processed with the related keys (for example, such conditions have been implemented for the protocol WEP [22]);
– if the hash function is an iterative construction built upon the block-cipher then the opponent also has a possibility of manipulating the encryption keys (e.g. Davies-Meyer scheme [7]);
– some protocols may use the related keys by design.

A detailed discussion of how reasonably consider the capabilities described above for the adversary can be found in [29].

At the moment the results on the security in a related-key model have been obtained for a number of block ciphers. For example, in [20], a method with complexity of $2^{99.5}$ for recovering AES-256 secret key was proposed with requirement on the existence of 4 related keys. Also the results of analysis are known for such ciphers as SIMON [12], PICARO [23], MISTY1 [24], HAS-160 [25], A5/3 [6] and RC4 [22]. As for Russian block ciphers, the work [21] proposes a method for recovering of a secret key of Magma cipher, using 12 related keys, which works «in a number of cases for an acceptable time». At the same time, as far as the authors of this work are aware, the no results for Kuznyechik cipher in the related-key model are available at the moment. Note also that the related-keys model is mostly used in order to increase the efficiency of the differential method [10]. Also related-key attack can be enhanced into more complex techniques, such as boomerang method [11] or boomerang switching techinque used in [20].

The security of a block cipher in the related-key model does not follow from its security in the standard model PRP-CPA [4] (a simple example confirming this is given in Appendix C) and is evaluated heuristically (based on the results of large-scale researches). Opportunities of the adversary in the related-key model are wider than in standard models, therefore, usually creation of the secure (in this model) cipher and its analysis requires significantly more effort from cryptographers. So, there are additional requirements on key schedule. However, the effort to develop and analyze the cipher in the related-key model is compensated by the fact that more efficient cryptosystems can be built upon those ciphers. The efficiency is achieved due to the ability to work not only with independently generated keys, but with keys, related by a simple relationship. Thus, we can omit the additional operations

of the keys diversification. The use of such a cipher can also solve some issues of the purely cryptographic nature (for example, see Section 5.1).

Also the modern security approach, so-called provable security [13] must be taken into account while incorporating the cipher into the high level cryptosystems. One of the main advantages of cryptosystems constructed according to this approach is that their security is based on the minimum number of unproven, but only heuristically verified, assumptions. The security of such cryptosystems is often based on the security of block ciphers in a model with a threat of distinguishing from some ideal object. An analogue of the standard model PRP-CPA for the case when the adversary can use the relation between keys has been proposed in [17]. This model is usually referred to as PRP-RKA.

## 4 Security of Kuznyechik in Related-key model

In this section we propose a related-key attack on a reduced variant of block cipher Kuznyechik. The reduction assume a 4-round variant of Kuznyechik with a simplified key schedule. The proposed approach exploits the ability of attacker to manipulate keys, and the similarity of the functions in encryption and the key schedule procedures. The attack is divided into two sequential parts: $K_1$ and $K_2$ recovery, where $K_1$ and $K_2$ are parts of secret key ($K = K_1||K_2$).

### 4.1 Reduced block cipher

We will use basic notations according to [18]. The description of Kuznyechik block cipher could also be found in [18].

The reduced version of Kuznyechik block cipher has only 4 rounds of the basic cipher, and each round of the key schedule has only 2 rounds of basic cipher's Feistel rounds.

The key schedule of the reduced cipher uses round constants $\mathbf{C_i} \in V_{128}$, $i = 1, 2, \ldots, 32$, which a defined as follows:

$$\mathbf{C_i} = \mathbf{L}(\mathbf{i}),\ i = 1, 2.$$

**Round keys** $\mathbf{K_i} \in V_{128}$, $i = 1, 2, 3, 4$, are derived from

$$K = k_{255}|| \ldots ||k_0 \in V_{256},$$

$k_i \in V_1$, $i = 0, 1, \ldots, 255$, and evaluated according to the following equations:

$$K_1 = k_{255}|| \ldots ||K_{128};$$
$$K_2 = k_{127}|| \ldots ||k_0;$$
$$(K_3, K_4) = F[C_2]F[C_1](K_1, K_2).$$

Ciphertext obtained as a result of the following transformation.

$$E_{K_1,K_2}(m) = X[K_4]LSX[K_3]LSX[K_2]LSX[K_1](m)$$

### 4.1.1 Attack description

Here we describe the proposed attack in brief. The formal description could be found in the Appendix A.

Key recovery consists of two phases: at the first phase we recover $K_1$ by 8-bit words, and at the second simply evaluate $K_2$, using obtained $K_1$.

Let $L_i(\delta)$ be a vector $L(\delta \lll 8i)$, where $\delta \in V_8$. Since $L$ is linear $\forall x \in V_{128} : L(x \oplus (0 \ldots 0, \underbrace{\delta}_{i}, 0 \ldots 0)) = L(x) \oplus L_i(\delta)$. Let also $A^i$ be a $i$-th 8-bit word of $A \in V_{128}$. So, $A = (A^{15}, \ldots, A^0)$.

Assume, for example, that $C_1^0 \oplus K_1^0 = \alpha$, and $\pi(\alpha) \oplus \pi(\alpha \oplus 1) = \beta$. Consider encryption procedure for plaintext $C_1$ with initial and related keys, when the latter is obtained by the following transformation: $K_1' = K_1 \oplus 00000001, K_2' = K_2 \oplus L_0(\beta)$.

Since we have the same plain text $m$ for both cases, initial internal states are the same and the corresponding difference is equal to zero ($\Delta = 0$). After bitwise XOR with the first round key we will have $\Delta = K_1 \oplus K_1' = (0, \ldots, 0, 1)$, because of the relation between the keys. After the consequent S-box would change the difference only in one active S-box. For this S-box input difference is 1. Then, the output difference is equal to $\beta$, and as a result $\Delta = (0, \ldots, \beta)$. The application of the linear transform affects the whole internal state: $\Delta = L_0(\beta)$.

Since $K_2 \oplus K_2' = L_0(\beta)$, bitwise XOR with the second round key makes internal states identical again ($\Delta = 0$). The consequent nonlinear and linear transforms do not affect this zero difference. Since $K_2 \oplus LSX[C_1](K_1) = K_2' \oplus LSX[C_1](K_1')$, $LSX[C_2](K_2 \oplus LSX[C_1](K_1)) = LSX[C_2](K_2' \oplus LSX[C_1](K_1'))$, after the next XOR with the round key we have $\Delta = K_1 \oplus K_1' = (0, \ldots, 1)$. As a result after the next steps we again have only one active S-box (which is actually used as a distinguisher), and $\Delta$ became an unknown difference $(0, \ldots, \gamma)$. After the linear transform we have $\Delta = L_0(\gamma)$. Since $K_2 \oplus LSX[C_1](K_1) = K_2' \oplus LSX[C_1](K_1')$, after the

final step the difference between the ciphertexts $\Delta$ is still equal to $L_0(\gamma)$.

So, the distinguisher used for correct key selection is based on the fact that before the last application of linear transform and XOR with the round key internal state has only 256 different values.

The distinguisher for $K_1$ works due to the following reasons. For the considered keys the difference between ciphertexts is equal to $L_0(\gamma)$. Despite the fact that the value of $\gamma$ is unknown, the attacker knows that all 8-bit subwords of $L^{-1}(L_0(\gamma))$ are equal to zero except the first one. This is used as a distinguishing criterion. Note, that if the attacker uses only one related keys pair at step 2.a0 (see Appendix A), the distinguishing step would have been passed by all $x$, such that $\pi(x) \oplus \pi(x \oplus 1) = \beta$. There are at least 2 such pairs: $C_1^l \oplus K_1^l$ and $C_1^l \oplus K_1^l \oplus 1$. As result the attacker is unable to learn the right value $k$.

In order to override this problem the attacker should use two differences for the $l$-th subword of $K_1$: $K_1^l \oplus K_{1,l}' = 1$ and $K_1^l \oplus K_{6,l}' = 6$. By total search it could be shown the the pair $\pi(x) \oplus \pi(x \oplus 1)$, $\pi(x) \oplus \pi(x \oplus 6)$ is unique for all $x = 0, 1, \ldots, 255$, that means that there are no $0 \leq x < y \leq 255$, such that $\pi(x) \oplus \pi(x \oplus 1) = \pi(y) \oplus \pi(y \oplus 1)$ and $\pi(x) \oplus \pi(x \oplus 6) = \pi(y) \oplus \pi(y \oplus 6)$.

For other related keys pairs, in order to determine the probability of type 1 error, we consider the block cipher as a random substitution of $V_{128}$ . For such assumption the probability, that for some other related keys pair the result of encryption would satisfy the distinguishing criterion, is upper bounded by $2^{-128} \cdot 2^8 = 2^{-120}$. The same estimates are true for $l = \overline{1, 15}$. The expected number of additional candidates for $K_1$ is equal to $(1 + 2^{-120})^{16} - 1 \approx 2^{-116}$, so we could consider that the unique key $K_1$ is found.

### 4.1.2 Attack parameters

In order to find the right $K_1^l$ the attacker needs one plaintext, up to 512 related keys and the same number of encryptions. In order to determine $K_2$ the attacker needs one encryption under the obtained key. As a result, the overall time complexity of the attack is $2^{12}$ encryptions and $2^{12}$ related keys.

If the attacker would evaluate the difference for the second key pair for $K_1^l$, relatively to the correct difference for the first pair, the expected complexity will reduce by a factor of 2.

Our non-optimized implementation of the proposed attack in Python 2.7 on a standard PC required approximately 10 minutes to recover the correct key. We didn't observe multiple candidates for $K_1$.

The feasibility of the proposed attack is based mainly on the extremely simplified key schedule of a reduced version of the Kuznyechik, which allows

the attacker to trace the desired differences in round keys. The main problem of using related-key approach for the basic cipher is high diffusion properties of Kuznyechik round function.

## 5 CTRR encryption mode

In this Section we introduce a new internally re-keyed encryption mode CTRR which based on using two related keys. Also we present security bound for this mode in LOR-CPA notion.

### 5.1 Description

One of the important task of cryptographic protocol is to enforce the keys usage limitations. Such key usage limitation called «key lifetime». Informally but in some detail this issue is covered in [2, 3]. In [1] the main approaches of increasing the key lifetime were classified. One of those approaches, called «internal re-keying» was introduced in [19] and firstly investigated in [15, 14]. The main feature of this approach is that the key is transformed during the processing of each message according to some encryption mode.

The CPKM internal re-keying method introduced in [19] has the following disadvantage which became one of the reasons for developing a new method ACPKM within the framework of the work of Russian Technical committee for standardization (TC26): there may be a collision of block cipher permutation inputs in cases of key transformation and message processing. This issue solved by ACPKM by dividing the cipher permutation table into two parts — for data processing and for key updating. But such trick is possible only for modes for which we can predict inputs of encryption transformation, for example, for CTR mode. Other classic modes such as CBC or CFB don't have this predictability property. So this is not a universal solution to this problem. One of the universal solutions is to use so-called re-keying with master key described in [1], but using this leads to a noticeable complication of the scheme and need additional operations before data processing. Another universal solution is to use two related keys for processing blocks of different types (data or key).

Next, we describe new internally re-keyed encryption mode called CTRR which uses two related keys. Let $2|n$, $n|k$ and $C_1, \ldots, C_{k/n} \in V_n$ are pairwise different constant bit strings. Parameters of the mode is section size $N$ such that $n|N$ and transformation $\phi : V_k \to V_k$. The processing of the plaintext $P$ by means of the initialization vector $IV \in V_{n/2}$ and key $K \in V_k$ is carried out as follows:

1. $ctr_1 = IV||0^{n/2}$, $ctr_j = Inc(ctr_{j-1})$, $j = 2, \ldots, |P|_n$;
2. $K_1 = K$, $K_j = E_{\phi(K_{j-1})}(C_1)|| \ldots ||E_{\phi(K_{j-1})}(C_{k/n})$, $j = 2, \ldots, \lceil |P|/N \rceil$;
3. $G_j = E_{K_i}(ctr_j)$, $j = 1, \ldots, |P|_n$, $i = \lceil j \cdot n/N \rceil$;
4. Ciphertext $C$ is equal to $P \oplus \mathrm{msb}_{|P|}(G_1||...||G_{|P|_n})$.

Briefly stated, the plaintext is processed in the same way as in the standard CTR mode but the key that is used to generate the encryption blocks $G_j$ is not constant and is updated after generating $N/n$ blocks $G_j$. The $\phi$ transformation can be any of the following:

- $\phi(X) = X \oplus c$, for some constant $c \in V_k, c \neq o^k$;
- $\phi(X) = str_k(int(X) + c \ mod \ 2^k)$, for some constant $c \in \{1, \ldots, 2^k - 1\}$.

For the rest of the paper we will denote this set of possible variants for $\phi$ as $\Phi$.

### 5.2 Security estimation

Standard «single-key» security notion for block cipher is PRP-CPA («Pseudo Random Permutation in Chosen Plaintext Attack»). Adversary $\mathcal{A}$ has access to oracle $\mathcal{O}^{PRP}$. It takes blocks $M \in V_n$ as queries. When the first request is received it chooses $b \in_{\mathcal{U}} \{0, 1\}$ and if $b = 0$ oracle $\mathcal{O}^{PRP}$ chooses permutation $P \in_{\mathcal{U}} Perm(V_n)$, otherwise it chooses $K \in_{\mathcal{U}} V_k$. As a response to a query $M$ oracle returns a string $P(M)$, if $b = 0$, and a string $E_K(M)$, if $b = 1$. Adversary's advantage is calculated as

$$\mathrm{Adv}_{\mathrm{E}}^{\text{PRP-CPA}}(\mathcal{A}) = \mathbb{P}(\mathcal{A} \to 1 \mid b = 1) - \mathbb{P}(\mathcal{A} \to 1 \mid b = 0).$$

If effective specific attacks for a cipher $E$ are not known the value $\mathrm{InSec}_E^{\text{PRP-CPA}}(t, q)$ is estimated according to general attacks (i.e. attacks that do not use the cipher structure properties). For PRP-CPA it is assumed that the most effective general attack is a key recovery by exhaustive search over the keyspace, so $\mathrm{InSec}_E^{\text{PRP-CPA}}(t, q) \approx t/2^k$.

An analogue of PRP-CPA notion for several related keys is PRP-RKA («Pseudo Random Permutation in Related-Key Attack»). This model has an additional parameter, which is the set $\Phi$ of key-derivation functions $\phi : V_k \to V_k$. Adversary $\mathcal{A}$ has access to oracle $\mathcal{O}^{\text{RKA}}$ which takes pairs $(\phi, M)$, $\phi \in \Phi$ and $M \in V_n$, as queries. When the first request is received it chooses $b \in_{\mathcal{U}} \{0, 1\}$ and, if $b = 0$, it chooses $G \in_{\mathcal{U}} Perm(V_k, V_n)$ and a key $K \in_{\mathcal{U}} V_k$, and, if $b = 1$, the oracle chooses only a key $K \in_{\mathcal{U}} V_k$. Oracle answers query $(\phi, M)$ with a string $G_{\phi(K)}(M)$, if $b = 0$, and a string $E_{\phi(K)}(M)$, if $b = 1$. Adversary's advantage is calculated as

$$\mathrm{Adv}_{\Phi,\mathrm{E}}^{\text{PRP-RKA}}(\mathcal{A}) = \mathbb{P}(\mathcal{A} \to 1 \mid b = 1) - \mathbb{P}(\mathcal{A} \to 1 \mid b = 0).$$

In Section 4 attack on 4-round version of Kuznyechik with key schedule restricted to 2 rounds using $2^{10}$ related key pairs was presented. Growth of encryption round number or Feistel net round number during key schedule (i.e, making modification closer to real cipher) makes this attack inapplicable. Due to lack of successful attacks on the full (10-round) version of the cipher, we assume Kuznyechik to be resistant to related-key attacks.

In [17] the advantage of the adversary attacking an ideal cipher was bounded with properties of key-derivative function set, number of $\mathcal{O}^{RKA}$ calls and number $N_E$ of encryptions/decryptions on a key selected by adversary. Properties of key-derivative function set used in CTRR make advantage to be bounded by $\frac{N_E \cdot |\Phi|}{2^k}$. In considered case $|\Phi| = 2$ and $N_E \leqslant t$, so

$$\mathrm{InSec}^{\text{PRP-RKA}}_{\{id,\phi\},\mathrm{E}}(t,q) \leqslant \frac{t}{2^{k-1}}.$$

For Magma cipher $\mathrm{InSec}^{\text{PRP-RKA}}_{E}$ may eventually become high, since attack that recovers full key using 12 related keys is known [21]. Thereby using related keys in high-level protocols with this cipher should be considered insecure, and Magma should not be used in CTRR mode.

We stress out that PRP-RKA is a standalone task and cipher's insecurity in this model is in general independent of PRP-CPA. An example of a cipher which is PRP-CPA-secure and PRP-RKA-insecure is presented in Appendix C. Thereby, related-key cryptanalysis is necessary for complete security estimation and can not be replaced with analysis in «single-key» models.

The security analysis of the CTRR mode has been carried out in the IND-CPNA («Indistinguishability under Chosen Plaintext and Nonce Attack») model. This model is similar to the standard IND-CPA security model [4] but considers nonce-respecting adversaries [5]. Informally, in this model the adversary has to distinguish the obtained ciphertexts from the «garbage», having the capability to adaptively choose plaintexts and nonces (in a unique manner). The IND-CPNA is the strongest standard security model (known at the time) which allows to analyze the cryptographic properties of the mode from the viewpoint of computational «closeness» to the ideal one-time pad encryption [5].

**Theorem 1.** *Let $N \in \mathbb{N}, \phi \in \Phi$ be parameters of CTRR mode. Then for any adversary $\mathcal{A}$ with time complexity at most $t$ that makes $q$ queries, where the maximal message length is at most $m$ ($m \leqslant 2^{n/2-1}$) blocks and the total message length is at most $\sigma$ blocks, there exists an adversary $\mathcal{B}$ solving PRP-*

*RKA task such that*

$$\mathsf{Adv}^{\mathit{IND\text{-}CPNA}}_{\mathit{CTRRN},\phi}(\mathcal{A}) \leqslant l \cdot \mathsf{Adv}^{\mathit{PRP\text{-}RKA}}_{\{id,\phi\},E}(\mathcal{B}) + \frac{(\sigma_1+2)^2 + \ldots + (\sigma_{l-1}+2)^2 + (\sigma_l)^2}{2^n}$$

*where* $l = \lceil m/N \rceil$, $\sigma_j$ *is the total data block length processed under the section key* $K^j$ *and* $\sigma_j \leqslant 2^{n-1}$, $\sigma_1 + \ldots + \sigma_l = \sigma$. *The adversary* $\mathcal{B}$ *makes at most* $\sigma_1+2$ *queries. Furthermore, the time complexity of* $\mathcal{B}$ *is at most* $t+cn(\sigma+2l)$, *where* $c$ *is a constant that depends only on the model of computation and the method of encoding.*

The proof can be found in Appendix B.

## 6 Conclusion

This paper proposes a new internally re-keyed block cipher mode of operation called CTRR that use several keys with fixed bitwise sum. This mode could be used with a cipher, which is proven to be secure in the related-key model. We obtain security bounds for the mode under the assumption of underlying cipher security in the related-key model.

We provide a preliminary related-key cryptanalysis of reduced Kuznyechik block cipher, which provides ground for use of full version of the cipher with CTRR.

## References


[1] Smyshlyaev S. (ed.), Housley R., Alekseev E., Smyshlyaeva E., Gueron S., Franke D.F., Ahmetzyanova L.: *Re-keying Mechanisms for Symmetric Keys*. Network Working Group, draft-irtf-cfrg-re-keying-11.

[2] Alekseev E.K., Ahmetzyanova L.R., Meshkov D.A., Smyshlyaev S.V., Smyshlyaeva E.S.: *On key lifetime. Part 1.* CryptoPro LLC Blog, 2017. (In Russian), `http://cryptopro.ru/blog/2017/05/17/o-nagruzke-na-klyuch-chast-1`.

[3] Alekseev E.K., Ahmetzyanova L.R., Meshkov D.A., Smyshlyaev S.V., Smyshlyaeva E.S.: *On key lifetime. Part 2.* CryptoPro LLC Blog, 2017. (In Russian), `http://cryptopro.ru/blog/2017/05/29/o-nagruzke-na-klyuch-chast-2`.

[4] Bellare M., Rogaway P.: *Introduction to modern cryptography: Lecture Notes.* - `http://www.cs.ucsd.edu/users/mihir/cse207/classnotes.html`. 2001.

[5] Rogaway P.: *Nonce-Based Symmetric Encryption // Fast Software Encryption* / Ed. by Bimal Roy, Willi Meier. – Berlin, Heidelberg : Springer Berlin Heidelberg, 2004. – P. 348-358.

[6] Dunkelman O., Keller N., Shamir A.: *A Practical-Time Attack on the A5/3 Cryptosystem Used in Third Generation GSM Telephony.* In Cryptology ePrint Archive, Report 2010/013, 2010.

[7] Preneel B.: *Hash Functions and MAC Algorithms Based on Block Ciphers.* Proc. of IMA'97, LNCS 1355, Springer, pp. 270–282 (1997).

[8] Kelsey J., Schneier B., Wagner D.: *Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES.* Advances in Cryptology - Crypto'96, LNCS 1109, Springer, pp. 237–251 (1996).

[9] Kelsey J., Schneier B., Wagner D.: *Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA.* Proc. of ICICS'97, LNCS 1334, Springer, pp. 233–246 (1997).

[10] Biham E., Shamir A.: *Differential Cryptanalysis of DES-like Cryptosystems.* Journal of Cryptology, vol. 4, pp. 3-72, IACR, 1991.

[11] Wagner D.: *The Boomerang Attack.* In: Knudsen L. (eds) Fast Software Encryption. FSE 1999. Lecture Notes in Computer Science, vol 1636. Springer, Berlin, Heidelberg.

[12] Lee J.-K., Koo B., Kim W.-H.: *Related-Key Linear Cryptanalysis on SIMON.* Cryptology ePrint Archive: Report 2018/152. 2018.

[13] Bellare M.: *Practice-Oriented Provable-Security //* Lectures on Data Security: Modern Cryptology in Theory and Practice / Ed. by Ivan B. Damgard.—Berlin, Heidelberg : Springer Berlin Heidelberg, 1999. – P. 1–15. – ISBN: 978-3-540-48969-6.

[14] Ahmetzyanova L.R., Alekseev E.K., Oshkin I.B., Smyshlyaev S.V., Sonina L.A.: *On the properties of the CTR encryption mode of Magma and Kuznyechik block ciphers with re-keying method based on CryptoPro Key Meshing.* Mat. Vopr. Kriptogr., 2017, Volume 8, Issue 2, Pages 39–50, 2017.

[15] Ahmetzyanova L.R., Alekseev E.K., Oshkin I.B., Smyshlyaev S.V.: *Increasing the Lifetime of Symmetric Keys for the GCM Mode by Internal Re-keying.* Cryptology ePrint Archive: Report 2017/697.

[16] Bellare M., Desai A., Jokipii E., Rogaway P.: *A Concrete Security Treatment of Symmetric Encryption*. In Proceedings of 38th Annual Symposium on Foundations of Computer Science (FOCS '97), pages 394–403. IEEE, 1997.

[17] Bellare M., Kohno T.: *A Theoretical Treatment of Related-Key Attacks: RKA-PRPs, RKA-PRFs, and Applications*, EUROCRYPT 2003: Advances in Cryptology — EUROCRYPT 2003 pp 491-506, 2003.

[18] *Information technology. Cryptographic Data Security. Block ciphers.* GOST R 34.12-2015, Federal Agency on Technical Regulating and Metrology, 2015.

[19] Popov V., Kurepkin I., Leontiev S.: *Additional cryptographic algorithms for use with GOST 28147-89, GOST R 34.10-94, GOST R 34.10-2001, and GOST R 34.11-94 algorithms*. RFC 4357, 2007.

[20] Biryukov A., Khovratovich D.: *Related-key Cryptanalysis of the Full AES-192 and AES-256*. ASIACRYPT 2009: Advances in Cryptology – ASIACRYPT 2009 pp 1-18, 2009.

[21] Pudovkina M. A, Khoruzhenko G. I.: *An attack on the GOST 28147-89 block cipher with 12 related keys*. Mat. Vopr. Kriptogr., 2013, Volume 4, Issue 2, Pages 127–152, 2013.

[22] Fluhrer S., Mantin I., Shamir A.: *Weaknesses in the Key Scheduling Algorithm of RC4*. Selected Areas of Cryptography: SAC 2001, Lecture Notes in Computer Science Vol. 2259, pp 1-24, 2001.

[23] Canteaut A., Lallemand V., Naya-Plasencia M.: *Related-Key Attack on Full-Round PICARO*. Cryptology ePrint Archive: Report 2015/754, 2015.

[24] Lu J., Yap W., Wei Y.: *Weak Keys of the Full MISTY1 Block Cipher for Related-Key Cryptanalysis*. Cryptology ePrint Archive: Report 2012/066, 2012.

[25] Fleischmann E., Gorski M, Lucks S.: *Related-Key Rectangle Attack of the Full 80-Round HAS-160 Encryption Mode*. Cryptology ePrint Archive: Report 2009/335, 2009.

[26] Knudsen L.R.: *Cryptanalysis of LOKI*. In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, ASIACRYPT'91, volume 739

of LNCS, pages 22–35, Fujiyoshida, Japan, November 11–14, 1991. Springer, Berlin, Germany.

[27] Biham E.: *New types of cryptoanalytic attacks using related keys (extended abstract)*. In Tor Helleseth, editor, EUROCRYPT'93, volume 765 of LNCS, pages 398–409, Lofthus, Norway, May 23–27, 1993. Springer, Berlin, Germany.

[28] Winternitz R.S., Hellman M.E.: *Chosen-key Attacks on a Block Cipher*. In Cryptologia 11, 1, pp. 16–20, 1987.

[29] Razali E., Phan R.C.-W.: *On The Existence of Related-Key Oracles in Cryptosystems based on Block Ciphers*. In Proceedings of International Workshop on Information Security (IS '06), in On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops (2006), R. Meersman, Z. Tari, and P. Herrero, Eds., LNCS 4277, Springer, pp. 425–438.

## Appendix

## A Description of the proposed key recovery attack

### A.1 Key recovery algorithm for $K_1^l, l = \overline{0,15}$:

1. Choose plaintext $m = C_1$
2. For each $k = \overline{0,255}$:
   (a) Choose 2 pairs of related keys:

$$(K_1, K_2), (K'_{1,l}, K''_{j,l}) = (K_1 \oplus 1 \lll 8l, K_2 \oplus L_l(j)),$$

   where $j$ such, that $\pi(k \oplus C_1^l) \oplus \pi(k \oplus C_1^l \oplus 1) = j$ and

$$(K_1, K_2), (K'_{6,l}, K''_{\hat{j},l} = (K_1 \oplus 6 \lll 8l, K_2 \oplus L_l(\hat{j}))),$$

   where $\hat{j}$ such, that $\pi(k \oplus C_1^l) \oplus \pi(k \oplus C_1^l \oplus 6) = \hat{j}$
   (b) If $L^{-1}(\hat{E}_{K_1,K_2}(m) \oplus \hat{E}_{K'_{1,l},K''_{j,l}}(m)) = (0, \ldots, 0, \underbrace{a}_{l}, 0, \ldots, 0)$ and $L^{-1}(\hat{E}_{K_1,K_2}(m) \oplus \hat{E}_{K'_{6,l},K''_{\hat{j},l}}(m)) = (0, \ldots, 0, \underbrace{b}_{l}, 0, \ldots, 0)$ for some $a, b \in V_8$, than store in memory $k$. If we have several $k$, which are satisfied the conditions, store them all in a possible values array.
3. Set $K_1^l$ to $k$ (if we have several possible values – make separate copies of the key).

After 16 applications of this algorithm (for all $l = 0, \ldots, 15$), we could have all $K_1$ (or an array of possible $K_1$, with the correct key among them).

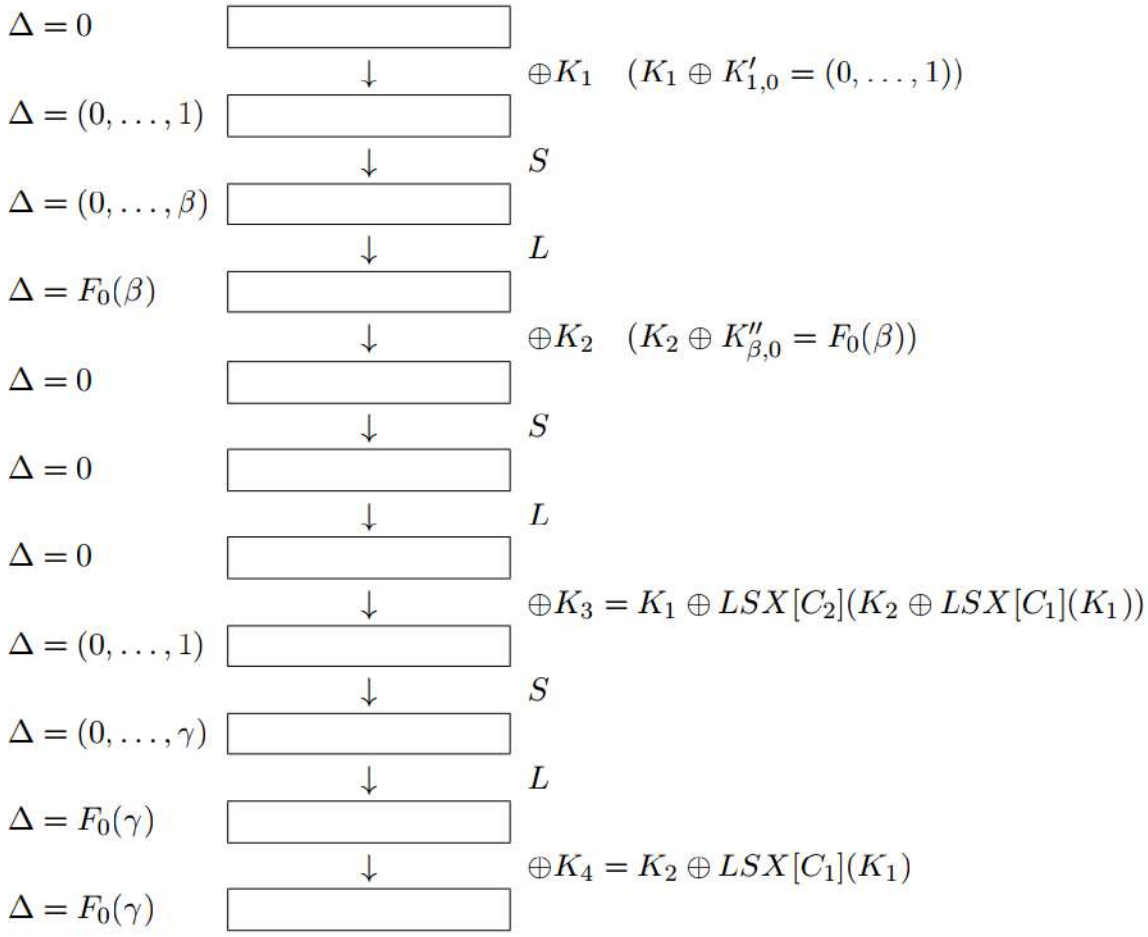



Figure 1: The propagation of difference $\Delta$ through the encryption rounds for the first attack phase.

We could evaluate $K_2$ with the help of $K_1$ by the following equation:

### A.2 Recovery for $K_2$ when $K_1$: is known

1. Select plaintext $m = (C_1^{15}, C_1^{14}, \ldots, C_1^2, C_1^1, \pi^{-1}(\pi(C_1^0 \oplus K_1^0) \oplus 2) \oplus K_1^0)$
2. $K_2 = \hat{E}_{K_1,K_2}(m) \oplus LSX[C_1](K_1) \oplus LS(K_1)$

Figure 2 shows the encryption of the message $m = (C_1^{15}, C_1^{14}, \ldots, C_1^2, C_1^1, \pi^{-1}(\pi(C_1^0 \oplus K_1^0) \oplus 2) \oplus K_1^0)$:

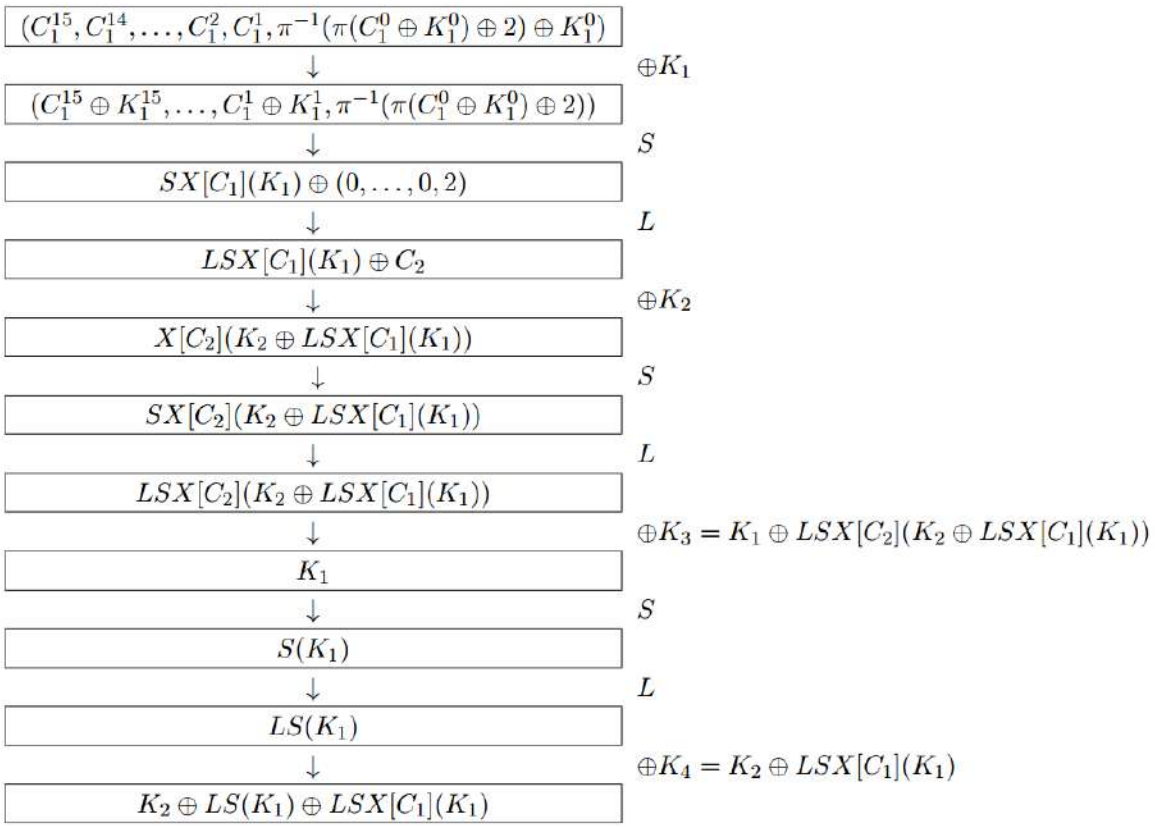



Figure 2:

## B Security bound for CTRR

Here we prove the theorem 1:

**Theorem 2.** *Let $N \in \mathbb{N}, \phi \in \Phi$ be parameters of CTRR mode. Then for any adversary $\mathcal{A}$ with time complexity at most $t$ that makes $q$ queries, where the maximal message length is at most $m$ ($m \leqslant 2^{n/2-1}$) blocks and the total message length is at most $\sigma$ blocks, there exists an adversary $\mathcal{B}$ solving PRP-RKA task such that*

$$\mathsf{Adv}^{\textit{IND-CPNA}}_{\textit{CTRRN},\phi}(\mathcal{A}) \leqslant l \cdot \mathsf{Adv}^{\textit{PRP-RKA}}_{\{id,\phi\},E}(\mathcal{B}) + \frac{(\sigma_1+2)^2 + \ldots + (\sigma_{l-1}+2)^2 + (\sigma_l)^2}{2^{n+1}}$$

*where $l = \lceil m/N \rceil$, $\sigma_j$ is the total data block length processed under the section key $K^j$ and $\sigma_j \leqslant 2^{n-1}$, $\sigma_1 + \ldots + \sigma_l = \sigma$. The adversary $\mathcal{B}$ makes at most $\sigma_1+2$ queries. Furthermore, the time complexity of $\mathcal{B}$ is at most $t+cn(\sigma+2l)$, where $c$ is a constant that depends only on the model of computation and the method of encoding.*

*Proof.* Define hybrid experiments $Hybrid_j(\mathcal{A})$, $j = 0, 1, \ldots, \lceil m/N \rceil$. In the experiment $Hybrid_j(\mathcal{A})$ the oracle in the IND-CPNA notion is replaced by

the oracle, which operates in the following way:

– The oracle chooses key $K^{j+1} \in_{\mathcal{U}} V_k$;

– In response to a query $(P, IV)$ the oracle returns $C$, where

$$C = M \oplus \mathrm{msb}_{|P|}(C' \| C^{j+1} \| \ldots \| C^{\lceil m/N \rceil}),$$

here $C' \in_{\mathcal{U}} V_{nNj}$ and $C^i,\ i = (j+1), \ldots, \lceil m/N \rceil$, is the result of the $i$-th section processing under the $K^i$ section key. Note that the $(j+1)$-th section is processed under the «truly» random $K^{j+1}$ key and each next key is produced according to $RKR$ key derivation algorithm.

The result of any experiment described above is what the adversary $\mathcal{A}$ returns as a result.

Note that the $Hybrid_0(\mathcal{A})$ experiment totally coincides with the $\mathbf{Exp}^{\text{IND-CPNA}-1}_{\text{CTRR}_N}(\mathcal{A})$ experiment, and the experiment $Hybrid_{\lceil m/N \rceil}(\mathcal{A})$ coincides with $\mathbf{Exp}^{\text{IND-CPNA}-0}_{\text{CTRR}_N}(\mathcal{A})$ experiment, i.e. the following equalities hold:

$$\mathbb{P}\left(Hybrid_0(\mathcal{A}) \to 1\right) = \mathbb{P}\left(\mathbf{Exp}^{\text{IND-CPNA}}_{\text{CTRR}_N}(\mathcal{A}) \to 1 \,|\, b = 1\right),$$

$$\mathbb{P}\left(Hybrid_{\lceil m/N \rceil}(\mathcal{A}) \to 1\right) = \mathbb{P}\left(\mathbf{Exp}^{\text{IND-CPNA}}_{\text{CTRR}_N}(\mathcal{A}) \to 1 \,|\, b = 0\right).$$

Construct a set of adversaries $\mathcal{B}_j$, $j = 1, \ldots, \lceil m/N \rceil$, for the block cipher in the PRF-RKA model, which uses $\mathcal{A}$ as a black box.

After receiving a query $(P, IV)$ from $\mathcal{A}$ the adversary $\mathcal{B}_j$ processes this query as in the $Hybrid_j(\mathcal{A})$ experiment but the encrypted blocks for masking the $j$-th section and blocks of the $(j+1)$-th section key are obtained by making queries to the oracle provided by the PRF experiment. Note that $\mathcal{B}_j$, $j = 1, \ldots, \lceil m/N \rceil - 1$, makes at most $\sigma_j + s$ queries and $\mathcal{B}_{\lceil m/N \rceil}$ makes at most $\sigma_{\lceil m/N \rceil}$ queries. The adversary $\mathcal{B}_j$ returns 1, if the adversary $\mathcal{A}$ returns 1, and returns 0, otherwise.

Note that

$$\mathbb{P}\left(\mathbf{Exp}^{\text{PRF-RKA}}_{\{id,\phi\},\,E}(\mathcal{B}_j) \to 1 \,|\, b = 1\right) = \mathbb{P}\left(Hybrid_{j-1}(\mathcal{A}) \to 1\right),$$

$$\mathbb{P}\left(\mathbf{Exp}^{\text{PRF-RKA}}_{\{id,\phi\},\,E}(\mathcal{B}_j) \to 1 \,|\, b = 0\right) = \mathbb{P}\left(Hybrid_j(\mathcal{A}) \to 1\right).$$

The last equality is proceeded from the fact that for the random function input blocks for producing the $K^{j+1}$ section key and the input blocks for masking the $j$-th section are processed with different keys combined with the fact that inside these groups blocks are also different. Therefore, the $K^{j+1}$ variable distribution is statistically indistinguishable from the «truly» random one.

Then for the advantages of the adversaries $B_j$ it holds that

$$\sum_{j=1}^{\lceil m/N \rceil} \mathsf{Adv}^{\text{PRF-RKA}}_{\{id,\phi\},E}(B_j) =$$

$$= \sum_{j=1}^{\lceil m/N \rceil} \mathbb{P}\left(Hybrid_{j-1}(\mathcal{A}) \to 1\right) - \sum_{j=1}^{\lceil m/N \rceil} \mathbb{P}\left(Hybrid_{j}(\mathcal{A}) \to 1\right) =$$

$$= \mathbb{P}\left(Hybrid_{0}(\mathcal{A}) \to 1\right) - \mathbb{P}\left(Hybrid_{\lceil m/N \rceil}(\mathcal{A}) \to 1\right) = \mathsf{Adv}^{\text{IND-CPNA}}_{\text{CTRR}_{N,\phi}}(\mathcal{A}).$$

From the PRP/PRF switching lemma [17] we have that for $\phi \in \Phi$ for any block cipher $E$ and any adversary $\mathcal{B}'$ making at most $q$ queries

$$\mathsf{Adv}^{\text{PRF-RKA}}_{\{id,\phi\},E}(\mathcal{B}') \leqslant \mathsf{Adv}^{\text{PRP-RKA}}_{\{id,\phi\},E}(\mathcal{B}') + \frac{2q(q-1)}{2^{n+1}} \leqslant \mathsf{Adv}^{\text{PRP-RKA}}_{\{id,\phi\},E}(\mathcal{B}') + \frac{q^2}{2^n}.$$

Thus,

$$\mathsf{Adv}^{\text{IND-CPNA}}_{\text{CTRR}_N}(\mathcal{A}) = \sum_{j=1}^{\lceil m/N \rceil} \mathsf{Adv}^{\text{PRF-RKA}}_{\{id,\phi\},E}(\mathcal{B}_j) \leqslant$$

$$\leqslant \sum_{j=1}^{\lceil m/N \rceil - 1} \left( \mathsf{Adv}^{\text{PRP-RKA}}_{\{id,\phi\},E}(\mathcal{B}_j) + \frac{(\sigma_i + s)^2}{2^n} \right) +$$

$$+ \mathsf{Adv}^{\text{PRP-RKA}}_{\{id,\phi\},E}(\mathcal{B}_{\lceil m/N \rceil}) + \frac{\left(\sigma_{\lceil m/N \rceil}\right)^2}{2^n} \leqslant$$

$$\leqslant \left\lceil \frac{m}{N} \right\rceil \mathsf{Adv}^{\text{PRP-RKA}}_{\{id,\phi\},E}(\mathcal{B}) + \frac{(\sigma_1 + s)^2 + \ldots + (\sigma_{\lceil m/N \rceil - 1} + s)^2 + \left(\sigma_{\lceil m/N \rceil}\right)^2}{2^n},$$

where $\mathcal{B}$ is an adversary which makes at most $\sigma_1 + s$ queries. The last relation is due to $\sigma_1 \geqslant \ldots \geqslant \sigma_{\lceil m/N \rceil}$ and $\mathsf{Adv}^{\text{PRP-RKA}}_{\{id,\phi\},E}(\mathcal{B}') \leqslant \mathsf{Adv}^{\text{PRP-RKA}}_{\{id,\phi\},E}(\mathcal{B}'')$ for such adversaries $\mathcal{B}'$ and $\mathcal{B}''$ with the same computational resources that the queries number made by $\mathcal{B}'$ is less than the queries number made by $\mathcal{B}''$. □

## C Irreducibility of PRP-RKA to PRP-CCA

We stress out that PRP-RKA can not be effectively reduced to PRP-CPA. The following example proves it:

**Theorem 3.** *There exists such cipher $E$ that*

$$InSec_{E}^{PRP\text{-}CPA}(t, q_1, q_2) \approx \frac{t}{2^k};$$
$$InSec_{\{id, x\oplus 1||0^k\}, E}^{PRP\text{-}RKA}(t, 0, 2) \approx 1.$$

*Proof.* Let $E' : V_k \times V_n \to V_n$ be an ideal cipher (for ideal cipher the first estimate considered to be true). Now we construct a new cipher $E : V_{k+1} \times V_n \to V_n, E_{1||K}(m) = E_{0||K}(m) = E_K(m) \forall K \in V_k$. Let $\mathcal{A}$ be adversary attacking $E$ in PRP-CPA model. Let's construct an adversary $B$, attacking $E'$ in PRP-CPA model using $\mathcal{A}$ as a black box: all $\mathcal{A}'s$ queries $\mathcal{B}$ redirects to his own oracle and redirects oracle's answers to $\mathcal{A}$ and in the end of experiment $\mathcal{B}$ puts the same value as $\mathcal{A}$ does. Note that due to the structure of cipher $E$, situation when $\mathcal{B}$ chooses key $K = K_0||K_1||\ldots||K_k$ and when $\mathcal{B}$ chooses key $K' = K_1||\ldots||K_k$ are observed absolutely equally from $\mathcal{A}'s$ perspective, so output distribution in both experiments will be the same. Thus,

$$\mathrm{Adv}_{\mathrm{E}}^{\text{PRP-CPA}}(B) =$$
$$= \mathbb{P}(K \in_{\mathcal{U}} V_{k+1} : B \to 1 \mid b = 1) - \mathbb{P}(K \in_{\mathcal{U}} V_{k+1} : B \to 1 \mid b = 0) =$$
$$= \frac{1}{2} \cdot \mathbb{P}(K \in_{\mathcal{U}} V_{k+1} : B \to 1 \mid b = 1, K_0 = 0) +$$
$$+ \frac{1}{2} \cdot \mathbb{P}(K \in_{\mathcal{U}} V_{k+1} : B \to 1 \mid b = 1, K_0 = 1) -$$
$$- \frac{1}{2} \cdot \mathbb{P}(K \in_{\mathcal{U}} V_{k+1} : B \to 1 \mid b = 0, K_0 = 0) -$$
$$- \frac{1}{2} \cdot \mathbb{P}(K \in_{\mathcal{U}} V_{k+1} : B \to 1 \mid b = 0, K_0 = 1) =$$
$$= \frac{1}{2} \cdot \mathbb{P}(K \in_{\mathcal{U}} V_{k+1}, K' = K_1||\ldots||K_k : A \to 1 \mid b = 1, K_0 = 0) +$$
$$+ \frac{1}{2} \cdot \mathbb{P}(K \in_{\mathcal{U}} V_{k+1}, K' = K_1||\ldots||K_k : A \to 1 \mid b = 1, K_0 = 1) -$$
$$- \frac{1}{2} \cdot \mathbb{P}(K \in_{\mathcal{U}} V_{k+1}, K' = K_1||\ldots||K_k : A \to 1 \mid b = 0, K_0 = 0) -$$
$$- \frac{1}{2} \cdot \mathbb{P}(K \in_{\mathcal{U}} V_{k+1}, K' = K_1||\ldots||K_k : A \to 1 \mid b = 0, K_0 = 1) =$$
$$= \frac{1}{2} \cdot \mathbb{P}(K' \in_{\mathcal{U}} V_k : A \to 1 \mid b = 1) + \frac{1}{2} \cdot \mathbb{P}(K' \in_{\mathcal{U}} V_k : A \to 1 \mid b = 1) -$$
$$- \frac{1}{2} \cdot \mathbb{P}(K' \in_{\mathcal{U}} V_k : A \to 1 \mid b = 0) - \frac{1}{2} \cdot \mathbb{P}(K' \in_{\mathcal{U}} V_k : A \to 1 \mid b = 0) =$$
$$= \mathrm{Adv}_{\mathrm{E'}}^{\text{PRP-CPA}}(A)$$

Note that there exists adversary $\mathcal{C}$, that successfully attacks $E$ in PRP-RKA model using set of related-key derivating functions $\Phi = \{id, x \oplus 1||0^k\}$ ($id$ is identical function) with only 2 oracle queries:

1. Choose $m \in_{\mathcal{U}} V_n$;
2. Send query $(id, m)$ to $\mathcal{O}^{\text{RKA}}$ and store answer as $R_1$;
3. Send query $(x \oplus 1||0^k, m)$ to $\mathcal{O}^{\text{RKA}}$ and store answer as $R_2$;
4. If $R_1 = R_2$ return 1, otherwise return 0.

Note that

$$\text{Adv}^{\text{PRP-RKA}}_{\Phi,\text{E}}(A) = \mathbb{P}(A \to 1 \mid b = 1) - \mathbb{P}(A \to 1 \mid b = 0) =$$
$$= \mathbb{P}(R_1 = R_2 \mid b = 1) - \mathbb{P}(R_1 = R_2 \mid b = 0) = 1 - \frac{1}{2^n}.$$

Thereby $E$ preserves PRP-CCA-robustness of $E'$, but $\text{InSec}^{\text{PRP-RKA}}_{\{id, x\oplus 1||0^k\}, E} \approx 1$, i.e. $E$ is completely PRP-RKA-insecure. $\square$

# Algebraic aspects

# Some Properties of Modular Addition

Victoria Vysotskaya

JSC «InfoTeCS», Moscow, Russia
vysotskaya.victory@gmail.com

**Abstract**

In this paper we study a problem which emerged during an attempt to apply a differential cryptanalysis method to the «Magma» algorithm. We evaluate the number of distinct distributions in the difference distribution table and give an asymptotically accurate estimation of it. Moreover, we provide an algorithm for generation of all the distinct distributions in $2^{O(\sqrt{n})}$ operations and counting them in polynomial time instead of $2^{\Omega(n)}$ in the brute-force method.

**Keywords:** modular addition, partitions, differential cryptanalysis.

## 1 Introduction

The problem studied in the paper emerged during an attempt to estimate the applicability of differential cryptanalysis to the Russian government standard symmetric key block cipher (GOST 28147-89) rounds [1]. It is vital since the algorithm (called «Magma») is still present in the modern Russian GOST R 34.12-2015 describing symmetric key block ciphers [2].

During the research on the topic the following equation emerged:

$$\Delta f = \big[(x \oplus \Delta x) \boxplus_n y\big] \oplus (x \boxplus_n y). \tag{1}$$

It is a special case of the difference of addition modulo $2^n$ studied in [3].

Let us introduce the function $P_n(\Delta x, \Delta f)$:

$$P_n(\Delta x, \Delta f) = \Big| \{(x, y) : \Delta f = \big[(x \oplus \Delta x) \boxplus_n y\big] \oplus (x \boxplus_n y);\ \Delta x, \Delta f \in \{0, \ldots, 2^n - 1\}\} \Big|,$$

and let us consider the table of values of this function $(P_n)_{\Delta x, \Delta f}$. In this table rows are indexed by $\Delta x$ and columns by $\Delta f$. Such a table is usually called difference distribution table (DDT).

Let us introduce an equivalence relation on the rows of matrix $P_n$ as follows: two rows are called *equivalent* if they coincide up to permutations of elements. Next, we study the set of equivalence classes into which matrix rows are divided. Let us call such equivalence classes as *distributions*.

**Note.** *Let us consider the calculation of number of different distributions or enumerating them, as an algorithmic tasks. Then trivial (brute force) algorithm requires* $2^{\Omega(n)}$ *operations as one needs to calculate the value of* $\Delta f$ *for all* $x$, $y$, $\Delta x \in \{0, \ldots, 2^n - 1\}$. *At the same time the algorithm based on the results presented in our article requires a polynomial number of operations for the first task and* $2^{O(\sqrt{n})}$ *operations for the second.*

## 2 Parametrization of distributions

**Lemma 1.** *Let matrix* $P_n$ *has the form*

$$P_n = \left[\begin{array}{cc} A & B \\ C & D \end{array}\right].$$

*Then matrix* $P_{n+1}$ *has the form*

$$P_{n+1} = 2\left[\begin{array}{cc|cc} 2A & B & 0 & B \\ C & D & C & D \\ \hline 0 & B & 2A & B \\ C & D & C & D \end{array}\right].$$

The proof of Lemma 1 is given in the Appendix A, since it is quite cumbersome.

This Lemma can be reworded: if

$$P_n = 2^{n+1}\left[\begin{array}{cc} A_n & B_n \\ B_n & A_n \end{array}\right],$$

then

$$A_n = \left[\begin{array}{cc} 2A_{n-1} & B_{n-1} \\ B_{n-1} & A_{n-1} \end{array}\right], \quad B_n = \left[\begin{array}{cc} 0 & B_{n-1} \\ B_{n-1} & A_{n-1} \end{array}\right].$$

Let us denote by $(\alpha_{n-1}, \alpha_{n-2}, \ldots, \alpha_1, \alpha_0)$ the binary representation of number $i$. Then let us match each distribution located in some row of matrix $P_n$ with a polynomial in the following way. A row $p_i$ corresponds to polynomial $\sum_{j=0}^{n+2} c_j x^j$, where $c_i$ is the amount of numbers $2^i$ in $p_i$. Hence multiplication by 2 corresponds to multiplication by $x$ and concatenation to addition of polynomials. For $a_n^i(x)$ and $b_n^i(x)$ corresponding to $i$-th rows of $A_n$ and $B_n$ respectively we have:

$$a_n^i(x) = \begin{cases} x a_{n-1}^i(x) + b_{n-1}^i(x), & \text{if } \alpha_{n-2} = 0, \\ a_{n-1}^i(x) + b_{n-1}^i(x), & \text{if } \alpha_{n-2} = 1; \end{cases}$$

$$b_n^i(x) = \begin{cases} b_{n-1}^i(x), \text{ if } \alpha_{n-2} = 0, \\ a_{n-1}^i(x) + b_{n-1}^i(x), \text{ if } \alpha_{n-2} = 1. \end{cases}$$

Thus,

$$\begin{bmatrix} a_n^i(x) \\ b_n^i(x) \end{bmatrix} = W_{\alpha_{n-2}} \begin{bmatrix} a_{n-1}^i(x) \\ b_{n-1}^i(x) \end{bmatrix},$$

where

$$W_0 = \begin{bmatrix} x & 1 \\ 0 & 1 \end{bmatrix}, \quad W_1 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

Moreover,

$$A_1 = \begin{bmatrix} 1 \end{bmatrix}, \ B_1 = \begin{bmatrix} 0 \end{bmatrix}, \ a_1 = 1 \ b_1 = 0.$$

Repeating the same argument $n-2$ more times we finally get

$$a_n^i(x) + b_n^i(x) = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} a_n^i(x) \\ b_n^i(x) \end{bmatrix} = \\ = \begin{bmatrix} 1 & 1 \end{bmatrix} W_{\alpha_{n-2}} W_{\alpha_{n-3}} \ \dots \ W_{\alpha_0} \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \quad (2)$$

Let us denote by $i'$ the number with binary representation $(\alpha_{n-2}, \alpha_{n-3}, \dots, \alpha_0)$. This choice is based on the knowledge that the most significant bit does not affect the distribution. Let us separate groups of 0's and 1's in $i'$. We assume that the first one is a group of 1's, and the last one is a group of 0's (both can be empty). The number of 1's is $K = k_1 + k_2 + \cdots + k_s$, the number of 0's is $L = \ell_1 + \cdots + \ell_s$ and $L + K = n - 1$. Then

$$i' = \underbrace{11\dots1}_{k_1}\underbrace{0\dots0}_{\ell_1}\underbrace{1\dots1}_{k_2}\underbrace{0\dots0}_{\ell_2}\dots\underbrace{1\dots1}_{k_s}\underbrace{0\dots0}_{\ell_s}$$

and expression (2) becomes

$$a_n^i(x) + b_n^i(x) = \begin{bmatrix} 1 & 1 \end{bmatrix} W_1^{k_1} W_0^{\ell_1} \dots W_1^{k_s} W_0^{\ell_s} \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \quad (3)$$

We will use the following statements, easily provable by induction:

$$W_1^k = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = 2^{k-1} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},$$

$$W_0^\ell = \begin{bmatrix} x & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x & 1 \\ 0 & 1 \end{bmatrix} \cdots \begin{bmatrix} x & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} x^\ell & x^{\ell-1} + x^{\ell-2} + \cdots + 1 \\ 0 & 1 \end{bmatrix}.$$

Then (3) may be represented as:

$$a_n^i(x) + b_n^i(x) =$$
$$= \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} 2^{k_1-1} \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x^{\ell_1} & x^{\ell_1-1} + \cdots + 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \cdots$$
$$\cdots \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x^{\ell_s} & x^{\ell_s-1} + \cdots + 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Note that

$$\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x^{\ell} & x^{\ell-1} + \cdots + 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = x^{\ell} + x^{\ell-1} + x^{\ell-2} + \cdots + 2.$$

Then

$$a_n^i(x) + b_n^i(x) = 2 \cdot 2^{K-s}(x^{\ell_1} + x^{\ell_1-1} + \cdots + 2) \ \ldots \ (x^{\ell_s} + x^{\ell_s-1} + \cdots + 2)x^{\ell_s}.$$

Hence

$$p_n^i(x) = 2^{K-s+1} \prod_{j=1}^{s-1} (x^{\ell_j} + x^{\ell_j-1} + \cdots + 2)x^{\ell_s}. \tag{4}$$

Let us denote by $Q_n$ the set of tuples $(s, L, \ell_s, \widetilde{\ell})$, where $s \in \{1, \ldots, n-1\}$, $\ell_s \in \{0, \ldots, n-1\}$, $L \in \{0, \ldots, n-s\}$ and $\widetilde{\ell}$ is a multiset of $s-1$ positive integers summing up to $L - \ell_s$. We now want to prove that there is a one-to-one correspondence between the set of polynomials $p_n^i(x)$ and the set $Q_n$. It is obvious that there is a corresponding set $q_i \in Q_n$ to each polynomial $p_n^i(x)$ and vice versa. So it is enough to show that if two polynomials are equal then corresponding sets of parameters coincide.

Let us fix numbers $d_1$ and $d_2$ and then compare two expressions

$$p_n^{d_1}(x) = 2^{K'-s'+1} \prod_{j=1}^{s'-1} (x^{\ell'_j} + x^{\ell'_j-1} + \cdots + 2)x^{\ell'_{s'}},$$

$$p_n^{d_2}(x) = 2^{K''-s''+1} \prod_{j=1}^{s''-1} (x^{\ell''_j} + x^{\ell''_j-1} + \cdots + 2)x^{\ell''_{s''}}.$$

If polynomials are equal, then $2^{K'-s'+1}x^{L'} = 2^{K''-s''+1}x^{L''}$, hence $L' = L''$ and $K' - s' + 1 = = K'' - s'' + 1$. Since the counts of 0's are equal, the counts of 1's are equal too, so $s' = s''$. Besides, the lower powers of the polynomials must coincide, hence $\ell'_{s'} = \ell''_{s''}$. Now it remains to prove that under stated assumptions the equality of polynomials also mean the equality of parameters

$\ell'_1, \ldots, \ell'_{s'}$ and $\ell''_1, \ldots, \ell''_{s''}$ up to a permutation.

Let us denote

$$\mathcal{G}_{\ell'_j}(x) = x^{\ell'_j} + x^{\ell'_j - 1} + \cdots + 2,$$
$$\mathcal{G}_{\ell''_j}(x) = x^{\ell''_j} + x^{\ell''_j - 1} + \cdots + 2.$$

We now show that if

$$\prod_{j'=1}^{s'-1} \mathcal{G}_{\ell'_{j'}}(x) = \prod_{j''=1}^{s''-1} \mathcal{G}_{\ell''_{j''}}(x), \tag{5}$$

then the multiset $\{\mathcal{G}_{\ell'_{j'}}(x)\}_{j'=1}^{s'-1}$ equals to the multiset $\{\mathcal{G}_{\ell''_{j''}}(x)\}_{j''=1}^{s''-1}$, or in other words the decomposition of such polynomials into factors of form $\mathcal{G}_j(x)$ is unique. For this purpose we prove that polynomials $\mathcal{G}_j(x)$ are pairwise coprime. Let us compute the greatest common divisor of $\mathcal{G}_u(x)$ and $\mathcal{G}_v(x)$ for $u > v$:

$$\begin{gathered}
(\mathcal{G}_u(x), \mathcal{G}_v(x)) = \\
= (x^u + x^{u-1} + \cdots + 2, x^v + x^{v-1} + \cdots + 2) = \\
= (x^{u-v-1} + \cdots + 1, x^v + x^{v-1} + \cdots + 2) = \\
= \left( \frac{x^{u-v} - 1}{x - 1}, \frac{x^{v+1} - 1}{x - 1} + 1 \right) = \\
= \frac{1}{x-1} \left( x^{u-v} - 1, x^v + x - 2 \right).
\end{gathered}$$

The roots of the polynomial $f(x) = x^{u-v} - 1$ are all roots of unity of the degree $(u - v)$. Let us check which of these roots can be the roots of the polynomial $h(x) = x^v + x - 2$.

Let $\varepsilon = \cos \frac{2\pi}{u-v} + i \cdot \sin \frac{2\pi}{u-v}$ be a primitive root of unity of the degree $(u - v)$, then $\{\varepsilon^\ell\}_{\ell=0}^{u-v-1}$ is a set of all roots of unity of the degree $(u - v)$. So

$$\varepsilon^{\ell v} + \varepsilon^\ell - 2 = 0.$$

Therefore $\varepsilon^{\ell v} = \varepsilon^\ell = 1$, as $|\varepsilon^k| \leqslant 1$ for all $k$. Hence

$$\frac{1}{x-1} \left( x^{u-v} - 1, x^v + x - 2 \right) = \frac{1}{x-1}(x - 1) = 1.$$

Now let us return to the case (5). We decompose polynomials of the left and right sides into irreducible ones. Then we consider the first irreducible polynomial $f(x)$ on the left-hand side. In order for equality to hold, $f(x)$ also has to be present on the right-hand side. So there are some $\mathcal{G}_u$ on the left-hand side and $\mathcal{G}_v$ on the right-hand side divisible by $f(x)$. Hence, $u$ must be

equal to $v$. Divide both sides by $\mathcal{G}_u$ and continue in the same fashion, arriving at the conclusion that the decomposition is unique up to a permutation.

Thus, we proved the following

**Theorem 1.** *There is a one-to-one correspondence between the set of distributions of the rows of matrix $P_n$ and the parameters set $Q_n$.*

Using Theorem 1 one can enumerate the distributions in time proportional to their number. More precisely, one can iterate over all distinct distribution and list them in time $O\left(|Q_n| \cdot \mathrm{poly}(n)\right)$, where $\mathrm{poly}(n)$ is a polynomial of $n$. The only tricky part is to enumerate all the multisets with given sum, but it can be done using one of various recursive algorithms in $O(1)$ amortised time per iteration (e. g. see [4]).

## 3 The number of distributions

Let $p(n,k)$ be the number of partitions of $n$ into exactly $k$ parts. Moreover, let $p(n,k) = 0$, if $k \leqslant 0$ or $n \leqslant 0$, but $p(0,0) = 1$. If we fix $s, L$ and $\ell_s$ then the number of tuples from the set $Q_n$ with these parameters is equal to $p(L-\ell_s, s-1)$. Obviously there are only $n$ tuples with $s = 1$: $(1,1,\ldots,1,1), (1,1,\ldots,1,0), \ldots, (1,0,\ldots,0,0), (0,0,\ldots,0,0)$. We will consider this case separately and we will assume that $s \geqslant 2$. Finally, note that $\sum\limits_{L=\ell_s}^{n-s} p(L-\ell_s, s-1) = \sum\limits_{L=0}^{n-s-\ell_s} p(L, s-1)$. Then

$$|Q_n| = \left[\sum_{s=2}^{n-1}\sum_{\ell_s=0}^{n-1}\sum_{L=0}^{n-s-\ell_s} p(L, s-1)\right] + n. \tag{6}$$

We make one more note to be used later.

**Lemma 2.** $p(n,k) = p(n-1,k-1) + p(n-k,k)$.

*Proof.* Note that the partition of number $n$ into $k$ parts can either include some number of 1's or not include any. In the first case, there is a one-to-one correspondence between such partitions and (unconstrained) partitions of $n-1$ into $k-1$ parts (just put additional 1 to a partition) — there are $p(n-1,k-1)$ of them. In the second case, there is a correspondence between such partitions and (unconstrained) partitions of $n-k$ into $k$ parts (just add 1 to each number in partition) — there are $p(n-k,k)$ of them. □

We now show that the expression (6) can be simplified.

**Theorem 2.** $|Q_n| = \sum_{j=1}^{n-1} p(j) + 1$, *where* $p(j) = \sum_{s=1}^{j} p(j,s)$, $n > 3$.

*Proof (by induction).* For $n = 4$ formula (6) gives $|Q_4| = 7$. At the same time $p(3) + p(2) + p(1) + 1 = 3 + 2 + 1 + 1 = 7$.

Let us show the induction step. In other words, let us prove that the following holds

$$\sum_{s=2}^{n}\sum_{\ell_s=0}^{n}\sum_{L=0}^{n-s+1-\ell_s} p(L,s-1) - \sum_{s=2}^{n-1}\sum_{\ell_s=0}^{n-1}\sum_{L=0}^{n-s-\ell_s} p(L,s-1) = p(n) - 1.$$

$$\sum_{s=2}^{n}\sum_{\ell_s=0}^{n}\sum_{L=0}^{n-s+1-\ell_s} p(L,s-1) - \sum_{s=2}^{n-1}\sum_{\ell_s=0}^{n-1}\sum_{L=0}^{n-s-\ell_s} p(L,s-1) =$$

$$= \sum_{s=2}^{n-1}\left[\sum_{\ell_s=0}^{n}\sum_{L=0}^{n-s+1-\ell_s} p(L,s-1) - \sum_{\ell_s=0}^{n-1}\sum_{L=0}^{n-s-\ell_s} p(L,s-1)\right] +$$

$$+ \underbrace{\sum_{\ell_s=0}^{n}\sum_{L=0}^{n-n+1-\ell_s} p(L,n-1)}_{=0} =$$

$$= \sum_{s=2}^{n-1}\left[\sum_{\ell_s=0}^{n-1}\left[\sum_{L=0}^{n-1-s+2-\ell_s} p(L,s-1) - \sum_{L=0}^{n-1-s+1-\ell_s} p(L,s-1)\right] + \right.$$

$$\left. + \underbrace{\sum_{L=0}^{n+1-s-n} p(L,s-1)}_{=0}\right] =$$

$$= \sum_{s=2}^{n-1}\sum_{\ell_s=0}^{n-1} p(n+1-s-\ell_s, s-1).$$

Now we will prove by induction that the latter is equal to $p(n) - 1$. For $n = 4$ both of them are equal to 4.

Induction step: let us check the validity of equation

$$p(n+1)-1=\sum_{s=2}^{n}\sum_{\ell_s=0}^{n}p(n+2-s-\ell_s,s-1)=$$

$$=\sum_{s=2}^{n}\sum_{\ell_s=-1}^{n-1}p(n+1-s-\ell_s,s-1)=$$

$$=\sum_{s=2}^{n-1}\sum_{\ell_s=0}^{n-1}p(n+1-s-\ell_s,s-1)+\underbrace{\sum_{\ell_s=-1}^{n-1}p(n+1-n-\ell_s,n-1)}_{=p(1-\ell_s,n-1)=0}+$$

$$+\sum_{s=2}^{n}p(n+1-s-(-1),s-1)=p(n)-1+\sum_{s=2}^{n}p(n+2-s,s-1).$$

Since

$$p(n)=\sum_{s=1}^{n}p(n,s),\ p(n+1)=\sum_{s=1}^{n+1}p(n+1,s),$$

the equation becomes

$$\sum_{s=1}^{n+1}p(n+1,s)=\sum_{s=1}^{n+1}p(n,s)+\sum_{s=1}^{n-1}p(n+1-s,s). \tag{7}$$

Let us continue transforming the expression (7):

$$\sum_{s=1}^{n}p(n+1,s)+\underbrace{p(n+1,n+1)}_{=1}=\sum_{s=2}^{n+1}p(n,s-1)+\sum_{s=1}^{n}p(n+1-s,s)-$$

$$-\underbrace{p(n+1-n,n)}_{=p(1,n)=0}=\sum_{s=1}^{n}p(n,s-1)+\underbrace{p(n,n)}_{=1}-$$

$$-\underbrace{p(n,0)}_{=0}+\sum_{s=1}^{n}p(n+1-s,s).$$

Eventually,

$$\sum_{s=1}^{n}p(n+1,s)=\sum_{s=1}^{n}p(n,s-1)+\sum_{s=1}^{n-1}p(n+1-s,s).$$

Lemma 2 ends the proof.

□

Theorem 2 makes it possible to solve the task of counting all the distributions. We just have to calculate values of $p(j,s)$ for $j \in \{1,\dots,n-1\},\ s \in \{1,\dots,j\}$, then all the $p(j)$ and finally $|Q_n|$. The complexity of computing $p(j,s)$ dominates the other steps and according to Lemma 2 may be done in $O(n^2)$ additions of $n$-bit numbers. Thus we need $O(n^3)$ bit operations for the counting problem.

## 4 Asymptotical approximation

In [5] the following asymptotic formula for the number of partitions $p(n)$ was obtained:

$$p(n) \sim \frac{1}{4\sqrt{3}n} e^{\pi\sqrt{\frac{2n}{3}}}.$$

Hence

$$|Q_n| \sim \sum_{j=1}^{n-1} \frac{1}{4\sqrt{3}j} e^{\pi\sqrt{\frac{2j}{3}}} + 1 \text{ as } n \to \infty. \tag{8}$$

The following Lemma allows us to claim it.

**Lemma 3.** *Let $f(n) \sim g(n)$ as $n \to \infty$, $f(n) \geqslant 0$, $g(n) \geqslant 0$, $f(n)$ and $g(n)$ monotonically increase and are unbounded, $F(n) = \sum_{k=1}^{n} f(k)$, $G(n) = \sum_{k=1}^{n} g(k)$, then $F(n) \sim G(n),\ n \to \infty$.*

You can find the proof of Lemma 3 in Appendix B.

Now we will prove an auxiliary Lemma.

**Lemma 4.**

$$\sum_{j=1}^{n-2\sqrt{n}\ln n} \frac{1}{4\sqrt{3}j} e^{\pi\sqrt{\frac{2j}{3}}} = o\left(\frac{1}{4\sqrt{3}n} e^{\pi\sqrt{\frac{2n}{3}}}\right) \text{ as } n \to \infty.$$

*Proof.* Let us show that

$$\lim_{n\to\infty} \sum_{j=1}^{n-2\sqrt{n}\ln n} \frac{n}{j} e^{\pi\sqrt{\frac{2}{3}}\left(\sqrt{j}-\sqrt{n}\right)} = 0.$$

Since

$$\frac{n}{j} < n,\ j < n - 2\sqrt{n}\ln n$$

and

$$n - 2\sqrt{n}\ln n < n,$$

it is sufficient to prove that

$$\lim_{n\to\infty} n^2 e^{\pi\sqrt{\frac{2}{3}}\left(\sqrt{n-2\sqrt{n}\ln n}-\sqrt{n}\right)} = 0.$$

From

$$\sqrt{n - 2\sqrt{n}\ln n} = \sqrt{n}\sqrt{1 - \frac{2\ln n}{\sqrt{n}}} = \sqrt{n}\left(1 - \frac{2\ln n}{2\sqrt{n}} + o\left(\frac{\ln n}{\sqrt{n}}\right)\right) =$$
$$= \sqrt{n} - \ln n + o(\ln n)$$

it follows that

$$\lim_{n\to\infty} n^2 e^{\pi\sqrt{\frac{2}{3}}\left(\sqrt{n-2\sqrt{n}\ln n}-\sqrt{n}\right)} = \lim_{n\to\infty} n^2 e^{\pi\sqrt{\frac{2}{3}}(-\ln n+o(\ln n))} =$$
$$\lim_{n\to\infty} n^{2-\pi\sqrt{\frac{2}{3}}+o(1)}.$$

Whereas the exponent is negative, Lemma is proved. □

**Theorem 3.**

$$\sum_{j=1}^{n} p(j) \sim \frac{e^{\sqrt{\frac{2n}{3}}}}{2\sqrt{2}\pi\sqrt{n}} \text{ as } n \to \infty.$$

*Proof.* It can be proved that there exists a number $N_0$ such that the function on the right-hand side monotonically increases on $[N_0; +\infty)$. We will estimate the sum from $N_0$ to $n$ as first $N_0 - 1$ summands do not influence the asymptotic.

The following holds

$$\int_{N_0}^{n} \frac{e^{\sqrt{\frac{2x}{3}}\pi} dx}{4\sqrt{3}x} =$$
$$= \int_{N_0}^{n} \frac{1}{2\sqrt{2}\pi\sqrt{x}} de^{\sqrt{\frac{2x}{3}}\pi} = \frac{e^{\sqrt{\frac{2x}{3}}\pi}}{2\sqrt{2}\pi\sqrt{x}}\Bigg|_{N_0}^{n} + \int_{N_0}^{n} \frac{e^{\sqrt{\frac{2x}{3}}\pi}}{2\sqrt{2}\pi x^{3/2}} dx =$$
$$= \frac{e^{\sqrt{\frac{2n}{3}}\pi}}{2\sqrt{2}\pi\sqrt{n}} - \frac{e^{\sqrt{\frac{2N_0}{3}}}}{2\sqrt{2}\pi\sqrt{N_0}} + \int_{N_0}^{n} \frac{e^{\sqrt{\frac{2x}{3}}\pi}}{2\sqrt{2}\pi x^{3/2}} dx.$$

Now we will show that the last two summands here are $o(e^{\sqrt{n}} n^{-\frac{1}{2}})$. For the first of them it is obvious, so let us focus on the second. For this purpose

we note that

$$\sum_{j=N_0}^{n-1} f(x) \leqslant \int_{N_0}^{n} f(x)dx \leqslant \sum_{j=N_0+1}^{n} f(x)$$

for non-decreasing function $f$. So by Lemma 4

$$\int_{N_0}^{n} \frac{e^{\sqrt{\frac{2x}{3}}}\pi}{2\sqrt{2}\pi x^{3/2}} \leqslant \sum_{N_0+1}^{n} \frac{e^{\sqrt{\frac{2x}{3}}}\pi}{2\sqrt{2}\pi x^{3/2}} \sim \sum_{j=n-2\sqrt{n}\ln n}^{n} \frac{e^{\sqrt{\frac{2x}{3}}}}{2\sqrt{2}\pi x^{3/2}} \sim \sum_{j=n-2\sqrt{n}\ln n}^{n} \frac{e^{\sqrt{x}}}{x^{3/2}}$$
$$\leqslant \frac{e^{\sqrt{n}}}{(n-2\sqrt{n}\ln n)^{\frac{3}{2}}} \cdot 2\sqrt{n}\ln n \sim \frac{e^{\sqrt{n}}}{n^{\frac{3}{2}}}\sqrt{n}\ln n = \frac{\ln n}{n}e^{\sqrt{n}}.$$

Finally for $n \to \infty$

$$\left(\frac{\ln n}{n}e^{\sqrt{n}}\right)\left(\frac{e^{\sqrt{n}}}{\sqrt{n}}\right)^{-1} = \frac{\ln n}{\sqrt{n}} \to 0.$$

In addition,

$$\sum_{j=N_0+1}^{n} f(x) - \sum_{j=N_0}^{n-1} f(x) = f(n) - f(N_0) = \frac{e^{\sqrt{\frac{2n}{3}}}}{4\sqrt{3}n} - \frac{e^{\sqrt{\frac{2N_0}{3}}}}{4\sqrt{3}N_0} = o\left(\frac{e^{\sqrt{n}}}{\sqrt{n}}\right)$$

the following equality holds

$$\sum_{j=N_0}^{n} \frac{e^{\sqrt{\frac{2j}{3}}}}{4\sqrt{3}j} \sim \int_{N_0}^{n} \frac{e^{\sqrt{\frac{2x}{3}}}}{4\sqrt{3}x}dx,$$

and it concludes the proof of the Theorem. □

According to the above Theorem and the note after Theorem 1 we can enumerate all the distributions in time $2^{O(\sqrt{n})}$ that is obviously substantially better than brute force algorithm with complexity $2^{\Omega(n)}$.

## 5 Conclusion

We obtained a general form of distributions in DDT. Moreover, we provided an efficient method for computing the distribution in a row with given index. The obtained results imply a possibility to substantially accelerate the construction of all possible distributions. We showed that all the distributions now can be generated in time proportional to the amount of them. We have proved that the number of distinct distributions is $2^{O(\sqrt{n})}$, so the

whole generating algorithm would take $2^{O(\sqrt{n})}$ operations. At the same time the brute force algorithm requires $2^{\Omega(n)}$ operations.

## References


[1] GOST 28147-89. National Standard of the USSR. Cryptographic Protection for Data Processing System. Government Committee of the USSR for Standards, 1989.

[2] GOST R 34.12-2015. National Standard of the Russian Federation. Cryptographic Data Security. Block Ciphers. Federal Agency on Technical Regulating and Metrology. Standartinform, 2015.

[3] Lipmaa H., Moriai S. Efficient Algorithms for Computing Differential Properties of Addition. Software Encryption, 2002, pp. 336–350.

[4] Kelleher J., O'Sullivan B. Generating All Partitions: A Comparison Of Two Encodings. CoRR, 2009, Vol. abs/0909.2331, http://arxiv.org/abs/0909.2331.

[5] Hardy G. H., Ramanujan S. Asymptotic Formulaæ in Combinatory Analysis. Proceedings of the London Mathematical Society, 1918, Vol. s2-17, No. 1, pp. 75–115.


## Appendix

## A The proof of Lemma 1

Let us find a rule by which, knowing the form of matrix $P_n$ for some $n$, we can construct such a matrix for $P_{n+1}$. Write down all variables, setting two most significant bits apart:

$$x = x_n \cdot 2^n + x_{n-1} \cdot 2^{n-1} + \widehat{x},$$

$$y = y_n \cdot 2^n + y_{n-1} \cdot 2^{n-1} + \widehat{y},$$

$$\Delta x = \Delta x_n \cdot 2^n + \Delta x_{n-1} \cdot 2^{n-1} + \Delta\widehat{x},$$

$$\Delta f = \Delta f_n \cdot 2^n + \Delta f_{n-1} \cdot 2^{n-1} + \Delta\widehat{f},$$

where

$$x,\ y,\ \Delta x,\ \Delta f \in \{0, \ldots, 2^{n+1} - 1\},$$

$$x_n,\ \Delta y_n,\ \Delta x_n,\ \Delta f_n \in \{0, 1\},$$

$$x_{n-1},\ \Delta y_{n-1},\ \Delta x_{n-1},\ \Delta f_{n-1} \in \{0, 1\},$$

$$\widehat{x},\ \widehat{y},\ \Delta\widehat{x},\ \Delta\widehat{f} \in \{0,\dots,2^{n-1}-1\}.$$

Besides, denote

$$m_k(a,b,c) = \begin{cases} 0, & \text{if } a+b+c < 2^k, \\ 1, & \text{if } a+b+c \geqslant 2^k, \end{cases}$$

the function that returns a carry bit of addition $a+b+c$ modulo $2^k$. We also denote $m_k(a,b) = m_k(a,b,0)$. In addition let us note that $m_1(a,b) = a \,\&\, b$ and $m_1(a,b,c) = a \,\&\, b \vee a \,\&\, c \vee b \,\&\, c$ (the last is called the «majority» function).

Let us denote:

$$c = m_{n-1}(\widehat{x},\widehat{y}),$$
$$c_\Delta = m_{n-1}(\widehat{x} \oplus \Delta\widehat{x},\widehat{y}).$$

So let us rewrite the first part of expression (1) in more detail:

$$\begin{gathered}(x+\Delta x) \boxplus_{n+1} y = \big[(\widehat{x} \oplus \Delta\widehat{x}) \boxplus_{n-1} \widehat{y}\big] + \\ + \big[(x_{n-1} \oplus \Delta x_{n-1}) + y_{n-1}\big] \cdot 2^{n-1} + \big[(x_n \oplus \Delta x_n) + y_n\big] \cdot 2^n = \\ = \big[(\widehat{x} \oplus \Delta\widehat{x}) \boxplus_{n-1} \widehat{y}\big] + \big[c_\Delta \oplus \big((x_{n-1} \oplus \Delta x_{n-1}) \oplus y_{n-1}\big)\big] \cdot 2^{n-1} + \\ + \big[m_1(c_\Delta, x_{n-1} \oplus \Delta x_{n-1}, y_{n-1}) \oplus \big((x_n \oplus \Delta x_n) \oplus y_n\big)\big] \cdot 2^n.\end{gathered}$$

Similarly, we get

$$\begin{gathered}x \boxplus_{n+1} y = \\ = (\widehat{x} \boxplus_{n-1} \widehat{y}) + (c \oplus x_{n-1} \oplus y_{n-1}) \cdot 2^{n-1} + \\ + \big[m_1(c, x_{n-1}, y_{n-1}) \oplus (x_n \oplus y_n)\big] \cdot 2^n.\end{gathered}$$

Then the equation (1) can be rewritten as

$$\begin{gathered}\Delta f = \big[(\widehat{x} \boxplus_{n-1} \widehat{y}) \oplus \big((\widehat{x} \oplus \Delta\widehat{x}) \boxplus_{n-1} \widehat{y}\big)\big] + (\Delta x_{n-1} \oplus c \oplus c_\Delta) \cdot 2^{n-1} + \\ + \big[\Delta x_n \oplus m_1(c_\Delta, x_{n-1} \oplus \Delta x_{n-1}, y_{n-1}) \oplus m_1(c, x_{n-1}, y_{n-1})\big] \cdot 2^n.\end{gathered}$$

Let us denote

$$\varphi(\widehat{x},\widehat{y},\Delta\widehat{x}) = \big[(\widehat{x} \boxplus_{n-1} \widehat{y}) \oplus \big((\widehat{x} \oplus \Delta\widehat{x}) \boxplus_{n-1} \widehat{y}\big)\big].$$

Hence, the equation (1) is equivalent to the following system:

$$\begin{cases} c = m_{n-1}(\widehat{x}, \widehat{y}), & (9)\\ c_\Delta = m_{n-1}(\widehat{x} \oplus \Delta\widehat{x}, \widehat{y}), & (10)\\ \varphi(\widehat{x}, \widehat{y}, \Delta\widehat{x}) = \Delta\widehat{f}, & (11)\\ c \oplus c_\Delta = \underbrace{\Delta f_{n-1} \oplus \Delta x_{n-1}}_{z_{n-1}}, & (12)\\ m_1(c_\Delta, x_{n-1} \oplus \Delta x_{n-1}, y_{n-1}) \oplus m_1(c, x_{n-1}, y_{n-1}) = \underbrace{\Delta f_n \oplus \Delta x_n}_{z_n}. & (13) \end{cases}$$

Let us fix $\Delta x$ and $\Delta f$ modulo $2^{n+1}$. We denote the set of solutions $(x, y)$ of the equation (1) modulo $2^k$, where $x, y \in \{0, \dots, 2^k - 1\}$ by $M_k$. Obviously, $M_{n-1}$ is a set of solutions of equations (9)–(11), $M_n$ — solutions of equations (9)–(12) and $M_{n+1}$ — of equations (9)–(13). Additionally, we introduce two additional sets: $\widehat{U}_{n-1}$ is the set of solutions $(\widehat{x}, \widehat{y})$ of the system (9)–(12) and $\widehat{U}^{(c)}_{n-1}$ for $c \in \{0, 1\}$ is a subset of $\widehat{U}_{n-1}$ where $m_{n-1}(\widehat{x}, \widehat{y}) = c$.

We will try to find sets $(x_{n-1}, y_{n-1}, c, c_\Delta)$ satisfying conditions (12), (13). But noting that $c_\Delta = c \oplus z_{n-1}$ we will search for solutions $(x_{n-1}, y_{n-1}, c)$ of equation

$$m_1(c \oplus z_{n-1}, x_{n-1} \oplus \Delta x_{n-1}, y_{n-1}) \oplus m_1(c, x_{n-1}, y_{n-1}) = z_n. \qquad (14)$$

Depending on values of $x_{n-1}$ and $y_{n-1}$ this equation may be rewritten as

| $x_{n-1}$ | $y_{n-1}$ | (14) |
|---|---|---|
| 0 | 0 | $m_1((c \oplus z_{n-1}), \Delta x_{n-1}, 0) \oplus m_1(0, 0, c) = z_n,$ |
| 0 | 1 | $m_1((c \oplus z_{n-1}), \Delta x_{n-1}, 1) \oplus m_1(0, 1, c) = z_n,$ |
| 1 | 0 | $m_1((c \oplus z_{n-1}), 1 \oplus \Delta x_{n-1}, 0) \oplus m_1(1, 0, c) = z_n,$ |
| 1 | 1 | $m_1((c \oplus z_{n-1}), 1 \oplus \Delta x_{n-1}, 0) \oplus m_1(1, 1, c) = z_n.$ |

Or, equivalently,

| $x_{n-1}$ | $y_{n-1}$ | (14) |
|---|---|---|
| 0 | 0 | $(c \oplus z_{n-1}) \cdot \Delta x_{n-1} = z_n,$ |
| 0 | 1 | $\big[(c \oplus z_{n-1}) \cdot \Delta x_{n-1} \vee (c \oplus z_{n-1}) \vee \Delta x_{n-1}\big] \oplus c = z_n,$ |
| 1 | 0 | $(c \oplus z_{n-1}) \cdot (1 \oplus \Delta x_{n-1}) \oplus c = z_n,$ |
| 1 | 1 | $\big[(c \oplus z_{n-1}) \cdot (1 \oplus \Delta x_{n-1}) \vee (c \oplus z_{n-1}) \vee (1 \oplus \Delta x_{n-1})\big] \oplus 1 = z_n.$ |

Finally, simplifying, we obtain

| $x_{n-1}$ | $y_{n-1}$ | (14) |
|---|---|---|
| 0 | 0 | $(c \oplus z_{n-1}) \cdot \Delta x_{n-1} = z_n,$ |
| 0 | 1 | $(c \oplus z_{n-1}) \vee \Delta x_{n-1} = z_n \oplus c,$ |
| 1 | 0 | $(c \oplus z_{n-1}) \cdot \Delta \overline{x_{n-1}} = z_n \oplus c,$ |
| 1 | 1 | $(c \oplus z_{n-1}) \vee \Delta \overline{x_{n-1}} = \overline{z_n}.$ |

Let us consider two cases, $\Delta x_{n-1} = 0$ and $\Delta x_{n-1} = 1$, separately. In the first case we see:

| $x_{n-1}$ | $y_{n-1}$ | (14) |
|---|---|---|
| 0 | 0 | $0 = z_n,$ |
| 0 | 1 | $c \oplus z_{n-1} = z_n \oplus c,$ |
| 1 | 0 | $c \oplus z_{n-1} = z_n \oplus c,$ |
| 1 | 1 | $1 = \overline{z_n}.$ |

That is, in fact we have only two conditions:

$$z_n = 0,\ z_{n-1} = z_n.$$

Thus, depending on the values $z_{n-1}$ and $z_n$ (that is, on the values of $\Delta x$ and $\Delta f$) the equation (14) may have a varying number of solutions $\sigma(z_{n-1}, z_n)$:

1. if $z_{n-1} = z_n = 0$, then $\sigma(z_{n-1}, z_n) = 4$.
2. if $z_{n-1} = 1, z_n = 0$, then $\sigma(z_{n-1}, z_n) = 2$.
3. if $z_{n-1} = z_n = 1$, then $\sigma(z_{n-1}, z_n) = 2$.
4. if $z_{n-1} = 0, z_n = 1$, then $\sigma(z_{n-1}, z_n) = 0$.

that is, the solution set is $\varepsilon \times \{0, 1\}$, where $\varepsilon$ is a set of zero, two or four pairs $(x_{n-1}, y_{n-1})$.

In the case of $\Delta x_{n-1} = 1$ the equation (14) has more complex form:

| $x_{n-1}$ | $y_{n-1}$ | (14) |
|---|---|---|
| 0 | 0 | $c \oplus z_{n-1} = z_n,$ |
| 0 | 1 | $1 = z_n \oplus c,$ |
| 1 | 0 | $0 = z_n \oplus c,$ |
| 1 | 1 | $1 = c \oplus z_{n-1} = \overline{z_n},$ |

which is equivalent to:

$$c = z_{n-1} \oplus z_n,\ c = z_n \oplus 1,\ c = z_n,\ c = z_{n-1} \oplus z_n \oplus 1.$$

It is obvious that at any values $z_{n-1}, z_n \in \{0, 1\}$ exactly two of these conditions will be fulfilled, since up to the permutation they are equivalent to

conditions

$$c = 0,\ c = 0,\ c = 1,\ c = 1.$$

So the solution set in this case is $(\varepsilon' \times \{0\}) \cup (\varepsilon'' \times \{1\})$, where $\varepsilon'$, $\varepsilon''$ are sets of pairs $(x_{n-1}, y_{n-1})$, $|\varepsilon'| = |\varepsilon''| = 2$.

Let us introduce the function $S_k(T, \alpha, \beta) = \{(x+\alpha\cdot 2^k, y+\beta\cdot 2^k) \mid (x, y) \in T\}$, where $T$ is a set of pairs $(x, y)$ for some $x, y \in \{0, \dots, 2^k - 1\}$. It is easy to see that for each $T$ holds $|S_k(T, \alpha, \beta)| = |T|$. Denote

$$\Psi = \Big\{(x_{n-1}, y_{n-1}, c) \;\Big|\; x_{n-1}, y_{n-1}, c \in \{0, 1\},\ (x_{n-1}, y_{n-1}, c) \text{ are solutions of the equation (14)}\Big\}.$$

We note that in the above notation

$$M_n = \bigsqcup_{x_{n-1}, y_{n-1} \in \{0,1\}} S_{n-1}(\widehat{U}_{n-1}, x_{n-1}, y_{n-1}),$$

$$M_{n+1} = \bigsqcup_{x_n, y_n \in \{0,1\}} \ \bigsqcup_{(x_{n-1}, y_{n-1}, c) \in \Psi} S_n(S_{n-1}(\widehat{U}^{(c)}_{n-1}, x_{n-1}, y_{n-1}), x_n, y_n).$$

Furthermore,

$$|M_n| = 4|\widehat{U}_{n-1}|,$$

$$|M_{n+1}| = 4 \sum_{(x_{n-1}, y_{n-1}, c) \in \Psi} |\widehat{U}^{(c)}_{n-1}|,$$

$$\widehat{U}_{n-1} = \widehat{U}^{(0)}_{n-1} \bigsqcup \widehat{U}^{(1)}_{n-1}.$$

That is,

$$\frac{|M_{n+1}|}{|M_n|} = \frac{\sum_{(x_{n-1}, y_{n-1}, c) \in \Psi} |\widehat{U}^{(c)}_{n-1}|}{|\widehat{U}_{n-1}|} = \begin{cases} \frac{\sum_{\Psi} |\widehat{U}_{n-1}|}{|\widehat{U}_{n-1}|}, & \text{if } \Delta x_{n-1} = 0, \\ \frac{2|\widehat{U}^{(0)}_{n-1}| + 2|\widehat{U}^{(1)}_{n-1}|}{|\widehat{U}_{n-1}|}, & \text{if } \Delta x_{n-1} = 1, \end{cases}$$

$$= \begin{cases} |\Psi|, & \Delta x_{n-1} = 0, \\ \frac{2|\widehat{U}_{n-1}|}{|\widehat{U}_{n-1}|}, & \Delta x_{n-1} = 1, \end{cases} = \begin{cases} \sigma(z_{n-1}, z_n), & \Delta x_{n-1} = 0, \\ 2, & \Delta x_{n-1} = 1, \end{cases}$$

Lemma 1 is proved.

## B The proof of Lemma 3

We will show that for any $\varepsilon > 0$ and $n \geqslant N$ for some number $N$, holds $\frac{F(n)}{G(n)} \leqslant 1 + \varepsilon$. By the assumption of Lemma, $\frac{f(k)}{g(k)} \leqslant 1 + \frac{\varepsilon}{2}$ for all $k \geqslant N_1$

for some number $N_1$. Equivalently, for all $k \geqslant N_1$ holds $f(k) \leqslant (1 + \frac{\varepsilon}{2})g(k)$. Then

$$F(n) = \sum_{k=1}^{n} f(k) =$$

$$= \sum_{k=1}^{N_1-1} f(k) + \sum_{k=N_1}^{n} f(k) \leqslant \sum_{k=1}^{N_1-1} f(k) + \left(1 + \frac{\varepsilon}{2}\right) \sum_{k=N_1}^{n} g(k) =$$

$$\sum_{k=1}^{N_1-1} f(k) - \left(1 + \frac{\varepsilon}{2}\right) \sum_{k=1}^{N_1-1} g(k) + \left(1 + \frac{\varepsilon}{2}\right) \sum_{k=1}^{n} g(k) =$$

$$= \sum_{k=1}^{N_1-1} \left(f(k) - \left(1 + \frac{\varepsilon}{2}\right) g(k)\right) + \left(1 + \frac{\varepsilon}{2}\right) G(n).$$

That is, for all $n \geqslant N_1$ for some number $N_1$

$$F(n) \leqslant \left(1 + \frac{\varepsilon}{2}\right) G(n) + c.$$

Since $g(k)$ is a monotonically increasing unbounded function, then for all $n \geqslant N_2$ for some number $N_2$ holds

$$g(n) \geqslant c \cdot \frac{2}{\varepsilon}.$$

Then

$$G(n) \geqslant c \cdot \frac{2}{\varepsilon},$$

hence

$$c \leqslant \frac{\varepsilon}{2} G(n).$$

And for all $n \geqslant \max\{N_1, N_2\}$

$$F(n) \leqslant \left(1 + \frac{\varepsilon}{2}\right) G(n) + c \leqslant \left(1 + \frac{\varepsilon}{2}\right) G(n) + \frac{\varepsilon}{2} G(n) = (1 + \varepsilon) G(n).$$

Similarly, for any $\varepsilon > 0$ starting with some $n$, $\frac{F(n)}{G(n)} \geqslant 1 - \varepsilon$ holds. So

$$\lim_{n \to \infty} \frac{F(n)}{G(n)} = 1 \Leftrightarrow F(n) \sim G(n).$$

# New Classes of 8-bit Permutations Based on a Butterfly Structure

Denis Fomin

Technical Committee for Standardization
«Cryptography and security mechanisms» (TC 26), Moscow, Russia
deniskafomin@gmail.com

**Abstract**

This work introduces new classes of 8-bit permutation based on a butterfly structure. These classes set up a new way for generating $2n$-bit permutation from $n$-bit ones. We introduce some classes that contain permutations with good cryptographic properties and could be efficiently implemented for hardware and software applications.

**Keywords:** boolean function, S-box, butterfly structure, bent function.

## 1 Introduction

Permutations are essential part of huge classes of cryptographic functions. These functions are used to build symmetric encryption functions such as stream ciphers, block ciphers and hash functions. According to Shannon's criteria [1] every strong cryptographic function should consist of ones that provide confusion and diffusion. One of the well studied way to hide the relationship between the key and plaintext (or provide confusion) is using a substitutional-box – S-Box. Today, after decades of cryptoanalysis of modern cryptographic functions there are well studied properties of S-Box to be a part of secure cryptographic function.

There are a lot of reasons to build S-Boxes from smaller ones: good software implementation with precomputed tables, better bit-sliced implementation, implementation for lightweight cryptography with smaller tables or lower gate count, efficient masking in hardware [2, 3]. Permutations which are build from smaller ones are more secure against cache timing attacks than those relying on general 8-bit S-boxes, which require table lookups in memory [4]. There are known a lot of ways to build large S-Box from smaller one: constructions based on Feistel network [5, 6, 7], Misty network [8, 5, 9], SPN network [10, 11, 12] or other constructions [13].

In this work we will study how to build 8-bit S-box using a butterfly structure that was suggested in [4]. In [4] this structure was obtained while studying decomposition of the Dillon APN permutation [14].

## 2 Definitions and Notations

We will use the following notations and definitions. Let $\mathbb{F}_{2^n}$ be a finite field of size $2^n$. Every $a \in \mathbb{F}_{2^n}$ could be presented as a $n$-bit vector $a = (a_0, a_1, \ldots, a_{n-1})$, $a_i \in \mathbb{F}_2$, $i \in \overline{0, n-1}$. For any $a, b \in \mathbb{F}_{2^n}$ operation $\langle a, b \rangle$ is a dot product: $\sum_{i=0}^{n-1} a_i \cdot b_i$.

S-Box $S$ is any nonlinear function $S : \mathbb{F}_2^n \mapsto \mathbb{F}_2^m$. In this work we will build a nonlinear bijective S-Box. These S-Boxes could be parts of a huge class of cryptographic functions based on block ciphers like SPN-network, Feistel network and etc. For every nonlinear function we can evaluate a set of measures of resistance against known methods of cryptanalysis. They are called properties of nonlinear function. Some of them are defined as follows.

**Definition 1.** *The Walsh-Hadamard Transform (WHT) of an S-Box $S$ $W_S(a, b)$ and fixed values $a \in \mathbb{F}_{2^n}$, $b \in \mathbb{F}_{2^m}$ is defined as:*

$$W_S(a, b) = \sum_{x \in \mathbb{F}_{2^n}} (-1)^{\langle a, x \rangle + \langle b, S(x) \rangle}.$$

This function is known to be used in correlation evaluation between the following Boolean $\langle b, S(x) \rangle$ and linear $\langle a, x \rangle$ functions.

**Definition 2.** *The nonlinearity $N_S$ of an S-Box $S$ is a measure that is defined as follows:*

$$N_S = 2^{n-1} - \frac{1}{2} \max_{a, b \neq 0} |W_S(a, b)| .$$

S-Box with larger nonlinearity has better resistance against linear cryptanalysis. As an example, for $\mathbb{F}_{2^8}$ the permutation that has the largest nonlinearity is the finite field inversion $x^{-1}$ with $N_{x^{-1}} = 112$.

**Definition 3.** *A nonlinear function $S : \mathbb{F}_2^n \mapsto \mathbb{F}_2^m$ is called a bent function when its nonlinearity is equal to $2^{n-1} - 2^{n/2-1}$.*

Let $n = 2m$, $x, y \in \mathbb{F}_{2^m}$. The Maiorana–McFarland construction [15] is the way to construct $2n$ bit bent-function from $n$ bit functions and finite field multiplication: every function $f : V_m \times V_m \mapsto V_n$ that has the following form is a bent function:

$$f(x, y) = \pi(x) \cdot l(y) + f(x),$$

where $\pi : \mathbb{F}_{2^m} \mapsto \mathbb{F}_{2^m}$ is a permutation, $l : \mathbb{F}_{2^m} \mapsto \mathbb{F}_{2^m}$ is a linear permutation and $f : \mathbb{F}_{2^m} \mapsto \mathbb{F}_{2^m}$ is a function.

**Definition 4.** *The algebraic degree of the S-Box $S$* $\deg(S)$ *is the minimum among all maximum numbers of variables of the terms in the algebraic normal form (ANF) of $\langle a, S(x)\rangle$ for all possible values $x$ and $a \neq 0$:*

$$\deg(S) = \min_{a \in \mathbb{F}_{2^m}/0} \deg\left(\langle a, S(x)\rangle\right).$$

For any permutation on $\mathbb{F}_{2^n}$ the maximum value of the algebraic degree is $n-1$.

**Definition 5.** *For a given $a \in \mathbb{F}_{2^m}/0, b \in \mathbb{F}_{2^m}$ we consider*

$$\delta_S(a,b) = \#\left\{x \in \mathbb{F}_{2_n} | S(x+a) + S(x) = b\right\}.$$

*The differential uniformity of an S-Box $S$ is*

$$\delta_S = \max_{a \in \mathbb{F}_{2^m}/0, b} \delta_S(a,b).$$

The S-Box with smaller differential uniformity has the better resistance against differential cryptanalysis. For $\mathbb{F}_{2^8}$, permutation with the smallest known differential uniformity is the finite field inversion $x^{-1}$ with $\delta_{x^{-1}} = 4$.

We will say that two permutation $S_1$ and $S_2$ are linear equivalent if there exist two linear permutations $L_1$ and $L_2$: $S_1 = L_1 \circ S_2 \circ L_2$. We will also say that two permutation are affine equivalent if there exist two affine permutations $A_1$ and $A_2$: $S_1 = A_1 \circ S_2 \circ A_2$.

## 3 Possible constructions

In this work we will study the butterfly structure that has been introduced in [4].

**Definition 6.** *Let $n = 2m$. We will call function $F : \mathbb{F}_{2^n} \mapsto \mathbb{F}_{2^n}$ with input $x_i \| y_i$ and output $x_o \| y_o$ $x_i, y_i, x_o, y_o \in \mathbb{F}_{2^m}$ a generalized butterfly structure if there are exist two functions $F_1, F_2 : \mathbb{F}_{2^m} \times \mathbb{F}_{2^m} \mapsto \mathbb{F}_{2^m}$:*

1. *$y_o$ depends on $x_i$, $y_i$ according to the equation: $y_o = F_1(x_i, y_i)$,*
2. *$y_i$ depends on $x_o$, $y_o$ according to the equation: $y_i = F_2(x_o, y_o)$.*

*When $F_1 = F_2$ function $F$ is a butterfly structure presented in [4].*

**Proposition 1.** *A generalized butterfly structure $F$ is a permutation if and only if for every fixed value $y \in \mathbb{F}_{2^m}$ functions $F_1(x,y)$ and $F_2(x,y)$ are permutations.*

*Proof.* Let $F$ be a permutation and $y \in \mathbb{F}_{2^m}$. Without loss of generality we'll prove it for function $F_1$. $F_1(x,y)$ denotes the least significant bits of the

output. If $F_1$ is not a permutation for a fixed value $y$ then there exist $x_1$ and $x_2 : F_1(x_1, y) = F_1(x_2, y)$ and

$$\#\{y_o \,|y_o = F_1(x, y),\ x \in \mathbb{F}_{2^m}\} \leqslant 2^m - 1$$

and this is a contradiction with the statement that $F$ is a permutation.

If $F_i\ i \in \overline{1,2}$ are permutations in terms on the proposition, there exists only one pair $x_o, y_o$ for every $x_i$ and $y_i$. □

In [4] there was revealed that only one known 6-bit APN permutation is CCZ equivalent to the so-called non bijective butterfly structure and that in our terms $F_1$, $F_2$ are bent functions. We want to construct a permutation with good cryptographic properties that were enumerated in section 2. In contrast with [5] we will focus on the nonlinearity because we can choose $F_1$ and $F_2$ separately and independently.

In this work we will consider that $m = 4$. The core idea of this work is in the following:

1. Choose functions $F_1$, $F_2$ that correspond to Proposition 1
2. These functions can be based on Maiorana–McFarland construction and [16]:

$$F_i'(x, y) = \begin{cases} \pi_i(x) \cdot l_i(y) + f_i(x),\ l_i(y) \neq 0; \\ \widehat{\pi}_i(x),\ l_i(y) = 0; \end{cases} \tag{1}$$

$$F_i''(x, y) = \begin{cases} \pi_i(y) \cdot l_i(x) + f_i(y),\ \pi_i(y) \neq 0; \\ \widehat{\pi}_i(x),\ \pi_i(y) = 0; \end{cases} \tag{2}$$

   where $\pi_i, \widehat{\pi}_i$ are $m$-bit permutations, $l_i$ is an $m$-bit linear permutation and $f_i$ is an $m$-bit function.
3. Make a generalized butterfly structure $F$ based on $F_1$ and $F_2$ and evaluate it's cryptographic properties.

### 3.1 Construction based on $F'$ function

**Proposition 2.** *The function $F_i'(x, y)$ from equation 1 is a bijective function for any fixed value $y$ if and only if $f(x)$ is a constant function.*

*Proof.* If $l_i(y)$ is equal to 0 then the proposition is obvious. Let $l_i(y)$ be not equal to 0.

Let us consider the function $\pi_i(x) \cdot l_i(y) + f_i(x)$. This function is not a permutation for a fixed value $y$ if there are no $x_1, x_2 \in \mathbb{F}_{2^m}$ :

$$\pi_i(x_1) \cdot l_i(y) + f_i(x_1) = \pi_i(x_2) \cdot l_i(y) + f_i(x_2).$$

Let us consider the following equations:

$$\pi_i(x_1) \cdot l_i(y) + f_i(x_1) \neq \pi_i(x_2) \cdot l_i(y) + f_i(x_2) \Leftrightarrow \\ \Leftrightarrow (\pi_i(x_1) + \pi_i(x_2))\, l_i(y) \neq (f_i(x_1) + f_i(x_2)) \quad (3)$$

Only a constant function $f_i(x)$ could satisfy equation (3) for every pair $x_1, x_2 \in \mathbb{F}_{2^m}$ because the set $\{(\pi_i(x_1) + \pi_i(x_2))\, l_i(y) \,|y \in \mathbb{F}_{2^m}\}$ is equal to the set of all invertible elements of finite field $\mathbb{F}_{2^m}$. □

There is another possible construction:

$$\widehat{F}'_i(x,y) = \begin{cases} \pi_i(x) \cdot l_i(y) + a \cdot \pi(x), \ (l_i(y) + a) \neq 0; \\ \widehat{\pi}_i(x), \ (l_i(y) + a) = 0. \end{cases}, \quad (4)$$

where $a \in \mathbb{F}_{2^m}$. It's obvious that constructions in equations 1 and 4 provide affine equivalent constructions. Moreover they provide constructions affine equivalent to the following one:

$$F'_i(x,y) = \begin{cases} \pi_i(x) \cdot y, \ y \neq 0; \\ \widehat{\pi}_i(x), \ y = 0. \end{cases}. \quad (5)$$

Let us denote

$$x \otimes_i y = \begin{cases} \pi_i(x) \cdot y, \ y \neq 0; \\ \widehat{\pi}'_i(x), \ y = 0. \end{cases} \quad (6)$$

We will use new $\otimes_i$ operation[(1)] to represent the construction on the Figure 1. We will call this construction "A".

[(1)]The permutation $\widehat{\pi}'_i(x)$ in the equation 6 is different from $\widehat{\pi}_i(x)$ in the equation 5 only for construction "A" . For this construction $\widehat{\pi}'_i(x) = \widehat{\pi}_i\left(\pi^{-1}(x)\right)$. For other constructions $\widehat{\pi}'_i(x) = \widehat{\pi}_i(x)$.

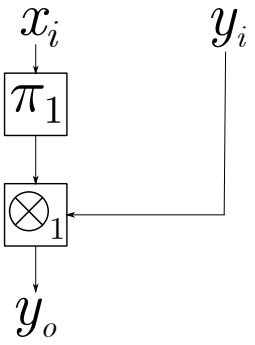



Figure 1: Construction "A"

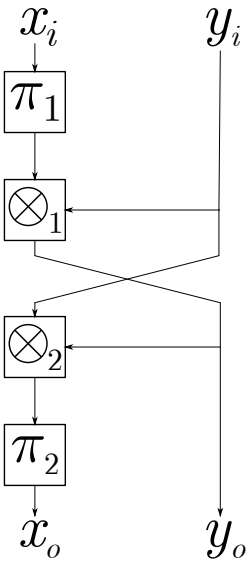



Figure 2: Permutation based on two "A" constructions

The following proposition tells us that at least a part of all WHT of S-Box based on selected construction will have a good nonlinearity.

**Proposition 3.**

$$\left|W_{F_i'(x,y)}\right| \leqslant 2^{m+1}.$$

*Proof.*

$$\left|W_{F_i'(x,y)}(\alpha\|\beta,\gamma)\right| = \left|\sum_{x,y\in\mathbb{F}_{2^m}} (-1)^{\langle\alpha,x\rangle+\langle\beta,y\rangle+\langle\gamma,F_i'(x,y)\rangle}\right| =$$

$$= \left|\sum_{\substack{x,y\in\mathbb{F}_{2^m},\\ y\neq 0}} (-1)^{\langle\alpha,x\rangle+\langle\beta,y\rangle+\langle\gamma,\pi_i(x)\cdot y\rangle} + \sum_{x\in\mathbb{F}_{2^m}} (-1)^{\langle\alpha,x\rangle+\langle\gamma,\widehat{\pi}_i(x)\rangle}\right| =$$

$$= \left|\sum_{\substack{x,y\in\mathbb{F}_{2^m},\\ y\neq 0}} (-1)^{\langle\alpha,x\rangle+\langle\beta,y\rangle+\langle\gamma,\pi_i(x)\cdot y\rangle} \pm \sum_{x\in\mathbb{F}_{2^m}} (-1)^{\langle\alpha,x\rangle} + \sum_{x\in\mathbb{F}_{2^m}} (-1)^{\langle\alpha,x\rangle+\langle\gamma,\widehat{\pi}_i(x)\rangle}\right| \leqslant$$

$$\leqslant \left|W_{\pi_i(x)\cdot y}\right| + \left|-\sum_{x\in\mathbb{F}_{2^m}} (-1)^{\langle\alpha,x\rangle} + \sum_{x\in\mathbb{F}_{2^m}} (-1)^{\langle\alpha,x\rangle+\langle\gamma,\widehat{\pi}_i(x)\rangle}\right|.$$

If $\alpha \neq 0$ the last summand is equal to $2^m$. If $\alpha = 0$

$$\left|-\sum_{x\in\mathbb{F}_{2^m}} (-1)^{\langle\alpha,x\rangle} + \sum_{x\in\mathbb{F}_{2^m}} (-1)^{\langle\gamma,\widehat{\pi}_i(x)\rangle}\right| = \left|-2^m + 0\right|,$$

because $\#\{x \,|\langle \gamma, \widehat{\pi}_i(x)\rangle = 0\} = 2^{m-1}$. And $\pi_i(x) \cdot y$ is a bent function so $\left|W_{\pi_i(x)\cdot y}\right| = 2^m$. □

Let us make a butterfly permutation based on the following construction (see Figure 2):

$$y_o = \begin{cases} \pi_1(x_i) \cdot y_i, y_i \neq 0; \\ \widehat{\pi}_1(x_i),\ y_i = 0. \end{cases}, \tag{7}$$

$$x_o = \begin{cases} \pi_2(y_i \cdot y_o), y_o \neq 0; \\ \widehat{\pi}_2(y_i),\ y_o = 0. \end{cases}, \tag{8}$$

To make evaluations easily we supposed $\pi_1$, $\pi_2$ to be monomial permutations of $\mathbb{F}_{2^m}$ among: $z^1, z^2, z^4, z^7, z^8, z^{11}, z^{13}, z^{14}$. We have implemented this construction (presented in Figure 2) and have used a simple version of an evolutionary algorithm [17] to execute a search among all permutations $\widehat{\pi}_1$, $\widehat{\pi}_2$ for all possible fixed monomial permutations $\pi_1$, $\pi_2$. There are some results that we have obtained:

1. We've found 32 constructions that provide us the way to construct permutations with semi-optimal cryptographic properties:
   - the nonlinearity is equal to 108,
   - the differential uniformity is equal to 6,
   - the algebraic degree is equal to 7.
2. There are all these constructions: $\pi_1(x)$ is any monomial function, $\pi_2(x) = x^\alpha$, $\alpha \in \{7, 11, 13, 14\}$.
3. For other pairs $\pi_1(x)$ and $\pi_2(x)$ permutations have the differential uniformity larger than 12.
4. These properties could be obtained with equal permutations $\widehat{\pi}_1(x)$, $\widehat{\pi}_2(x)$ and, for the note, that semi-optimal cryptographic properties are obtained for all proposed constructions with: $\widehat{\pi}_1(x) = \widehat{\pi}_2(x) = x^{-1}$.
5. These properties could be obtained for $\widehat{\pi}_1(x) \neq \widehat{\pi}_2(x)$ .
6. Semi-optimal cryptographic properties could be obtained even for non monomial permutation $\pi_1(x)$ and $\pi_2(x)$. Let $\mathbb{F}_{2^{2m}} = \mathbb{F}_2(x)/\left(x^4+x+1\right)$. An example of such a permutation is:

$$\widehat{\pi}_1 = \widehat{\pi}_2 = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 0 & 9 & 10 & 15 & 3 & 11 & 13 & 4 & 2 & 6 & 14 & 12 & 1 & 7 & 8 & 5 \end{pmatrix},$$

$$\pi_1 = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 0 & 8 & 9 & 1 & 13 & 5 & 4 & 12 & 7 & 15 & 14 & 6 & 10 & 2 & 3 & 11 \end{pmatrix},$$

$$\pi_2 = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 0 & 14 & 11 & 4 & 2 & 3 & 15 & 1 & 10 & 8 & 12 & 7 & 13 & 9 & 6 & 5 \end{pmatrix}.$$

### 3.2 Construction based on $F''$ function

Let us consider $F_i''(x, y)$ function. Three constructions could be implemented with such function (see Figure 3, 4, 5 ). These constructions have absolutely the same output function $y_o = F_1''(x_i, y_i)$, but constructions "C" and "D" change $y_i$ correspondingly by permutation $\pi_1$ and composition of permutations $\pi_1$ and $\pi_{f1}$. Actually, constructions "C" and "D" are not even possible functions for generalized butterfly construction because in term of our definition $y_i$ is an output of $F_2''(x, y)$ and it can't be changed by $F_1''(x, y)$.

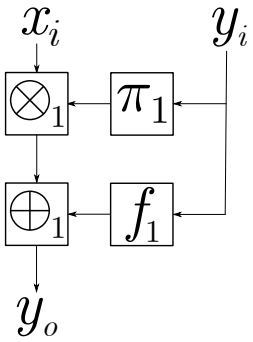



Figure 3: Construction "B"

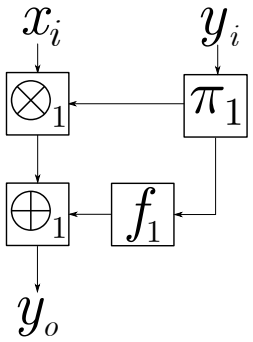



Figure 4: Construction "C"

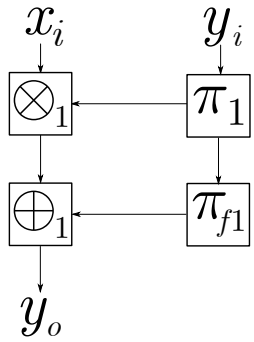



Figure 5: Construction "D"

At the same time all these constructions are bijective for any fixed value $y$ and for any function $f_i$. And output functions have the same nonlinearity as construction "A".

In this work we will study permutation based on two "B" constructions with $f_i = 0$ (see Figure 6). In this construction $y_o = x_i \otimes_1 \pi_1(y_i)$, and $x_o = y_i \otimes_2 \pi_2(x_i \otimes_1 \pi_1(y_i))$. If both $\pi_1(y_i)$ and $\pi_2(y_0)$ are not equal to 0, $x_o = y_i \cdot \pi_2(x_i \cdot \pi_1(y_i))$. We suppose that $\pi_2(x)$ is linear equivalent to $x^\alpha$ and $\pi_1(x)$ is linear equivalent to $x^\beta$, then $x_o$ is linear equivalent to $x_i^\alpha \cdot y_i^{\alpha\beta+1}$.

We've implemented this construction and have used an evolutionary algorithm to execute a search among all permutations $\widehat{\pi}_1$, $\widehat{\pi}_2$ for all fixed monomial permutations $\pi_1$, $\pi_2$. We've found four possible constructions that obtain semi-optimal cryptographic properties:

1. $\pi_1(x) = x$, $\pi_2(x) = x^{13}$,
2. $\pi_1(x) = x^2$, $\pi_2(x) = x^{14}$,
3. $\pi_1(x) = x^4$, $\pi_2(x) = x^7$,
4. $\pi_1(x) = x^8$, $\pi_2(x) = x^{11}$.

Constructions 2 and 4 are inverse permutations for corresponding 1 and 3 constructions.

## 4 Comparison with other constructions

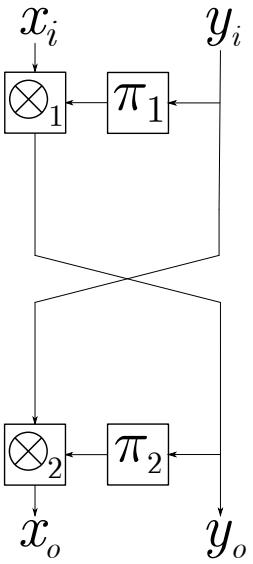



Figure 6: Permutation based on two "B" constructions

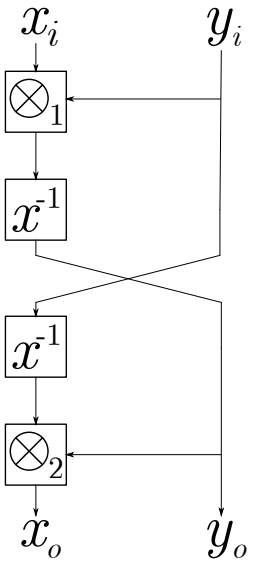



Figure 7: Permutation published in [18]

In [18] the following construction was presented (see Figure 7) (in terms of our work):

$$x_o = \begin{cases} (x_i \cdot y_i)^{-1}, \ y_i \neq 0; \\ \widehat{\pi}_1(x_i), \ y_i = 0. \end{cases},$$

$$y_o = \begin{cases} x_o \cdot y_i^{-1}, \ x_o \neq 0; \\ \widehat{\pi}_2(x_i), \ x_o = 0. \end{cases}.$$

Permutations with following properties were also found in [18]:
– the nonlinearity is equal to 108,
– the differential uniformity is equal to 6,
– the algebraic degree is equal to 7.

Except these properties two additional one were considered in the work:
– absence of fixed points,
– maximum graph algebraic immunity.

Our construction based on two "A" constructions with $\pi_1(x) = \pi_2(x) = x^{-1}$ looks similar with construction presented in 7 (see Figure 2 and Figure 7) but was found independently. There were no theoretical foundation and principles of choosing this particular construction in [18] and we have

no chance to compare it with presented in this work. At the same time in [18] there were found that the value of graph algebraic immunity of a constructed permutation depends on permutations $\widehat{\pi}_i$ and for $\widehat{\pi}_i(x) = x^{-1}$ the permutation has almost optimal cryptographic properties except the value of graph algebraic immunity. We evaluated this value for our constructions and founded out that some construction with $\widehat{\pi}_i(x) = x^{-1}$ have the value of graph algebraic immunity equal to 2. But if we chose $\widehat{\pi}_i(x)$ via our search algorithm it doesn't have simple algebraic structure and this permutation has cryptographic properties like in [18]. In [18] it was also stressed that permutations with such properties have almost optimal cryptographic properties. Comparison with other results could also be found in [18].

We've generalized that construction on Figure 7 and replace $x^{-1}$ by monomial function $\pi_1$ and $\pi_2$. We have searched among permutations $\widehat{\pi}_1$, $\widehat{\pi}_2$ for all fixed monomial permutations $\pi_1$, $\pi_2$ and found the following:

– for the following 12 constructions almost optimal cryptographic properties are obtained:

$$(\pi_1, \pi_2) \in \left\{ \left(x^7, x\right), \left(x^7, x^4\right), \left(x^7, x^7\right), \left(x^{11}, x^2\right), \left(x^{11}, x^8\right), \left(x^{11}, x^{11}\right), \left(x^{13}, x\right), \left(x^{13}, x^4\right), \left(x^{13}, x^{13}\right), \left(x^{14}, x^2\right), \left(x^{14}, x^8\right), \left(x^{14}, x^{14}\right) \right\},$$

– for 4 constructions the differential uniformity is up to 8 and the nonlinearity is up to 104:

$$(\pi_1, \pi_2) \in \left\{ \left(x^7, x^2\right), \left(x^{11}, x\right), \left(x^{13}, x^8\right), \left(x^{14}, x^4\right) \right\},$$

– for 8 constructions the differential uniformity is up to 8 and the nonlinearity is up to 100:

$$(\pi_1, \pi_2) \in \left\{ \left(x^7, x^{11}\right), \left(x^7, x^{14}\right), \left(x^{11}, x^7\right), \left(x^{11}, x^{13}\right), \left(x^{13}, x^{11}\right), \left(x^{13}, x^{14}\right), \left(x^{14}, x^7\right), \left(x^{14}, x^{13}\right) \right\}.$$

## 5 Future work

In this work we only presented several new classes of constructions that can help to find a permutation with rather good cryptographic properties. But at the same time there are too many questions that are necessary to be solved. Among them:

– How many possibilities to choose $F_1$ and $F_2$ to construct a permutation with good cryptographic properties?
– How many possibilities to choose $\pi_i$ and $f_i$ in all these constructions?

– Can we choose permutations $\widehat{\pi}_i$ for our constructions to obtain good cryptographic properties without a search algorithm?
– Can we find a construction that will be an involution?
– Can we use mixed construction for butterfly structure (as example permutation based on "A" and "B" constructions ) to find a permutation with rather good cryptographic properties?
– How to find permutations with good hardware, FPGA or bit-sliced implementations?

## 6 Conclusion

This work has presented some new constructions to build permutation $\mathbb{F}_{2^{2m}} \mapsto \mathbb{F}_{2^{2m}}$, $m = 4$ based on butterfly structure. There are at least 36 new constructions for permutations that have the nonlinearity 108, differential uniformity 6, algebraic degree 7 and the value of graph algebraic immunity 3.

## Acknowledgements

Author would like to thank the reviewers for the helpful remarks.

## References

[1] C. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal, Vol 28, pp. 656–715*, Oktober 1949.

[2] Erik Boss, Vincent Grosso, Tim Güneysu, Gregor Leander, Amir Moradi, and Tobias Schneider. Strong 8-bit sboxes with efficient masking in hardware extended version. *J. Cryptographic Engineering*, 7(2):149–165, 2017.

[3] Sebastian Kutzner, Phuong Ha Nguyen, and Axel Poschmann. Enabling 3-share threshold implementations for all 4-bit s-boxes. In Hyang-Sook Lee and Dong-Guk Han, editors, *ICISC*, volume 8565 of *Lecture Notes in Computer Science*, pages 91–108. Springer, 2013.

[4] Alex Biryukov, Léo Perrin, and Aleksei Udovenko. Reverse-engineering the s-box of streebog, kuznyechik and stribobr1. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT (1)*, volume 9665 of *Lecture Notes in Computer Science*, pages 372–402. Springer, 2016.

[5] Anne Canteaut, Sébastien Duval, and Gaëtan Leurent. Construction of lightweight s-boxes using Feistel and MISTY structures (full version).

*IACR Cryptology ePrint Archive*, 2015:711, 2015. `http://eprint.iacr.org/2015/711`.

[6] Chae Hoon Lim. CRYPTON: A new 128-bit block cipher - specification and analysis, 1998.

[7] Benoît Gérard, Vincent Grosso, María Naya-Plasencia, and François-Xavier Standaert. Block ciphers that are easier to mask: How far can we go? In Guido Bertoni and Jean-Sébastien Coron, editors, *CHES*, volume 8086 of *Lecture Notes in Computer Science*, pages 383–399. Springer, 2013.

[8] Mitsuru Matsui. New block encryption algorithm MISTY. In Eli Biham, editor, *FSE*, volume 1267 of *Lecture Notes in Computer Science*, pages 54–68. Springer, 1997.

[9] Vincent Grosso, Gaëtan Leurent, François-Xavier Standaert, and Kerem Varici. Ls-designs: Bitslice encryption for efficient masked software implementations. In Carlos Cid and Christian Rechberger, editors, *FSE*, volume 8540 of *Lecture Notes in Computer Science*, pages 18–37. Springer, 2014.

[10] François-Xavier Standaert, Gilles Piret, Gaël Rouvroy, Jean-Jacques Quisquater, and Jean-Didier Legat. ICEBERG : An involutional cipher efficient for block encryption in reconfigurable hardware. In Bimal K. Roy and Willi Meier, editors, *FSE*, volume 3017 of *Lecture Notes in Computer Science*, pages 279–299. Springer, 2004.

[11] Vincent Rijmen and Paulo Barreto. The KHAZAD block cipher. 02 2018.

[12] Chae Hoon Lim. A revised version of Crypton - Crypton v1.0. In Lars R. Knudsen, editor, *FSE*, volume 1636 of *Lecture Notes in Computer Science*, pages 31–45. Springer, 1999.

[13] William Stallings. The Whirlpool secure hash function. *Cryptologia*, 30(1):55–67, 2006.

[14] K.A. Browning, J.F. Dillon, M.T. McQuistan, and A.J. Wolfe. An APN permutation in dimension six. 518, 01 2010.

[15] Robert L. McFarland. A family of difference sets in non-cyclic groups. *J. Comb. Theory, Ser. A*, 15(1):1–10, 1973.

[16] Hans Dobbertin. Construction of bent functions and balanced boolean functions with high nonlinearity. In Bart Preneel, editor, *FSE*, volume 1008 of *Lecture Notes in Computer Science*, pages 61–74. Springer, 1994.

[17] Stephan Olariu and Albert Y. Zomaya, editors. *Handbook of Bioinspired Algorithms and Applications.* Chapman and Hall/CRC, 2005.

[18] Reynier Antonio de la Cruz Jiménez. Generation of 8-bit s-boxes having almost optimal cryptographic properties using smaller 4-bit s-boxes and finite field multiplication. `www.cs.haifa.ac.il/~orrd/LC17/paper60.pdf`.

# On a New Classification of the Boolean Functions

Sergey Fedorov

Information Security Institute of Lomonosov University, Moscow, Russia
s.n.feodorov@gmail.com

**Abstract**

We discuss a recent approach to the study of Boolean functions. The approach is based on a notion of $\Delta$-equivalence class, which is a set of Boolean functions all having the same autocorrelation function. Such a classification has an apparently profitable property: a substantial number of the Boolean functions' cryptographic characteristics remains unchanged within a $\Delta$-equivalence class.

**Keywords:** Boolean function, Walsh-Hadamard transform, cross-correlation, autocorrelation, nonlinearity, correlation immunity, propagation criterion, global avalanche characteristics.

## 1 Introduction

Boolean functions, as a natural representation of data transformations in cryptographic primitives, has a crucial significance in cryptography. For choosing a «cryptographically good» function one needs to examine a number of specific properties arisen from Shannon's confusion and diffusion concepts or from the need to withstand a certain method of cryptanalysis, etc.

The notion of cross-correlation function provides a very useful tool for investigation of cryptographic properties of Boolean functions. Autocorrelation function (which is a special case of cross-correlation) admits to define an equivalence relation on the Boolean functions as it is suggested in the forthcoming paper by Logachev, Fedorov and Yashchenko. A $\Delta$-equivalence class consisting of all functions with the same autocorrelation function has the following property that makes clear the importance of this notion for cryptography. Namely, as it stated in Theorems 1, 2 and 3 below, many of the basic cryptographic properties and parameters stay invariant within each $\Delta$-equivalence class. Among them there are nonlinearity, correlation immunity, propagation and avalanche characteristics, etc.

Since any fixation of cryptographic parameters produces a partition of Boolean functions into equivalence classes, we can also assert that the $\Delta$-equivalence refines upon many existing «cryptographic» equivalences at once

so that the corresponding classification is more universal in the following sense: a $\Delta$-equivalence class contains Boolean functions which have not only one common characteristic but a whole collection of common cryptographic properties.

## 2 Necessary notation and concepts

First of all, let's give some common notation.

| | |
|---|---|
| $V_n = \mathbb{F}_2^n$ | is the $n$-dimensional vector space over two-element field $\mathbb{F}_2 = \{0, 1\}$; |
| $\oplus$ | denotes addition operation in $\mathbb{F}_2$ and in $\mathbb{F}_2^n$ (componentwise); |
| $\mathcal{F}_n$ | is the set of all Boolean functions of $n$ variables; |
| $\mathcal{A}_n \subset \mathcal{F}_n$ | is the set of affine functions; |
| $\langle u, v\rangle$ | is the scalar product of vectors $u, v \in V_n$, i. e., $u_1v_1 \oplus \ldots \oplus u_nv_n$; |
| $\mathrm{wt}(\cdot)$ | is the Hamming weight of a Boolean vector or function; |
| $\mathrm{wt}(f \oplus g)$ | is the Hamming distance between functions $f$ and $g$; |
| $\operatorname{supp}\varphi$ | is the support of a function $\varphi\colon V_n \to \mathbb{R}$, i. e., $\{u \in V_n \mid \varphi(u) \neq 0\}$; |
| $\lvert\cdot\rvert$ | is the absolute value of a number or the cardinality of a set. |

### 2.1 Walsh transform and correlation

*The Walsh (or Walsh-Hadamard) transform* for arbitrary function $f \in \mathcal{F}_n$ is defined by setting down so-called *Walsh coefficients*

$$W_f(u) = \sum_{v \in V_n} (-1)^{f(v) \oplus \langle u, v\rangle}, \qquad u \in V_n,$$

all of them together being said to be *the Walsh spectrum* of function $f$. *Walsh spectrum support* is the support of the function $W_f$.

On the base of Walsh spectrum notion one marks out classes of plateaued functions. A function $f \in \mathcal{F}_n$ is *plateaued of order* $r$ iff $W_f(u) \in \{0, \pm 2^{n-r}\}$ for every $u \in V_n$. In this case $|\operatorname{supp} W_f| = 2^{2r}$. A special instance of plateaued functions when $r = \frac{n}{2}$ is *bent functions*.

We will denote by $\Delta_{f,g}$ *the cross-correlation function* of Boolean func-

tions $f, g \in \mathcal{F}_n$,

$$\Delta_{f,g}(u) = \sum_{v \in V_n} (-1)^{f(v) \oplus g(v \oplus u)}, \qquad u \in V_n.$$

In the case when $f = g$ we get the notion of *autocorrelation function* for $f$:

$$\Delta_f(u) = \sum_{v \in V_n} (-1)^{f(v) \oplus f(v \oplus u)}, \qquad u \in V_n.$$

## 2.2 Cryptographic characteristics

Let's rehearse cryptographic properties we will treat of when considering $\Delta$-equivalent functions (one can find more detailed information, for example, in [3]).

The balancedness property of a function is actively exploited in cryptography. A balanced function can be viewed as one whose value is distributed uniformly on $\{0, 1\}$ (each value has probability $\frac{1}{2}$) when its argument is distributed uniformly on $V_n$. Also, inessential variables of a function have obviously a certain significance for its cryptographic application. In some cases the support of a function's Walsh spectrum allows to determine cryptographic qualities of the function as well.

*The nonlinearity* of a function $f$ [6], which is the minimum Hamming distance between $f$ and an affine function, is denoted by

$$\mathrm{nl}(f) = \min_{l \in \mathcal{A}_n} \mathrm{wt}(f \oplus l).$$

When $n$ is even, bent functions provide the example of the functions with maximum nonlinearity.

In addition to this concept, one another characteristic of the difference from the affine functions was treated in [2] and [4]. Namely, *the curvature* [(1)] of a Boolean function $f \in \mathcal{F}_n$ is defined by the equality

$$\mathrm{curv}(f) = \sum_{u \in V_n} |W_f(u)|.$$

The bent functions have the maximum curvature $2^{\frac{3}{2}n}$, whereas the affine functions have the minimum one, i. e., $2^n$.

*The space of linear structures* for a Boolean function $f$ is the following set:

$$L_f = \{u \in V_n \mid \forall v \in V_n \;\; f(v \oplus u) = f(v) \oplus \varepsilon_u, \; \varepsilon_u \in \mathbb{F}_2\}.$$

[(1)] In [2], this parameter has no special name and is denoted by $\sigma(f)$.

Each its nonzero element is called *linear structure* (or *linear translator*) of function $f$. Thus, a vector $u \in V_n$ is a linear structure for $f \in \mathcal{F}_n$ iff the function $f'_u(v) = f(v \oplus u) \oplus f(v)$ is constant on $V_n$. Functions having linear structures are considered as weak for cryptographic usage.

A function $f \in \mathcal{F}_n$ is said to satisfy *the strict avalanche criterion* (SAC) if changing any one of the $n$ argument components causes the changing of exactly half of the values in the $f$'s truth table, that is, the changing of the function's value with probability $\frac{1}{2}$. The notion was introduced in [9] to capture the issues concerning the design of DES-like ciphers.

The *global avalanche characteristics* (GAC) of a function $f$ from $\mathcal{F}_n$ are specified by two numerical parameters that were introduced in [10] in the following way:

$$\sigma_f = \sum_{u \in V_n} \Delta_f^2(u) \quad \text{and} \quad \delta_f = \max_{0 \neq u \in V_n} |\Delta_f(u)|.$$

The less these parameters, the better the cryptographic quality of the function. We note that in fact $\delta_f$ first appeared in [6] where the authors considered the distance of a function $f$ to the set of Boolean functions having linear structures. Actually, this distance is

$$\mathrm{ls}(f) = 2^{n-2} - \frac{1}{4}\delta_f.$$

Another «distance to» characteristic of Boolean functions was carefully investigated in the paper [1] devoted to the notion of algebraic degeneration. A function $f \in \mathcal{F}_n$ is called *algebraic degenerate* if there are $g \in \mathcal{F}_k$ and a binary $(k \times n)$-matrix $D$, $0 \leqslant k \leqslant n$, such that $f(u) = g(Du)$ for all $u \in V_n$. The $f$'s distance to the set of all algebraic degenerate Boolean functions of $n$ variables is denoted by $\rho(f)$.

One says that a Boolean function $f$ satisfies *the propagation criterion with respect to a vector $a$* if the addition of $a$ to any argument vector results in changing exactly half of $2^n$ function's values. The set of all such vectors $a$ will be denoted by $PC_f$. A Boolean function $f$ is said to satisfy *the propagation criterion of degree $k$* (PC($k$)) if it satisfies the propagation criterion with respect to all vectors $a$ such that $1 \leqslant \mathrm{wt}(a) \leqslant k$. Thus, a function $f \in \mathcal{F}_n$ satisfies PC($k$) iff changing values of any $m$ variables ($1 \leqslant m \leqslant k$) entails the changing of exactly $2^{n-1}$ values of the function $f$. This notion introduced in [7] generalizes the SAC, which is PC(1).

A Boolean function $f$ is called *correlation immune with respect to a vector $a$* if the function $f(u) \oplus \langle a, u \rangle$ is balanced. We will denote by $CI_f$ the set of

all vectors $a$ such that $f$ is correlation immune with respect to $a$. A Boolean function $f$ of $n$ variables is *correlation immune of order* $k$, $1 \leqslant k \leqslant n$, if it is correlation immune with respect to all vectors $a$ such that $1 \leqslant \mathrm{wt}(a) \leqslant k$. This means that $f$'s value is statistically independent of any subset of $k$ variables. The notion of correlation immunity was introduced in [8]. It is aimed at measuring resistance to the correlation cryptanalysis method. A Boolean function with high-order correlation immunity is more preferable, for example, in a stream cipher as a combining function for linear feedback shift registers. We'll denote the maximum value of $k$ such that $f$ is correlation immune of order $k$ by $\mathrm{cor}(f)$.

Finally, a Boolean function of $n$ variables is called *$k$-resilient* [5] if its value is uniformly distributed on $\{0, 1\}$ when arbitrary set of $k$ variables get fixed values and other $n - k$ variables are independent and uniformly distributed.

### 2.3 $\Delta$-equivalence

In [10] Zhang and Zheng proposed to assess cryptographic quality of Boolean functions by the global avalanche characteristics. These are essentially the two parameters of the autocorrelation function. GAC turned out a very helpful tool. The forthcoming paper of Logachev, Fedorov and Yashchenko introduces a new approach based also on the autocorrelation function.

**Definition 1.** *Two Boolean functions $f$ and $g$ are* $\Delta$-equivalent *if their autocorrelation functions are identical, i. e., $\Delta_f = \Delta_g$. This relation on $\mathcal{F}_n$ will be denoted by the symbol $\overset{\Delta}{\sim}$.*

It is evident that this is indeed an equivalence relation, that partitions the set $\mathcal{F}_n$ into the classes of function with the same autocorrelation. As it will be seen below, the functions in any such class are unified by some other cryptographic properties as well.

An example of $\Delta$-equivalence class is provided by any set of plateaued functions with fixed Walsh spectrum support.

**Proposition 1.** *Let $f, g \in \mathcal{F}_n$ be two plateaued functions such that $\operatorname{supp} W_f = \operatorname{supp} W_g$. Then $f \overset{\Delta}{\sim} g$.*

*Proof.* According to the Proposition conditions and the plateaued functions definition we have $|W_f(u)| = |W_g(u)|$ for all $u \in V_n$.

By a corollary of the well-known cross-correlation theorem (see, for example, [3, Th. 2.3.1]) any function $h$ satisfies

$$\sum_{v \in V_n} \Delta_h(v)(-1)^{\langle u,v \rangle} = W_h^2(u). \qquad (\bigstar)$$

Applying the Fourier transform inversion formula we obtain

$$\Delta_h(u) = \frac{1}{2^n} \sum_{v \in V_n} W_h^2(v)(-1)^{\langle u,v \rangle}.$$

So, autocorrelation is determined by the absolute values of Walsh coefficients unambiguously. Hence, $\Delta_f(u) = \Delta_g(u)$ for all $u \in V_n$. □

In particular, all bent functions form one $\Delta$-equivalence class. However, each class containing an affine function has the cardinality 2, that is, any affine function $f$ has no $\Delta$-equivalent functions but $f$ and $f \oplus 1$.

## 3 Common properties of $\Delta$-equivalent functions

As it could be seen from the above (see subsection 2.2), there is a number of subsets in $V_n$ which are associated with a given Boolean function and characterizes its cryptographic properties. In the following Theorem 1 we list some of such vector sets that are identical for all $\Delta$-equivalent functions.

**Theorem 1.** *For any $f, g \in \mathcal{F}_n$ being $\Delta$-equivalent to each other, i. e., $f \overset{\Delta}{\sim} g$, the following statements are true:*

1. *$f$'s Walsh coefficient $W_f(u)$ is equal to 0 if and only if corresponding Walsh coefficient $W_g(u)$ of $g$ equals 0, so $f$ and $g$ have the same Walsh spectrum support ($\operatorname{supp} W_f = \operatorname{supp} W_g$);*
2. *vector $u$ is a linear structure for $f$ if and only if $u$ is linear structure for $g$, so $f$ and $g$ have the same space of linear structures ($L_f = L_g$);*
3. *$f$ is correlation immune w. r. t. a vector $u$ if and only if $g$ is correlation immune w. r. t. $u$ ($CI_f = CI_g$);*
4. *$f$ satisfies the propagation criterion w. r. t. a vector $u$ if and only if $g$ satisfies the propagation criterion w. r. t. $u$ ($PC_f = PC_g$).*

*Proof.* We'll demonstrate the proof of the «only if» (necessity) implications, as the backward implications can be proven in the «symmetric» way (by substituting $g$ for $f$).

(1) Since $f$ and $g$ have identical autocorrelation ($\Delta_f = \Delta_g$), accordingly to the equation ($\bigstar$) we obtain $W_f^2(u) = W_g^2(u)$ for each $u \in V_n$. So if $W_f(u) = 0$ then $W_g(u) = 0$.

(2) By the condition, $f(v \oplus u) \oplus f(v)$ is constant (as a function of $v$). Therefore $\Delta_f(u) = \sum_{v \in V_n} (-1)^{f(v \oplus u) \oplus f(v)} = \pm 2^n$. So, $\Delta_g(u) = \pm 2^n$, but it is only possible when all the summands $(-1)^{g(v \oplus u) \oplus g(v)}$ have the same sign, that is, if $g'_u(v) = g(v \oplus u) \oplus g(v)$ is constant.

(3) We have $W_f(u) = \sum_{v \in V_n} (-1)^{f(v) \oplus \langle u, v \rangle} = 0$, as for the given $u$ the function $f(v) \oplus \langle u, v \rangle$ is balanced. Hence, by the item (1), $\sum_{v \in V_n} (-1)^{g(v) \oplus \langle u, v \rangle} = W_g(u) = 0$, that is, $g(v) \oplus \langle u, v \rangle$ is a balanced function.

(4) Since the propagation criterion property of an arbitrary function $h$ w. r. t. $u$ is equivalent to the balancedness of the function $h'_u(v) = h(v) \oplus h(u \oplus v)$, the value $\Delta_f(u) = \sum_{v \in V_n} (-1)^{f'_u(v)}$ equals 0 for the given $f$ and $u$. So $\Delta_g(u) = \Delta_f(u) = 0$ and hence $g'_u(v)$ is balanced too. □

The following Theorem presents some of the cryptographic properties which are common for the functions from any given $\Delta$-equivalence class.

**Theorem 2.** *Let for $f, g \in \mathcal{F}_n$ the relation $f \overset{\Delta}{\sim} g$ holds. Then*

1. *$f$ is balanced if and only if $g$ is balanced;*
2. *$f$ has inessential $i$-th variable if and only if $g$ has inessential $i$-th variable;*
3. *$f$ satisfies the strict avalanche criterion (SAC) if and only if $g$ satisfies SAC;*
4. *$f$ satisfies the propagation criterion of degree $k$ if and only if $g$ satisfies the propagation criterion of degree $k$;*
5. *$f$ is correlation immune of order $k$ if and only if $g$ is correlation immune of order $k$;*
6. *$f$ is $k$-resilient function if and only if $g$ is $k$-resilient.*

*Proof.* As above, we omit the proof of the backward (sufficiency) implications.

(1) If $f$ is balanced then $W_f(0^n) = \sum_{v \in V_n} (-1)^{f(v)} = 0$ where $0^n$ is the all zero component vector. According to the item (1) of Theorem 1, $W_g(0^n) = 0$ and so $g$ is balanced.

(2) If $i$-th variable is inessential for $f$ then for all $v \in V_n$ we have $f(v) = f(v \oplus e_i)$ where the vector $e_i$ has no 1-s except on the $i$-th place. In other words, $f$ has the linear structure $e_i$. By the item (2) of Theorem 1 this vector is a linear structure for $g$ as well. So $i$-th variable is inessential for $g$.

(3, 4) This is the direct consequence of the statement (4) of Theorem 1 (SAC being PC(1)).

(5) This is evident from the statement (3) of Theorem 1.

(6) Since a function is $k$-resilient iff it is at the same time balanced and correlation immune of order $k$, we get the statement from (1) and (5). □

The following Theorem 3 provides examples of Boolean function's numerical parameters remaining invariant for all functions in a $\Delta$-equivalent class.

**Theorem 3.** *If $f, g \in \mathcal{F}_n$ and $f \overset{\Delta}{\sim} g$, then $f$ and $g$ have the same*

1. *cardinality of Walsh spectrum support,* $|\operatorname{supp} W_f| = |\operatorname{supp} W_g|$;
2. *nonlinearity,* $\mathrm{nl}(f) = \mathrm{nl}(g)$;
3. *curvature,* $\mathrm{curv}(f) = \mathrm{curv}(g)$;
4. *global avalanche characteristics,* $(\sigma_f, \delta_f) = (\sigma_g, \delta_g)$;
5. *distance to functions with linear structure,* $\mathrm{ls}(f) = \mathrm{ls}(g)$;
6. *distance to algebraic degenerate functions,* $\rho(f) = \rho(g)$;
7. *maximum order of correlation immunity,* $\mathrm{cor}(f) = \mathrm{cor}(g)$.

*Proof.* (1) It follows trivially from the item (1) of Theorem 1.

(2) It is well known that $\mathrm{nl}(h) = 2^{n-1} - \frac{1}{2}\max_{u \in V_n} |W_h(u)|$ for any $h \in \mathcal{F}_n$. From the equation ($\bigstar$) we know that $|W_f(u)| = |W_g(u)|$ for any pair of $\Delta$-equivalent functions $f$ and $g$ and any $u \in V_n$. So $\mathrm{nl}(f) = \mathrm{nl}(g)$.

(3) As above, $|W_f(u)| = |W_g(u)|$ for $\Delta$-equivalent functions $f$ and $g$. By the curvature definition, these functions have equal curvatures.

(4, 5) Since the GAC (and so, the distance to linear structure functions) are defined solely by the values of autocorrelation function, any two functions with identical autocorrelation have the same GAC.

(6) Accordingly to [1, Cor. 1], $\rho(f) = 2^{n-2} - \frac{1}{4}\max_{0 \neq u \in V_n} \Delta_f(u)$. So, this parameter is the same for all $\Delta$-equivalent functions.

(7) It is a trivial consequence of the statement (4) of Theorem 2. $\square$

## 4 Conclusion

The $\Delta$-equivalence have an extremely simple definition in terms of just one function. This concept meets most of known cryptographic properties of Boolean functions, not only those based on the autocorrelation function, but some others having different nature as well. Besides the necessary conditions of $\Delta$-equivalence given in Theorems 1, 2, 3, there is a sufficient condition formulated in Proposition 1. And the fact that all the bent functions form one $\Delta$-equivalence class confirms in some sense the adequacy of such classification.

At the same time, too fine partition of the functions with low nonlinearity, especially affine functions, may look like a redundant differentiation. Moreover, there are cryptographic properties of Boolean function that are not invariant within $\Delta$-equivalence classes. We don't succeed in handling

algebraic properties of Boolean functions like the algebraic immunity, for example.

## Acknowledgements

Author is very grateful to Valeriy Vladimirovich Yashchenko and especially to Oleg Alexeyevich Logachev for suggestions and helpful discussion during the preparion of this paper. Also, author thanks the CTCrypt symposium program chair for the very valuable reviewers' remarks on the paper.

## References

[1] Alekseyev, E. K. *On some nonlinearity mesures of Boolean function.* Applied Discrete mathematics, 2011 2 (12), pp. 5-16. In Russian.

[2] de la Cruz Jimenez, R. A., Kamlovskiy, O. V. *The sum of modules of Walsh coefficients of Boolean functions.* Discrete Math. Appl., 26 (5), pp. 259–272, 2016.

[3] Logachev, O. A., Salnikov, A. A., Yashchenko, V. V. *Boolean Functions in Coding Theory and Cryptography Providence*, Rhode Island American Mathematical Society. 2011. In Russian.

[4] Logachev, O. A., Fedorov, S. N., Yashchenko, V. V. *Boolean functions as points of hypersphere in Euclidean space.* Discrete Math. Appl., 30 (1), pp. 39–55, 2018. In Russian.

[5] Chor, B., Goldreich, O., Håstad, J., Friedman, J., Rudich, S., Smolensky, R. *The bit extraction problem of t-resilient functions (Preliminary version).* 26th Annual Symposium on Foundations of Computer Science (FOCS '85), pp. 396–407. IEEE Computer Society. 1985.

[6] Meier, W., Staffelbach, O. *Nonlinearity criteria for cryptographic functions.* Advances in Cryptology – EUROCRYPT '89. Lecture Notes in Computer Science, vol 434, Springer, pp. 549–562. 1990.

[7] Preneel, B., Van Leekwijck, W. V., Van Linden, L. V., Govaerts, R., Vandewalle, J. *Propagation characteristics of Boolean functions.* Advances in Cryptology — EUROCRYPT '90. Lecture Notes in Computer Science, vol 473, Springer, pp. 161–173. 1991.

[8] Siegenthaler, T. *Correlation-immunity of nonlinear combining functions for cryptographic applications.* IEEE Transactions on Information Theory, 30 (5), pp. 776–780. 1984.

[9] Webster, A. F., Tavares, S. E. *On the design of S-boxes.* Advances in Cryptology – CRYPTO '85. Lecture Notes in Computer Science, 219, Springer, pp. 523–534. 1986.

[10] Zhang X. M., Zheng Y. *GAC – the criterion for global avalanche characteristics of cryptographic functions.* Journal for Universal Computer Science, 1 (5), pp. 316–333. 1995.

# Public key cryptography

# Constructing of Strong Elliptic Curves Suitable for Cryptography Applications

Alexey Nesterenko

National Research University Higher School Of Economics, Moscow, Russia
nesterenko_a_y@mail.ru

**Abstract**

An algorithm for constructing of elliptic curves with special requirements is presented. The set of requirements follows from known attacks which may be applicable to solve the elliptic curve discrete logarithm problem in special cases. The results of practical experiments and parameters of concrete elliptic curves are given.

**Keywords:** cryptography, elliptic curve, discrete logarithm problem, complex multiplication.

## 1 Motivation

Let $p > 3$ is a prime. Consider an elliptic curve given in short Weierstrass form by equation

$$E_{a,b}: \quad y^2 \equiv x^3 + ax + b \pmod{p}, \tag{1}$$

where $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$. Let $P = (x, y)$ is a fixed point on $E_{a,b}$ and $q$ is an order of $P$, i.e

$$[q]P = \underbrace{P + \cdots + P}_{q \text{ times}} = \mathcal{O},$$

where $\mathcal{O}$ is a neutral element of group of elliptic curve points. We suppose that $q$ is a prime and $q|m$, where $m$ is a number of all points on elliptic curve. From Hasse theorem, see [20, Ch. 5], we know, that

$$p + 1 - 2\sqrt{p} < m < p + 1 + 2\sqrt{p}.$$

We can define an elliptic curve discrete logarithm problem (ECDLP) as the problem of searching an integer $k \in \mathbb{F}_q^*$ such that

$$Q = [k]P, \quad \text{and} \quad Q \in \langle P \rangle.$$

For solving this problem, see [4, 23], we can use Pollard's Rho and Lambda methods [14] or parallel algorithm, introduced by van Oorschot and Wiener

[11]. These algorithms have the running time $O(\sqrt{q})$ and can be applied to any elliptic curve.

In special cases we can use the methods with lower running time. When the condition $m = p$ holds we have linear time to solve the ECDLP using methods of Sato, Araki [17], Smart [21] or Semaev [19].

When the multiplicative order of $p$ modulo $q$ is small we can apply MOV-attack, see [8], and solve ECDLP with lower running time using the calculations in multiplicative group of some finite extension of $\mathbb{F}_p$.

In 2016 a new method for solving ECDLP was proposed by Petit, Kosters and Messeng, see [12]. This method based on solving a system of non-linear polinomials over $\mathbb{F}_p$, generated by Semaev's summation polinomials, and can be applied in case when $p-1$ is smooth, i.e. $p-1$ has many small divisors. Nowadays we don't have any practical realizations of this method for some value of $p$, but in the future this can be done.

At the same time Nesterenko presented a new method for solving ECDLP wich running time depends on multiplicative order of secret value $k$ modulo $q$, see [10]. If $r$ is a divisor of $q-1$ then solving ECDLP has running time $O(\sqrt{r}\log q)$, hence when $q-1$ has many small divisors then [10] shows that ECDLP can be efficiently solved for many «weaked» values of $k$. Note that two last methods gives us a situation similar to RSA modulus $N = pq$, where $p$, $q$ needs to be a safe, see [16].

On this basis we can lay down conditions on parameters of elliptic curve $E_{a,b}$ which guarantee us inapplicability of methods reffered above.

**Definition 1.** *Let $0 < \alpha < \beta$ are natural numbers. We would say that elliptic curve $E_{a,b}$ defined by equation (1) is $(\alpha, \beta)$-strong elliptic curve if exists a point $P \in E_{a,b}$ with $|\langle P\rangle| = q$ and the following condition holds.*

1. *$m \neq p$;*
2. *$p$ is safe prime, that means $\frac{p-1}{2}$ is also prime;*
3. *$j(E_{a,b}) \not\equiv 0, 1728 \pmod p$, where $j(E_{a,b}) \equiv 1728\frac{4a^3}{4a^3+27b^2} \pmod p$;*
4. *$q$ is safe prime, that means $\frac{q-1}{2}$ is also prime;*
5. *$2^\alpha < q < 2^\beta$;*
6. *for fixed $B$ the condition $p^t \not\equiv 1 \pmod q$ holds for all $t = 1, 2, \ldots, B$.*

The constants $\alpha, \beta$ and $B$ can be defined with different values. In [7] we can find $\alpha = 254, \beta = 256$ and $B = 31$ or $\alpha = 508, \beta = 512$ and $B = 131$. In [22] we can find another values $\alpha = 224, \beta > \alpha$ and $B = 10^4$.

Also remark that for safe prime $p$ condition $j(E_{a,b}) \not\equiv 0, 1728$ gives us inequality $m \neq p + 1$, see [6, Ch.13].

It's clear that conditions of safety for $p$ and $q$ in definition 1 are supplement to [7]. Furthermore staying on unproved assumptions we can lay down more rigorous conditions given in therms of complex multiplication's theory for elliptic curves, see [6, 20].

**Definition 2.** *Let $h$ is a natural number. We would say that $(\alpha, \beta)$-strong elliptic curve $E_{a,b}$ defined by equation (1) is very $(\alpha, \beta)$-strong elliptic curve if the following conditions holds.*

1. *The class number of the ring $\mathbb{Z}[\sqrt{-\Delta}]$ should be at least $h$, where $End(E_{a,b}) \subseteq \mathbb{Z}[\sqrt{-\Delta}]$ is an endomorphisms ring of elliptic curve $E_{a,b}$.*
2. *The order of twist of elliptic curve $E_{a,b}$ must have a safe prime divisor $r$ where $2^{\alpha} < r < 2^{\beta}$.*

The first condition is based on ideas of technical guideline [22], where we can find an appropriate bound $h = 200$. More impulsive condition present in [3]. Bernstein and Lange says that the fundamental discriminant $\Delta$ of $\mathbb{Z}[\sqrt{-\Delta}]$ should satisfy inequality $|\Delta| > 2^{100}$. This requirement makes a theory of complex multiplication fully inapplicable for solving ECDLP from practical reasons. From this point of view our condition is slightly weaker.

Our second condition based on well know fact that elliptic curve $E_{a,b}$ and its twist are isomorphic over small finite extension of $\mathbb{F}_p$, see [6]. This condition is similar to [2] when $r$ is prime, but not safe.

In the next section we describe the results of our attempts to construct the elliptic curves suitable for definition 1 or definition 2. We give a detailed description of the constructing algorithm for the benefit of rigidity confirmation.

## 2 An algorithm

The construction of elliptic curve can be performed in several ways. The first one is based on SEA-algorithm of counting points on elliptic curve, see [18], for which it is necessary to determine the prime $p$ and random coefficients $a, b$. The generation of random values $a, b$ should continue until the order of elliptic curve does not satisfy to conditions of definition 1 or definition 2.

When we use some pseudo-random function to generate the coefficients $a, b$ with predefined seed value we obtain a deterministic algorithm for constructing pseudo-random elliptic curve. From practical experiments we conclude that probability of successful ending of this algorithm is very small. Therefore we need another deterministic algorithm that will step by step reduce the distance to expected elliptic curve.

Our algorithm is based on theory of complex multiplication when we can define a coefficients of elliptic curve relatively simple, when we know prime modulo $p$ and prime order $q$. The ideas on which the algorithm is based can be found in [4].

From practical reasons we would find a safe prime $p$ near degree of 2. We know many elliptic curves possessing this property, say «E-382» [1], «Curve25519» [2] or «paramsetA» curve from [13]. Similar elliptic curve can be efficiently used in cryptography applications. At the beginning we can construct an elliptic curve in short Weierstrass form (1), where $m = 2q$ and safe prime $q$ satisfying inequalities $2^\alpha < q < p < 2^\beta$. This form of elliptic curve is applicable for evaluation of digital signatures. Later we can reduce this elliptic curve to Montgomery form, see [9], which is more applicable for key agreement protocols and public key encryption.

Also we would search safe prime $p$ satisfying $p \equiv 11 \pmod{12}$. Since we need to find odd prime $p = 2p_1 + 1$, where $p_1 = 2p_2 + 1$ is also odd prime, we immediately have $p \equiv 3 \pmod 4$.

From the other hand if condition $p \equiv 1 \pmod 3$ holds we have equality $p = 3s+1 = 2p_1+1$ for some natural integer $s$ or $3s = 2p_1$. The last equality doesn't hold since 3 and $p_1$ is coprime, hence $p \equiv 2 \pmod 3$.

The algorithm is follows.

1. We start from maximal odd integer $p_0$ for which the conditions
$$p_0 \equiv 11 \pmod{12}, \quad p_0 < 2^\beta$$
holds.
2. For every number in a decreasing sequence $p_n = p_0 - 12n$, where index $n = 0, 1, \ldots$, we use twice the Miller-Rabin test, see [15], and check that $p_n$ is a safe prime.
3. For safe prime $p_n$ we try to solve the Cornaccia's equation
$$4p_n = x^2 + dy^2 \tag{2}$$
for every square free integer $d = 5, 6, 7, 11, \ldots, d_0$, where $h_0$ is an appropriate higher bound, say $d_0 = 10^6$. If the equation (2) is solved then the value of $d$ gives the value of fundamental discriminant of $\mathbb{Z}[\sqrt{-\Delta}]$ since
$$\Delta = \begin{cases} d, & \text{if} \quad d \equiv 1 \pmod 4, \\ 4h, & \text{if} \quad d \equiv 2, 3 \pmod 4. \end{cases}$$

4. If we find suitable $d$ satisfying (2), we immediately calculate

$$m_1 = p + 1 - x, \quad m_2 = p + 1 + x$$

the orders of some elliptic curve $E_{a,b}$ and its twist.

5. Now we must check that the conditions of definitions 1 or 2 hold for given values of $m \in \{m_1, m_2\}$ and $p$. If it's true then we try to construct the coefficients $a, b$ of elliptic curve $E_{a,b}$ using a theory of complex multiplication in the following way.
   (a) Let $\omega \in \mathbb{C}$ and $\{1, \omega\}$ is a basis of some order in $\mathbb{Z}[\sqrt{-\Delta}]$. Find a value of $j(\omega)$ where $j$ is a modular function.
   (b) Find a polynomial $H_d(x) \in \mathbb{Z}[x]$ for which the equality

$$H_d(j(\omega)) = 0$$

holds and degree of polynomial $H_d(x)$ is equal to $h$ — class number of $\mathbb{Z}[\sqrt{-\Delta}]$. Since $j(\omega)$ is an algebraic integer of degree $2h$ we know that every root of $H_d(x)$ has the same degree and generate the Hilbert class field $\mathbb{K}$ which is a maximal unramified extension of $\mathbb{Q}$ of degree $2h$.

When $d$ satisfies the additional conditions it's possible to construct the field $\mathbb{K}$ by means of values of another functions called Weber functions, see [4].

   (c) Every root of $H_d(x)$ modulo $p$ gives a value of $j(E_{a,b})$ which means as $j$-invariant of elliptic curve $E_{a,b}$ defined over $\mathbb{F}_p$. The coefficients of this curve satisfy to equalities

$$\begin{cases} a \equiv 3kc^2 \pmod{p}, \\ b \equiv 2kc^3 \pmod{p}, \end{cases}$$

where

$$k \equiv \frac{j(E_{a,b})}{1728 - j(E_{a,b})} \pmod{p} \tag{3}$$

and $c$ is quadratic residue modulo $p$. The order of constructed elliptic curve $E_{a,b}$ is $m_1$ or $m_2$. When $c$ is non-residue modulo $p$ the coefficients $a, b$ determine the twist of elliptic curve which order is $m_2$ or $m_1$ respectively.

Since value $a = -3$ is useful for effective calculations on elliptic curve we need to check the equality $\left(\frac{-k}{p}\right) = 1$, where $\left(\frac{\cdot}{\cdot}\right)$ is a Legandre symbol.

   (d) At last we need to choose a random point $P$ and check the order of

the constructed elliptic curve.

6. After constructing the elliptic curve $E_{a,b}$ we need to give a correct proof that $p, q$ is a safe primes.

The basic complexity of presented algorithm lies on solution of equation (2) for $10^6$ possible values of $d$. For decreasing the complexity and satisfying the definition 2 we can try only values of $d$ which give us large class number of $\mathbb{Z}[\sqrt{-\Delta}]$.

## 3 Results of practical experiments

For $\alpha = 254$ and $\beta = 256$ the first elliptic curve which satisfy the definition 1 was found for

$$p = 2^{256} - 1593437.$$

The value $p_1 = \frac{p-1}{2}$ is equal to

$$p_1 = 57896044618658097711785492504343953926634992332820282019728792003956564023249$$

and also prime. To prove that we can factor $p_1 - 1$ to product

$$p - 1 = 2^4 \times 7 \times 919 \times 16454377 \times 9489318407 \times 108549876105863 \times \\ \times 296429175913041139 \times 111956320988599655568967$$

and check that inequality $3^{\frac{p_1-1}{q_i}} \not\equiv 1 \pmod{p_1}$ holds for every prime $q_i | p_1 - 1$. Since $3^{p_1-1} \equiv 1 \pmod{p_1}$ we conclude, due to Lukas theorem, see [5, Ch. 4] that $p_1$ is a prime and 3 is a primitive root modulo $p_1$.

In the same manner we easily prove that $p = 2p_1 + 1$ is a prime and 2 is a primitive root modulo $p$. Solving the equation (2) we find

$$\begin{aligned} d &= 2362, \\ x &= 4990853564168029116837668019960636060 58, \\ y &= 9520292600215848112600803911811223594. \end{aligned}$$

This give us $m = p + 1 + x = 2q$ where

$$q = 57896044618658097711785492504343953926884535011028683475570675404954595826279.$$

Define $q_1 = \frac{q-1}{2}$. Since $q_1$ is odd we find the factorization

$$q_1 - 1 = 2 \times 73 \times 2383 \times 310111 \times 4064012927811943 \times \\ \times 66019122610212984465443141905015302611602361770967$$

and check that $q_1$ is a prime and 2 is a primitive root modulo $q_1$, and $q$ is a safe prime and 7 is a primitive root modulo $q$. Also we can check that multiplicative order of $p$ modulo $q$ is equal to prime $q_1$.

For given $d = 2363$ the fundamental discriminant $\Delta = 9448$, class number of $\mathbb{Z}[\sqrt{-\Delta}]$ is equal to 26 and founded pair of safe primes $p, q$ satisfy to definition 1 and not satisfy to definition 2.

Continuing on, we find a polynomial $H_{2363}(x) \in \mathbb{Z}[x]$ whose root modulo $p$ gives us an invariant of strong elliptic curve

$$E_{a,b}: \quad y^2 \equiv x^3 + x + b \pmod{p},$$

where $a = 1$,

$$b = 57606493662099144146364116733561312117718233732249793938442955956836916404394,$$

and point $P = (4, y)$ with $|\langle P \rangle| = q$ and

$$y = 21773190679040975288323797591449072754840507123874447898008453120896624694078.$$

Note that for this elliptic curve we cannot choose the coefficient $a = -3$. For every root $j(E_{a,b})$ of $H_{2363}(x)$ modulo $p$ we check that pair $a = -3$, $b = 2kc^3$, where $c^2 \equiv -k \pmod{p}$ and $k$ defined by (3), gives a twist of elliptic curve, whose order is not equal to $2q$. We need to construct elliptic curve for another safe prime $p$ if the condition $a = -3$ exactly needs.

## 4 Conclusion

During numerical evaluations we found several strong elliptic curves for fixed values $\alpha = 254$ and $\beta = 256$. Some of this curves has very large fundamental discriminant $\Delta$ but neither one has safe prime divisor for twist order. We hope that this white spot will be colored in the nearest future.

## References


[1] Aranha D., Barreto P., Pereira G., Ricardini J. *A note on high-security general-purpose elliptic curves.* Available from https://eprint.iacr.org/2013/647.pdf.

[2] Bernstein D. *Curve25519: New Diffie-Hellman speed records* // M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, editors, Public Key Cryptography – PKC 2006, volume 3958 of Lecture Notes in Computer Science, pages 207–228. Springer, 2006.

[3] Bernstein D., Lange T. *SafeCurves: choosing safe curves for elliptic-curve cryptography.* Available from `https://safecurves.cr.yp.to`, accessed February 18, 2018.

[4] Blake I., Seroussi G., Smart N. *Elliptic Curves In Cryptography.* – London Mathematical Society Lecture Notes. – Vol. 265 – Cambridge University Press. – 1999.

[5] Crandal R., Pomerance C. *Prime numbers: a computational perspective.* – 2nd Edition. – Springer. – 2005.

[6] Husemöller D. *Elliptic Curves.* – 2nd Edition. – Springer. – 2004.

[7] Information technology. Cryptographic data security. Signature and verification processes of [electronic] digital signature, GOST R 34.10-2012, Federal Agency on Technical Regulating and Metrology, 2012.

[8] Menezes A., Vanstone S., Okamoto T. *Reducing elliptic curve logarithms to logarithms in a finite field* // Proc. 23rd ACM Symp. Theory of Computing. – 1991. – pp. 80–89.

[9] Montgomery P.L. *Speeding The Pollard and Elliptic Curve Methods of Factorization* // Math. of Comp. – 1987. – Vol. 48. – No. 177. – pp. 243-267.

[10] Nesterenko A. Yu. *Some remarks on the elliptic curve discrete logarithm problem* // Mathematical Aspects of Cryptography. – No. 2. – Vol. 7. – 2016. – pp. 115-120.

[11] van Oorschot P.C., Wiener M.J. *Parallel collision search with cryptanalytic applications* // Journal of Cryptology. – Vol.12. – No. 1. – Jan. 1999. – pp. 1-28.

[12] Petit C., Kosters M., Messeng A. *Algebraic Approaches for the Elliptic Curve Discrete Logarithm Problem over Prime Fields*. In: Cheng CM., Chung KM., Persiano G., Yang BY. (eds) Public-Key Cryptography – PKC 2016. PKC 2016. Lecture Notes in Computer Science, vol 9615. Springer, Berlin, Heidelberg. 2016.

[13] Popov V., Kurepkin I., Leontiev S. *Additional Cryptographic Algorithms for Use with GOST 28147-89, GOST R 34.10-94, GOST R 34.10-2001, and GOST R 34.11-94 Algorithms*. – RFC4357. – 2006.

[14] Pollard J.M. *Monte Carlo methods for index computation (*mod *p) //* Mathematics Of Computation. – No. 143. – Vol. 32. — 1978. – pp. 918-924.

[15] Rabin M.O. *Probabilistic Algorithm for Testing Primality //* Journal of Number Theory. – No. 1. – Vol. 12. – 1980. – pp. 128-138.

[16] Rivest R.L., Silverman R.D. *Are «Strong» primes needed for RSA?* – 1999. – 23pp.

[17] Satoh T., Araki K. *Fermat quotients and the polynomial time discrete log algorithm for anomalous curves //* Comm. Math. Univ. Sancti Pauli. – Vol. 47. – 1998. – pp. 81-92.

[18] Schoof, R. *Counting points on elliptic curves over finite fields //* Journal de théorie des nombres de Bordeaux. – No.1 – Vol. 7. – 1995. – pp. 219-254.

[19] Semaev I. *Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p //* Mathematics of Computation. – No. 221. — Vol. 67. – 1998. – pp. 353-356.

[20] Silverman J.H. *The Arithmetic of Elliptic Curves*. – Springer-Verlag. – 1986. – 400 p.

[21] Smart N. *The discrete logarithm problem on elliptic curves of trace one //* Journal of Cryptology. – Vol. 12. – 1999. – pp. 193–196.

[22] Technical Guideline TR-03111. *Elliptic curve cryptography. German Federal Office for Information Security*. – February 14, 2007.

[23] Teske E. *Square-root algorithms for the discrete logarithm problem (a survey) //* Public-key cryptography and computational number theory (Warsaw, 2000). – Berlin. – 2001. – pp. 283-301.

# Considering Two MAC under SIG Variants of the Basic SIGMA Protocol

Trieu Quang Phong

Institute of Cryptography Science and Technology,
Government Information Security Committee, Hanoi, Vietnam
phongtrieu53@gmail.com

**Abstract**

The basic SIGMA protocol is one of the authenticated Diffie-Hellman key exchange protocol based on the «sig-and-mac» mechanism and can be used in the IPsec standards. This protocol was proved to be secure in a variant of the Canetti-Krawczyk (pre-specified peer) model where peer identities are not necessarily known or disclosed from the start of the protocol, namely the Canetti-Krawczyk «post-specified peer» model. In this paper, we will consider two variants of the basic SIGMA protocol in which the $MAC$ tag is not sent separately but rather it is computed under the signature operation. As a consequence, these variants are both secure in the Canetti-Krawczyk «post-specified peer» model.

**Keywords:** SIGMA protocol, M-SIGMA protocol, M1-SIGMA protocol, the Canetti-Krawczyk «post-specified peer» model.

## 1 Introduction

The Internet Key-Exchange (IKE) protocols are the core components to ensure Internet security. Thus, there are many publications on design and analyse of the (secure) IKE protocol (such as IKEv1 [1], IKEv2 [8], OPTLS [9]). IKE provides several key exchange mechanisms that support Diffie-Hellman exchanges but differ in the way *authentication* is provided. A common mechanism for providing authentication in IKE protocols is to use the signature, and SIGMA [4] is a family of IKE protocol that uses such a mechanism.

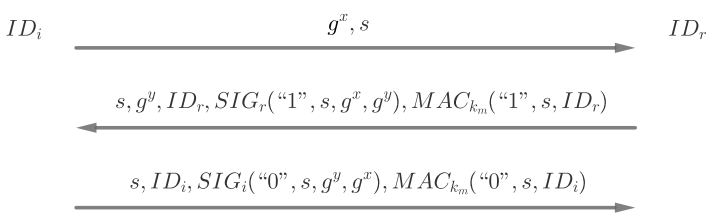



Figure 1: The basic SIGMA protocol

In [3], R. Canetti and H. Krawczyk presented a security proof for the basic SIGMA protocol [(1)] in a variant of the Canetti-Krawczyk (pre-specified peer) model [2] where peer identities are not necessarily known or disclosed from the start of the protocol. This variant is known as the Canetti-Krawczyk «post-specified peer» model, and is a common partial setting, which includes the case of IKE and other protocols that provide confidentiality of identities over the network. (Some other key exchange protocols that are also considered in the «post-specified peer» model are DIKE[7], GC-KKN[6],...)

Besides the basic SIGMA protocol, [3] also considered the security of several variants of the basic SIGMA protocol. One of those variants is designed with IKE signature-mode in which the $MAC$ tag is not sent separately but rather it is computed under the signature operation. In more detail, in the response message of this variant, the responder does not send $SIG_r(\text{"1"}, s, g^x, g^y)$, $MAC_{k_m}(\text{"1"}, s, ID_r)$ as in the basic SIGMA protocol, but rather sends his signature $SIG_r(MAC_{k_m}(s, g^x, g^y, ID_r))$. Similarly, the final message is of the form $SIG_i(MAC_{k_m}(s, g^y, g^x, ID_i))$.

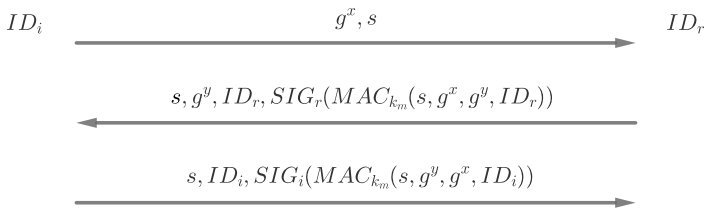



Figure 2: The variant in [3] of the basic SIGMA protocol that put the $MAC$ under the signature

According to [3], the reason for this modification is twofold: to save the extra space taken by the $MAC$ tag and to provide a message format consistent with other authentication modes of IKE. Also, [3] claimed that the security of this variant is essentially analyzed in the same extent as for the basic SIGMA protocol. However, there is no proof for the security of this variant. Hence, we are interested in studying the security of this variant.

Unfortunately, we see that the analysis of this variant is quite complex, and we then do not obtain a complete proof for its security, even though R. Canetti and H. Krawczyk claimed that its security can be obtained by applying the analysis and the following result (Lemma 17 in [3]).

*«If $SIG$ is a secure signature scheme and $MAC$ a secure message authentication function then it is infeasible for an attacker to find differ-*

[(1)] The basic SIGMA protocol is illustrated in Figure 1 in which $ID_I$ and $ID_r$ are the initiator and the responder; $s$ is a session identifier. The values $x$ and $y$ are the ephemeral secrete keys of $ID_i$ and $ID_r$, respectively; $g^x$ and $g^y$ are the ephemeral public keys of $ID_i$ and $ID_r$, respectively. $SIG_i$ and $SIG_r$ denote the signature of $ID_i$ and $ID_r$, respectively. The value $MAC_{k_m}$ is produced with $k_m = PRF_{g^{xy}}(1)$ where $PRF$ is a pseudorandom function family. The session key $k_s$ is computed as $k_s = PRF_{g^{xy}}(0)$.

*ent messages $M$ and $M'$ such that for a randomly chosen secret MAC-key $k_m$ the attacker an compute $SIG(MAC_{k_m}(M'))$ even after seeing $SIG(MAC_{k_m}(M))$.»*

They also said that all the arguments in the proof of Basic SIGMA that use the unforgeability of signatures remain valid in this case by using the above result. However, in our opinion, there are two issues when applying the above result to analyze the first requirement[(2)] of security definition in the Canetti-Krawczyk «post-specified peer» model for the protocol in Figure 2:

- By a «man-in-the-middle» attack, an attacker may send $g^{x'}$ to $ID_r$ instead of $g^x$ from $ID_i$. Therefore, the response of $ID_r$ is $SIG_r(MAC_{k_m^1}(s, g^{x'}, g^y, ID_r))$ where $k_m^1$ is derived from $(g^{x'})^y$. However, we have no (formal) arguments to guarantee that the attacker can find $y'$ such that $MAC_{k_m^1}(s, g^{x'}, g^y, ID_r) = MAC_{k_m^2}(s, g^x, g^{y'}, ID_r)$ and $MAC_{k_m^2}(s, g^{y'}, g^x, ID_i) = MAC_{k_m^1}(s, g^y, g^{x'}, ID_i)$ with only negligible probability (where $k_m^2$ is derived from $(g^{y'})^x$). This implies that there are no (formal) arguments to guarantee that the attacker can violate the first requirement of security definition in the Canetti-Krawczyk «post-specified peer» model with only negligible probability. Moreover, we note that Lemma 17 in [3] cannot be applied in this case, because there is only one MAC-key in the statement of that lemma.
- The condition in Lemma 17 in [3] is that the MAC-key is a randomly chosen secret key and the attacker cannot chose it[(3)]. Howerver, in a «man-in-the-middle» attack, the attacker knows the MAC-key of $ID_r$ and therefore Lemma 17 in [3] cannot be applied in this case.

However, we found out two other variants of the basic SIGMA protocol that the $MAC$ tag is also put under the signature and realize that they can be proved secure in the same way as the basic SIGMA protocol. These two variants will be described in Session 2.

## 2 The description of variants

### 2.1 Variant 1

In this section, we consider a modification of the basic SIGMA protocol in which the responder does not send $SIG_r(\text{"1"}, s, g^x, g^y)$, $MAC_{k_m}(\text{"1"}, s, ID_r)$ as in the basic SIGMA protocol, but rather sends his

[(2)]This requirement guarantee that two honest parties who complete matching sessions compute the same session key (except with negligible probability).

[(3)]R. Canetti and H. Krawczyk have noted that if the attacker can chose the MAC-key, Lemma 17 in [3] would not hold.

signature $SIG_r(\text{"1"}, s, g^x, g^y, MAC_{k_m}(s, ID_r))$. Similarly, the final message is of the form $SIG_i(\text{"0"}, s, g^x, g^y, MAC_{k_m}(s, ID_i))$. Also, the $MAC$ key $k_m$ and the session key $k_s$ of this protocol are both computed in the same way as the basic SIGMA protocol. In particular, $k_m = PRF_{g^{xy}}(1)$ and $k_s = PRF_{g^{xy}}(1)$, where $PRF$ is a pseudo-random function. For convenience, this variant will be called by M-SIGMA.

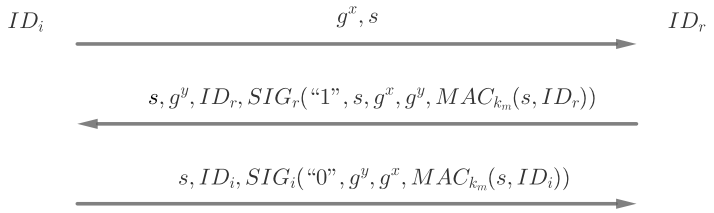



Figure 3: The M-SIGMA protocol

Essentially, M-SIGMA also provides two good properties as the protocol in Figure 2, that is: to save the extra space taken by the $MAC$ tag and to provide a message format consistent with other authentication modes of IKE. Besides, this protocol can be analyzed easier than the protocol in Figure 2.

## 2.2 Variant 2

The second variant we will discuss is identical with M-SIGMA, except that the component $s$ in the signed message will be eliminated. That is, the responder will send $SIG_r(\text{"1"}, g^x, g^y, MAC_{k_m}(s, ID_r))$ in the second message flow, and the initiator will send $SIG_i(\text{"0"}, s, g^y, g^x, MAC_{k_m}(s, ID_i))$ in the finish message flow. We will denote this variant by M1-SIGMA.

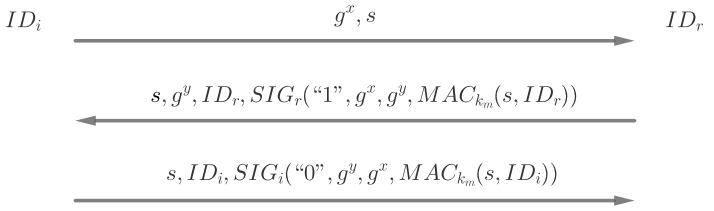



Figure 4: The M1-SIGMA protocol

Although M1-SIGMA is a slight modification of M-SIGMA, the analysis of M-SIGMA should be significantly changed to apply for M1-SIGMA. In Section 4, we will present our analysis of these two protocols in more detail.

# 3 The security model

The overviews of the Canetti-Krawczyk pre- and post-specified peer models for key agreement and the associated security definitions are provided in [5]. For full details and further explanations refer to [3] and [2].

**Pre-specified peer model.** Communications take place in a multi-party system, where the parties are identified by $\widehat{A}$, $\widehat{B}$, $\widehat{C}$,... At any given point in time, a party may be engaged in multiple instances of the protocol, each called a session. A session is created at $\widehat{A}$ via a message containing at least three parameters $(\widehat{A}, \widehat{B}, s)$, where $\widehat{A}$ is the session's owner, $\widehat{B}$ is the intended peer, and $s$ is a number that is unique among all sessions owned by $\widehat{A}$. ($\widehat{A}$ uses $s$ to direct incoming messages to the appropriate session within $\widehat{A}$.) Once created, a session is said to be active and maintains a session state where session-specific short-lived data such as an ephemeral private key is stored. The session processes incoming messages and produces outgoing messages. A session may abort without producing a session key, or may complete by accepting a session key and erasing its session state.

The attacker $\mathcal{M}$, modeled as a probabilistic Turing machine, controls all communications between parties as well as the activation of sessions. In order to model the possible leakage of secret information, $\mathcal{M}$ is allowed to issue the following queries to parties:

- **SessionStateReveal:** $\mathcal{M}$ learns the contents of the session state for a (not yet completed) session of its choosing. The session can no longer be activated and stops producing output.
- **Expire:** $\mathcal{M}$ directs a completed session to delete its session key.
- **SessionKeyReveal**: $\mathcal{M}$ learns the session key held by a (completed but unexpired) session of its choosing.
- **Corrupt**: $\mathcal{M}$ learns all the secret information held by a party of its choosing, including the party's static private key, all session states, and all session keys. The party can no longer be activated and stops producing output.

The attacker's goal is to distinguish a session key from a random key. Obviously, the attacker should not be allowed to learn the session key by trivial means, for example by asking for the session key via a SessionKeyReveal query. To this end, a session $(\widehat{A}, \widehat{B}, s)$ is said to be locally exposed if $\mathcal{M}$ issued a SessionStateReveal or SessionKeyReveal query to that session, or if $\mathcal{M}$ issued a Corrupt query to $\widehat{A}$ before the session expired (this includes the case in which $\widehat{A}$ is corrupted before the session is created). Moreover, the session $(\widehat{B}, \widehat{A}, s)$ is defined to be *matching* to the session $(\widehat{A}, \widehat{B}, s)$, and $(\widehat{A}, \widehat{B}, s)$ is said to be *unexposed* if neither this session nor its matching session are locally exposed. Now, $\mathcal{M}$ selects a session that is completed, unexpired, and unexposed, and issues a special Test query to that session. ($\mathcal{M}$ is not allowed to issue the Test query more than once.) In response, $\mathcal{M}$ is given with equal probability either the session key held by the test session or a random key.

$\mathcal{M}$ can continue to issue queries, however must ensure that the test session remains unexposed. Finally, $\mathcal{M}$ is said to win its distinguishing game (and thereby break the protocol) if it guesses correctly whether the key is random or not with success probability significantly greater than 1/2.

**Definition 1.** ***[2]*** *A key agreement protocol is said to be secure (in the Canetti-Krawczyk pre-specified peer model) if:*

*(i) Uncorrupted parties who complete matching sessions compute the same session key (except with negligible probability).*

*(ii) There is no attacker $\mathcal{M}$ who wins the distinguishing game.*

**Post-specified peer model.** The Canetti-Krawczyk post-specified peer model and associated security definition [3] are essentially the same as in the Canetti-Krawczyk pre-specified model, but there are two important differences.

First, a session at $\widehat{A}$ is created via a message containing (at least) three parameters $(\widehat{A}, \widehat{d}, s)$, where $\widehat{d}$ is a destination address to which outgoing messages should be delivered. That is, party $\widehat{A}$ does not know the identifier of its peer when it starts the session. During the course of the protocol run, $\widehat{A}$ learns the (alleged) identifier $\widehat{B}$ of the communicating party; this party is referred to as $\widehat{A}$'s peer for that session.

Second, the definition of a matching session is different. The new matching session definition is stated as follows.

**Definition 2.** ***[2]*** *Let $(\widehat{A}, s)$ be a session that has completed with public output $(\widehat{A}, \widehat{B}, s)$. Then a session $(\widehat{B}, s)$ is said to be matching to $(\widehat{A}, s)$ if either:*

*(i) $(\widehat{B}, s)$ has not yet completed; or*

*(ii) $(\widehat{B}, s)$ has completed and its peer is $\widehat{A}$.*

Condition $(i)$ is necessary because the incomplete session $(\widehat{B}, s)$ may not yet have determined its peer and hence could have been communicating with $(\widehat{A}, s)$, in which case exposure of $(\widehat{B}, s)$ could possibly reveal non-trivial information about the session key held by $(\widehat{A}, s)$.

**Definition 3.** ***[2]*** *The security definition in the Canetti-Krawczyk post-specified peer model is defined identically as in Definition 1 but with the notion of matching sessions re-formulated via Definition 2.*

## 4 The security of basic SIGMA

**Theorem 1.** ***[2]*** *Assuming DDH and the security of the underlying cryptographic functions $SIG$, $MAC$, $PRF$, the basic SIGMA protocol is secure in*

*the Canetti-Krawczyk post-specified model.*

Proving this theorem in [3] requires that the basic SIGMA protocol satisfies the two properties in Definition 3:

**P1.** If two uncorrupted parties $ID_i$ and $ID_r$ complete matching sessions ($(ID_i, s, ID_r)$ and $(ID_r, s, ID_i)$, respectively) under the basic SIGMA protocol then, except for a negligible probability, the session key output in these sessions is the same.

**P2.** No efficient attacker on the basic SIGMA protocol can distinguish a real response to the test-session query from a random response with non-negligible advantage. More precisely, if for a given attacker $\mathcal{M}$ we define:

- $P_{REAL}(\mathcal{M}) =$
  $= Pr(\mathcal{M}$ `outputs 1 when given the real test session key`$)$
- $P_{RAND}(\mathcal{M}) =$
  $Pr(\mathcal{M}$ `outputs 1 when given the random test session key`$)$

then we need to prove that for any attacker $\mathcal{M}$: $|P_{REAL}(\mathcal{M}) - P_{RAND}(\mathcal{M})|$ is negligible.

In [3], the property P1 of basic SIGMA protocol is based on the security of $SIG$ function; and the proof of property P2 for basic SIGMA protocol use the following notions.

**The simulators.** A simulator $S = S(\mathcal{M})$ is defined on parameters $n$ (number of parties) and a given attacker $\mathcal{M}$, simulates a run of (basic SIGMA) protocol against attacker $\mathcal{M}$. Simulator $S$ starts by choosing the initialization information for each of the $n$ parties (private signature keys and their corresponding public verification keys) and execute the queries of $\mathcal{M}$.

We recall several variants of the above simulator $S$ (in [3]), which is generally denoted by $\widehat{S}$ and is similar to $S$ except for the following differences.

1. Let $l$ be an a-priori upper bound of the number of that $\mathcal{M}$ initiates during its run with $n$ parties. At the beginning, $\widehat{S}$ chooses the following values: a number $t$ chosen uniformly in $[1, l]$, an identity $R_0$ randomly chosen among the identities of $n$ parties; two elements $x, y \in \mathbb{Z}_q$, and two values $k_m$ and $k_s$ (depended on the variants $\widehat{S}$) of the same length as the output of the $PRF$ functions.
2. $\widehat{S}$ performs a usual simulation of $\mathcal{M}$ like $S$ does except that it takes two types of special actions:
   (a) the actions related to the $t$-th session initiated by $\mathcal{M}$ as described in step 3 below; and
   (b) stopping its run upon the occurrence of any of the *«abort events»* that is listed below, in which case $\widehat{S}$ stops with output 0.

3. Let the $t$-th session is initiated by $\mathcal{M}$ be $(I_0, s_0)$. The following actions take place as long as an abort event does not happen. The start message of session $(I_0, s_0)$ is generated by $\widehat{S}$ using the value $x$ chosen in step 1 (i.e., the start message output by $(I_0, s_0)$ is $s_0, g^x$). In case that session $(R_0, s_0)$ is activated by $\mathcal{M}$ with $R_0$ as responder then $\widehat{S}$ outputs a response message on behalf of $R_0$ using the exponent $g^y$ (with $y$ chosen in step 1). Moreover, the MAC computation for this message uses the key $k_m$ chosen by $\widehat{S}$ in step 1. If any of the sessions $(I_0, s_0)$ or $(R_0, s_0)$ complete then the secret session key is set to $k_s$ as chosen by $\widehat{S}$ in step 1.
4. If $\mathcal{M}$ chooses $(I_0, s_0)$ or $(R_0, s_0)$ as its test session then the response to the test query by $\widehat{S}$ is $k_s$.
5. If $\mathcal{M}$ ends its run (without $\widehat{S}$ having aborted) then $\widehat{S}$ outputs the same bit as $\mathcal{M}$ outputs.

**The $\widehat{S}$ variants.** The five variants $\widehat{S}$ listed below (which differ by the way $k_s$ and $k_m$ are defined) are described in [3].

$\widehat{S}_{REAL}$: $k_s \leftarrow PRF_{g^{xy}}(0)$, $k_m \leftarrow PRF_{g^{xy}}(1)$
$\widehat{S}_{RPPF}$: $k_s \leftarrow PRF_k(0)$, $k_m \leftarrow PRF_k(1)$, $k \leftarrow random()$
$\widehat{S}_{ALLR}$: $k_s \leftarrow random()$, $k_m \leftarrow random()$
$\widehat{S}_{HYBR}$: $k_s \leftarrow random()$, $k_m \leftarrow PRF_k(1)$, $k \leftarrow random()$
$\widehat{S}_{RAND}$: $k_s \leftarrow random()$, $k_m \leftarrow PRF_{g^{xy}}(1)$

**Abort event.** If any of the following events happen $\widehat{S}$ stops its run and outputs 0 (recall that we denote by $(I_0; s_0)$ the $t$-th session initiated by $\mathcal{M}$, and by $R_0$ the identity randomly chosen by $\widehat{S}$ in step 1 of its run):

- $\mathcal{M}$ corrupts $I_0$ or $R_0$ before $(I_0, s_0)$ is complete.
- $\mathcal{M}$ issues a state-reveal query against $(I_0, s_0)$ or $(R_0, s_0)$.
- Session $(R_0, s_0)$ is initiated as responder before $(I_0, s_0)$ sent its start message; or $(R_0, s_0)$ is initiated as responder with a start message containing a DH exponent which is different than the DH exponent in the start message output by $(I_0, s_0)$.
- The response message received by $(I_0, s_0)$ arrives before $(R_0, s_0)$ was activated as responder, or this response message has a different DH exponent than the DH exponent appearing in the response message output by session $(R_0, s_0)$.
- Session $(I_0, s_0)$ aborts.
- $\mathcal{M}$ chooses a test session other than $(I_0, s_0)$ or $(R_0, s_0)$, or it chooses one of these but the session completes with a peer different than $R_0$, $I_0$, respectively.
- $\mathcal{M}$ completes the game without having chosen a test session, or $\mathcal{M}$ stops

before having initiated $t$ sessions $\mathcal{M}$.

**GUESS event.** Let $\widehat{S}$ be one of the simulators defined above and $\mathcal{M}$ be an attacker. We say that a guess event happens in a run of $\widehat{S}(\mathcal{M})$ if the following conditions are satisfied:

- $\mathcal{M}$ initiates at least $t$ sessions in this run where $t$ is the parameter chosen by $\widehat{S}$ in step 1 of its run (we denote by $I_0$ the initiator of this session and by $s_0$ the session id);
- if $R_0$ denotes the random chosen party by in step 1 of its run $\widehat{S}$
  - $\mathcal{M}$ chooses $(I_0, s_0)$ as its test session and this session completes with peer $R_0$; or
  - $\mathcal{M}$ chooses $(R_0, s_0)$ as its test session and this session completes with peer $I_0$.

## 5 Our security analysis

In this section, we will analyze the security of two variants M-SIGMA and M1-SIGMA that based on the security analysis of basic SIGMA protocol in [3]. We note that the notions used in our analysis for two variants M-SIGMA and M1-SIGMA is the same as in the analysis for basic SIGMA protocol in [3].

### 5.1 The M-SIGMA protocol

In this section, we will show that M-SIGMA satisfies the two requirements of Definition 3 under the DDH assumption and the security of the underlying cryptographic functions $SIG$, $MAC$, $PRF$.

**M-SIGMA satisfies the first security requirement.** The idea of proving this property is that if an attacker $\mathcal{M}$ can make two parties $ID_i$ and $ID_r$ complete matching sessions $((ID_i, ID_r, s)$ and $(ID_r, ID_i, s)$, respectively) but they have different keys, then $\mathcal{M}$ must alter either $g^x$ (in the first message sent from $ID_i$ to $ID_r$) or $g^y$ (in the response message sent from $ID_r$ to $ID_i$) by $g^{x'}$ or $g^{y'}$. However, in order to $ID_i$ or $ID_r$ accept the message that he receives is legitimate, $\mathcal{M}$ must be able to generate a signature on the message containing $g^{x'}$ or $g^{y'}$, and the value $s$ as the session id. This implies that the security assumption of SIG function is violated. Hence, the basic SIGMA protocol satisfies the first requirement of Definition 3. In a similar way, M-SIGMA also achieves this property.

**M-SIGMA satisfies the second security requirement.** We prove that the property P2 of the basic SIGMA protocol is still valid for M-SIGMA.

**Proposition 1.** *For all attackers $\mathcal{M}$ on M-SIGMA, the following holds except for negligible probability.*

*(a) Consider a regular run by $\mathcal{M}$ in which $\mathcal{M}$ chooses a test session with output $(\widehat{A}, \widehat{B}, s)$ where $\widehat{A}$ is the initiator. Then:*
- *(1) $\widehat{A}$ and $\widehat{B}$ are never corrupted before expiration of the test session.*
- *(2) Sessions $(\widehat{A}, s)$ and $(\widehat{B}, s)$ are never revealed by $\mathcal{M}$.*
- *(3) $(\widehat{B}, s)$ is initiated as responder with the start message sent by $(\widehat{A}, s)$.*
- *(4) $(\widehat{A}, s)$ receives a response message after $(\widehat{B}, s)$ was activated as responder, and this message carries the same DH exponent as in the response message output by $(\widehat{B}, s)$.*
- *(5) Session $(\widehat{A}, s)$ does not abort.*

*(b) Consider a regular run by $\mathcal{M}$ in which $\mathcal{M}$ chooses a test session with output $(\widehat{B}, \widehat{A}, s)$ where $\widehat{B}$ is the responder. Then:*
- *(1) $\widehat{A}$ and $\widehat{B}$ are never corrupted before expiration of the test session.*
- *(2) Sessions $(\widehat{A}, s)$ and $(\widehat{B}, s)$ are never revealed by $\mathcal{M}$.*
- *(3) $(\widehat{B}, s)$ is initiated as responder with the start message sent by $(\widehat{A}, s)$.*
- *(4) $(\widehat{A}, s)$ receives a response message after $(\widehat{B}, s)$ was activated as responder, and this message carries the same DH exponent as in the response message output by $(\widehat{B}, s)$.*
- *(5) Session $(\widehat{A}, s)$ does not abort.*

*Proof.* It is directly applied the proof of Lemma 7 in [3].

Now, we consider the indistinguishability of the above simulators. Here, the proofs of Proposition 6, 7, and 8 are similar to that of Lemma 8, 9, 10 in [3], respectively.

**Proposition 2.** *For all attackers $\mathcal{M}$ on M-SIGMA, the outputs of $\widehat{S}_{RAND}(\mathcal{M})$ and $\widehat{S}_{HYBR}(\mathcal{M})$ are indistinguishable.*

**Proposition 3.** *For any attacker $\mathcal{M}$ on M-SIGMA, the probability of a guess event under a run of $\widehat{S}_{RAND}(\mathcal{M})$ is at least $1/(mn)$, where $m$ is the number of sessions initiated by $\mathcal{M}$ and $n$ is the number of parties in the M-SIGMA protocol.*

**Proposition 4.** *For any attacker $\mathcal{M}$ on M-SIGMA, the probability of a guess event under a run of $\widehat{S}_{HYBR}(\mathcal{M})$ is, up to a negligible difference, the same as the probability of a guess event under a run of $\widehat{S}_{RAND}(\mathcal{M})$.*

The same as Lemma 11 in [3], the following result (Proposition 9) is our central analysis. However, the proof of Proposition 9 is more complex than that of Lemma 11 in [3] and based on the idea of Lemma 17 in [3].

We note that the proof of this proposition is the only difference between the security proofs for the M-SIGMA and basic SIGMA protocols.

**Proposition 5.** *For all attackers $\mathcal{M}$ on M-SIGMA, if a guess event happens under a run of $\widehat{S}_{HYBR}(\mathcal{M})$ then the following properties hold (except for negligible probability):*

*(i) if $(I_0, s_0)$ was chosen by $\mathcal{M}$ as the test session then $(R_0, s_0)$ (either if completed or not) is its matching session;*

*(ii) if $(R_0, s_0)$ was chosen by $\mathcal{M}$ as the test session then $(I_0, s_0)$ is its matching session.*

*Proof.* $(i)$ According to [3] when GUESS event happens, then if $(I_0, s_0)$ was chosen by $\mathcal{M}$ as the test session then this session completes with peer $R_0$. By Definition 2, as long as $(R_0, s_0)$ is incomplete it is matching to $(I_0, s_0)$. If $(R_0, s_0)$ is complete and its outputs is $(R_0, ID, s_0)$ then by Definition 2, $(R_0, s_0)$ matches $(I_0, s_0)$ if and only if $ID = I_0$. Therefore, we want to prove that if $(R_0, s_0)$ completes then $ID = I_0$.

Assume that $(R_0, s_0)$ completes with peer $ID$. This implies that $(R_0, s_0)$ receives the final message flow before it completes. Therefore, $R_0$ receives a signature in the form $SIG_{ID}(\text{"0"}, ..., MAC_{k_m}(s_0, ID))$, where $k_m = PRF_k(1)$ and $k$ is a random key chosen by $\widehat{S}_{HYBR}$ and never provided to the attacker. However, there could have been two signatures of messages containing the $MAC$ values (under the key $k_m$) computed in the protocol, namely, $MAC_{k_m}(s_0, R_0)$ and $MAC_{k_m}(s_0, I_0)$. We consider two following cases:

- The party $ID$ has been never corrupted. If based on two signatures $SIG_{R_0}(\text{"1"}, ..., MAC_{k_m}(s_0, R_0))$ and $SIG_{I_0}(\text{"0"}, ..., MAC_{k_m}(s_0, I_0))$ the attacker $\mathcal{M}$ has non-negligible probability of producing $SIG_{ID}(\text{"0"}, ..., MAC_{k_m}(s_0, ID))$ for $ID \neq I_0$ and $R_0$, there are two cases. The first, if $MAC_{k_m}(s_0, ID) = MAC_{k_m}(s_0, R_0)$ or $MAC_{k_m}(s_0, ID) = MAC_{k_m}(s_0, I_0)$, then we can build, based on $\widehat{S}_{HYBR}$, a forger to $MAC$ function under the key $k_m = PRF_k(1)$, where $k$ is a random independent key. This forger can then be turned into a distinguisher to $PRF$, or into a forger against the $MAC$ function (with a random key). The second, if $MAC_{k_m}(s_0, ID) \neq MAC_{k_m}(s_0, R_0)$ and $MAC_{k_m}(s_0, ID) \neq MAC_{k_m}(s_0, I_0)$ then $ID$ has never produced the signature $SIG_{ID}(\text{"0"}, ..., MAC_{k_m}(s_0, ID))$ because the $MAC$ value $MAC_{k_m}(s_0, ID)$ has never been computed by $\widehat{S}_{HYBR}$, therefore this signature is a forgery signature. Since we assume the $SIG$, $MAC$, $PRF$ function to be secure then the probability that $(R_0, s_0)$ end with peer $ID \neq I_0$ is negligible. (We note that $R_0$ has never generated

the signature $SIG_{R_0}(\text{"0"}, ..., MAC_{k_m}(s_0, R_0))$ before, except negligible probability).

– The party $ID$ was corrupted (so $ID \neq R_0$). In this case, $\mathcal{M}$ can generate ID's signature on any message. However, $\mathcal{M}$ can compute the value $MAC_{k_m}(s_0, ID)$, because of the security assumption of $MAC$ function with only negligible probability. Therefore, $\mathcal{M}$ can generate a signature $SIG_{ID}(\text{"0"}, ..., MAC_{k_m}(s_0, ID))$ with only negligible probability. It implies that in this case the probability that $(R_0, s_0)$ end with peer $ID \neq I_0$ is negligible.

Hence, according to these above cases, we obtain $(R_0, s_0)$ is the matching session of $(I_0, s_0)$.

$(ii)$ It is similar to $(i)$. □

The following propositions are versions (in M-SIGMA) corresponding to Lemma 12, 13, 14, 15, 16 in [3]. In addition, their proofs are quite similar to the proofs of the corresponding lemmas. Therefore, here we only state these propositions without proving.

**Proposition 6.** *Proposition 9 holds for $\widehat{S}_{RAND}$ as well.*

**Proposition 7.** *For all attackers $\mathcal{M}$ on M-SIGMA,*

$$P_{RAND}(\mathcal{M}) = Pr(\widehat{S}_{RAND} \texttt{ outputs 1: GUESS events}).$$

**Proposition 8.** *For all attackers $\mathcal{M}$ on M-SIGMA,*

$$P_{REAL}(\mathcal{M}) = Pr(\widehat{S}_{REAL} \texttt{ outputs 1: GUESS events}).$$

**Proposition 9.** *For all attackers $\mathcal{M}$ on M-SIGMA, the simulators $\widehat{S}_{REAL}(\mathcal{M})$ and $\widehat{S}_{RAND}(\mathcal{M})$ are indistinguishable.*

**Proposition 10.** *The M-SIGMA protocol satisfies the second requirement of Definition 3: for all attacker $\mathcal{M}$ on M-SIGMA, $|P_{REAL}(\mathcal{M}) - P_{RAND}(\mathcal{M})|$ is negligible.*

As a consequence, the M-SIGMA protocol is secure in the Canetti-Krawczyk post-specified peer model.

**Proposition 11.** *Assuming DDH and the security of the underlying cryptographic functions $SIG$, $MAC$, $PRF$, the M-SIGMA protocol is secure in the Canetti-Krawczyk post-specified model.*

### 5.2 The M1-SIGMA protocol

In this section, we will show that M1-SIGMA also satisfies the two requirements of Definition 3 under the DDH assumption and the security of

the underlying cryptographic functions $SIG$, $MAC$, $PRF$.
**M1-SIGMA satisfies the first security requirement.** This property is proved via the following result.

**Proposition 12.** *The M1-SIGMA protocol satisfies the first requirement of Definition 3.*

*Proof.* Let $\mathcal{M}$ be an attacker on M1-SIGMA protocol, and let $ID_i$ and $ID_r$ be identities of two (uncorrupted) parties that complete the matching sessions $(ID_i, ID_r, s)$ and $(ID_r, ID_i, s)$, respectively. We need to prove that regardless of $\mathcal{M}$'s operations both sessions output the same session key. That is, if $g^x$ and $g^y$ are corresponding to the nonce values that $ID_i$ and $ID_r$ generate, then both $ID_i$ and $ID_r$ compute the same DH value $g^{xy}$ (from which the session key $k_s$ is deterministically derived).

Let $u_i$ be the nonce value that $ID_i$ receives in the second message flow and $u_r$ be the nonce value that $ID_r$ receives in the first message flow of session $s$. Therefore, the signature produced by $ID_r$ during session $s$ is $SIG_r(\text{"1"}, u_r, g^y, MAC_{k_m}(s, ID_r))$ (where $k_m$ is derived from the value $u_r^y$), and it is is the only one that $ID_r$ ever produces in session $s$ as the session id. However, the signature that $ID_i$ receives in the response message is $SIG_r(\text{"1"}, g^x, u_i, MAC_{k'_m}(s', ID_r))$ (here, this signature may be generated by $ID_r$ in some session $s' \neq s$ and $k'_m$ is derived from the value $u_i^x$, since $\mathcal{M}$ does not generate any signature by the unforgeability assumption of the $SIG$ function).

There are two ways that $\mathcal{M}$ can obtain the signature of $ID_r$ in session $s'$ to send to $(ID_i, s)$:

- Replaying the signature that $ID_r$ was activated as a responder in some session $s'$ (in which the parties are uncorrupted) and then send to $(ID_i, s)$. However, since the parties are uncorrupted, so the probability that the nonce value of the partner of $(ID_r, s')$ is equal to $g^x$ is negligible.
- Upon $(ID_i, s)$ is activated and it generates the nonce value $g^x$, the attacker $\mathcal{M}$ initiates a (corrupted) party $E$ to send the message flow $s', g^x$, and then activates $ID_r$ as a responder to receive this message (note that $s' \neq s$). After that, $ID_r$ generates a respone message flow $s', g^{y_0}, ID_r, SIG_r(\text{"1"}, g^x, g^{y_0}, MAC_{k'_m}(s', ID_r))$ (in this way, $k'_m$ is derived from $g^{xy_0}$) and sends it to $\mathcal{M}$. Upon receiving this signature, the attacker $\mathcal{M}$ sends the message flow $s, g^{y_0}, ID_r, SIG_r(\text{"1"}, g^x, g^{y_0}, MAC_{k'_m}(s', ID_r))$ to $(ID_i, s)$. If $(ID_i, s)$ accept the validation of this response message flow, then

$MAC_{k'_m}(s, ID_r) = MAC_{k'_m}(s', ID_r)$. (In the other case, we have $SIG_r(\text{"1"}, g^{y_0}, g^x, MAC_{k'_m}(s', ID_r))$ is a signature on the message $\text{"1"}, g^{y_0}, g^x, MAC_{k'_m}(s, ID_r)$ on which $ID_r$ has never generated any signature. So, it contradicts the unforgeability of $SIG$ function). However, if $MAC_{k'_m}(s, ID_r) = MAC_{k'_m}(s', ID_r)$ with non-negligible probability (where $s' \neq s$), it will imply that one of the assumptions (the DDH assumption, and the security assumption of $MAC$, $PRF$ functions) is violated.

Hence, the signature that $\mathcal{M}$ sends to $(ID_i, s)$ is the signature produced by $ID_r$ in the session $s$ (so, $u_i = g^y$ and $u_r = g^x$). This implies that, the DH value computed by $(ID_i, s)$ is $u_i^x = g^{xy}$, while the DH value computed by $(ID_r, s)$ is $u_r^y = g^{xy}$. And therefore both compute the same session key. □

**Remark.** The proof of this property for the basic SIGMA (and M-SIGMA) protocol is only based on the unforgeability assumption of the $SIG$ function. The reason is that, when $s$ is used as a component of the signed message, we know that this signature is unique in session $s$. Hence, the signed message checked by $(ID_i, s)$ and the signed message produced by $ID_r$ are identical. As a consequence, $ID_r$ received the nonce value $g^x$ generated by $ID_i$; and $ID_i$ received the nonce value $g^y$ generated by $ID_r$. However, it is a problem when we omit the component $s$ in the signature (and outside MAC) as in the M1-SIGMA protocol. Therefore, the values $s$ is only contained in the $MAC$ message. And then, in order to prove that the M1-SIGMA satisfies the first requirement of Definition 3, we need to use some additional assumptions (that are the DDH assumption and the security of $MAC$ and $PRF$ functions).

**M1-SIGMA satisfies the second security requirement.** we prove that the property P2 of the basic SIGMA protocol are still valid for M1-SIGMA.

The following proposition is a version (corresponding to M1-SIGMA) of Lemma 7 in [3] and Proposition 5 in the previous section. However, the proof of this proposition is more complex than that of Lemma 7 in [3] and Proposition 5. And, it is similar to Proposition 16, our proof is based on the security assumption of $MAC$, $PRF$, and $SIG$ functions, and the DDH assumption. In particular, we will use the proof of Proposition 16 to prove this proposition.

**Proposition 13.** *For all attackers $\mathcal{M}$ on M1-SIGMA, the following holds except for negligible probability.*

*(a) Consider a regular run by $\mathcal{M}$ in which $\mathcal{M}$ chooses a test session with output $(\widehat{A}, \widehat{B}, s)$ where $\widehat{A}$ is the initiator. Then:*
*(1) $\widehat{A}$ and $\widehat{B}$ are never corrupted before expiration of the test session.*
*(2) Sessions $(\widehat{A}, s)$ and $(\widehat{B}, s)$ are never revealed by $\mathcal{M}$.*

*(3)* $(\widehat{B}, s)$ *is initiated as responder with the start message sent by* $(\widehat{A}, s)$.
*(4)* $(\widehat{A}, s)$ *receives a response message after* $(\widehat{B}, s)$ *was activated as responder, and this message carries the same DH exponent as in the response message output by* $(\widehat{B}, s)$.
*(5) Session* $(\widehat{A}, s)$ *does not abort.*

*(b) Consider a regular run by* $\mathcal{M}$ *in which* $\mathcal{M}$ *chooses a test session with output* $(\widehat{B}, \widehat{A}, s)$ *where* $\widehat{B}$ *is the responder. Then:*
*(1)* $\widehat{A}$ *and* $\widehat{B}$ *are never corrupted before expiration of the test session.*
*(2) Sessions* $(\widehat{A}, s)$ *and* $(\widehat{B}, s)$ *are never revealed by* $\mathcal{M}$.
*(3)* $(\widehat{B}, s)$ *is initiated as responder with the start message sent by* $(\widehat{A}, s)$.
*(4)* $(\widehat{A}, s)$ *receives a response message after* $(\widehat{B}, s)$ *was activated as responder, and this message carries the same DH exponent as in the response message output by* $(\widehat{B}, s)$.
*(5) Session* $(\widehat{A}, s)$ *does not abort.*

*Proof.* **Proof of** $(a)$:
(1) $\mathcal{M}$ is not allowed to corrupt the peers to the test session and we have assumed (wlog) that it does not do that.
(2) $(\widehat{A}, s)$ cannot be revealed by $\mathcal{M}$ since $\mathcal{M}$ is not allowed to expose the test session. As for $(\widehat{B}, s)$, a state-reveal query can be done only against incomplete session (since upon completion sessions erase their state). However, while incomplete, $(\widehat{B}, s)$ is the matching session to the test session so $\mathcal{M}$ cannot issue state-reveal query against it.
(3) Since $(\widehat{A}, \widehat{B}, s)$ completes, it means that $\widehat{A}$ received a response message with identity $\widehat{B}$ in it. In particular, it means that $\widehat{A}$ verified the signature $SIG_{\widehat{B}}(\text{"1"}, g^x, g^y, MAC_{k_m}(s, \widehat{B}))$ under $\widehat{B}$'s public key and where $g^x$ was the value included by $(\widehat{A}, s)$ in its start message, and $k_m = PRF_{g^{xy}}(1)$. It is similar to the proof of Proposition 16, we see that the probability that $\mathcal{M}$ can activate $(\widehat{B}, s)$ as a responder with the initiator message $g^{x'}, s$ (where, $g^{x'} \neq g^x$) is negligible. (It is based on the DDH assumption and the security assumptions of $MAC$, $PRF$ and $SIG$ functions). Therefore, the signature produced by $\widehat{B}$ that $\mathcal{M}$ sends to $(\widehat{A}, s)$ is under session $s$ and $\widehat{B}$ is activated as a responder of session $s$ under the DH exponent $g^x$ as output in the start message by $(\widehat{A}, s)$.
(4) $(\widehat{A}, s)$ completes with an output $(\widehat{A}, \widehat{B}, s)$ so it must have received a response message which included $\widehat{B}$ as the identity. Moreover, $\widehat{A}$ verified the signature in the response message under $\widehat{B}$'s public key, namely $SIG_{\widehat{B}}(\text{"1"}, g^x, g^y, MAC_{k_m}(s, \widehat{B}))$. If $(\widehat{B}, s)$ was not activated as a responder then $\widehat{B}$ would have never generated a signature

$SIG_{\widehat{B}}(\text{"1"}, ..., MAC_{k_m}(s, Q))$, then follow to the proof of Proposition 16, we have that at least one of the assumptions (the DDH assumption and the security assumption of MAC, $PRF$ and SIG functions) is violated. If $\widehat{B}$ generated such a signature then we have that $g^y$ included under that signature was the DH exponent in the response message generated by $(\widehat{B}, s)$, and since $\widehat{A}$ verified it using the DH exponent it received in the response message then we have that either this is the same exponent generated and sent by $\widehat{B}$ or at least one of the assumptions (the DDH assumption and the security assumption of $MAC$, $PRF$ and $SIG$ functions) is violated (by using the argument in the proof of Proposition 16).

(5) Clearly, session $(\widehat{A}, s)$ does not abort since it completes.

**Proof of** $(b)$: It is similar to $(a)$. □

The following propositions are versions (in M1-SIGMA) corresponding to Proposition 6, 7,..., 15. In addition, their proofs are quite similar to the proofs of the corresponding lemmas. Therefore, here we only state these propositions without proving.

**Proposition 14.** *For all attackers $\mathcal{M}$ on M1-SIGMA, the outputs of $\widehat{S}_{RAND}(\mathcal{M})$ and $\widehat{S}_{HYBR}(\mathcal{M})$ are indistinguishable.*

**Proposition 15.** *For any attacker $\mathcal{M}$ on M1-SIGMA, the probability of a guess event under a run of $\widehat{S}_{RAND}(\mathcal{M})$ is at least $1/(mn)$, where $m$ is the number of sessions initiated by $\mathcal{M}$ and $n$ is the number of parties in the M1-SIGMA protocol.*

**Proposition 16.** *For any attacker $\mathcal{M}$ on M1-SIGMA, the probability of a guess event under a run of $\widehat{S}_{HYBR}(\mathcal{M})$ is, up to a negligible difference, the same as the probability of a guess event under a run of $\widehat{S}_{RAND}(\mathcal{M})$.*

**Proposition 17.** *Proposition 20 holds for $\widehat{S}_{RAND}$ as well.*

**Proposition 18.** *For all attackers $\mathcal{M}$ on M1-SIGMA,*

$$P_{RAND}(\mathcal{M}) = Pr(\widehat{S}_{RAND} \texttt{ outputs 1: GUESS events}).$$

**Proposition 19.** *For all attackers $\mathcal{M}$ on M1-SIGMA,*

$$P_{REAL}(\mathcal{M}) = Pr(\widehat{S}_{REAL} \texttt{ outputs 1: GUESS events}).$$

**Proposition 20.** *For all attackers $\mathcal{M}$ on M1-SIGMA, the simulators $\widehat{S}_{REAL}(\mathcal{M})$ and $\widehat{S}_{RAND}(\mathcal{M})$ are indistinguishable.*

**Proposition 21.** *The M1-SIGMA protocol satisfies the second requirement of Definition 3: for all attacker $\mathcal{M}$ on M1-SIGMA, $|P_{REAL}(\mathcal{M}) - P_{RAND}(\mathcal{M})|$ is negligible.*

As a consequence, the M1-SIGMA protocol is secure in the post-specified peer model.

**Proposition 22.** *Assuming DDH and the security of the underlying cryptographic functions $SIG$, $MAC$, $PRF$, the M1-SIGMA protocol is secure in the Canetti-Krawczyk post-specified model.*

## 6 Conclusion

In this paper, we considered and analyzed two variant of the basic SIGMA protocol, namely M-SIGMA and M1-SIGMA, in which the $MAC$ tag is not sent separately but rather it is computed under the signature operation. We obtained that these two variants are secure in the post-specified model. Our analysis is essentially based on the analysis of the basic SIG protocol in [3]. However, there are some differences between our analysis and the analysis in [3], that is the proofs of Proposition 9, 16, 17 and 21.

Besides, there are some significant differences between the proofs of M-SIGMA and M1-SIGMA. Hence we can see that, a slight modification can make our analysis is more complex than its origin. This is one of our reasons that we consider the security of M-SIGMA and M1-SIGMA instead of the protocol in Figure 2, and then we hope that we can provide a security proof for that protocol.

## References


[1] Harkins, D. and Carreal, D. The Internet key-exchange (IKE). IETF (The Internet Engineering Task Force), New York, NY, USA, Tech. Rep. 2409. Nov. 1998.

[2] Canetti, R. and Krawczyk, H. Analysis of Key Exchange Protocols and Their Use for Building Secure Channels. Advances in Cryptology – EUROCRYPT 2001. `http://eprint.iacr.org/2001/040`.

[3] Canetti, R. and Krawczyk, H. Security Analysis of IKE's Signature-based Key-Exchange Protocol. Crypto 2002. LNCS Vol. 2442. Full version in: Cryptology ePrint Archive at `http://eprint.iacr.org/2002/120`.

[4] Krawczyk, H. SIGMA: The 'SIGn-and-MAc'approach to authenticated Diffie-Hellman and its use in the IKE protocols. Annual International Cryptology Conference, Springer, Berlin, Heidelberg. 2003.

[5] Menezes, A. and Ustaoglu, B. Comparing the Pre- and Post-specified Peer Models for Key Agreement. Australasian Conference on Information Security and Privacy – ACISP 2008: Information Security and Privacy, pages 53–68. 2008.

[6] Yang, Z. Efficient eck-secure authenticated key exchange protocols in the standard model. In: International Conference on Information and Communications Security, Springer, Cham, pages 185–193. 2013.

[7] Yao, A.C.C. and Zhao, Y. Privacy-preserving authenticated key-exchange over Internet. In: International Conference on Information and Communications Security. 2014.

[8] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P. and Kivinen, T. Internet key exchange protocol version 2 (IKEv2), (No. RFC 7296). 2014.

[9] Krawczyk, H. and Wee, H. The OPTLS protocol and TLS 1.3. In: Security and Privacy, 2016 IEEE European Symposium on. IEEE, pages 81–96. 2016.

# A New LWE-based Verifiable Threshold Secret Sharing Scheme

Saba Karimani, Zahra Naghdabadi,
Taraneh Eghlidos, and Mohammad Reza Aref

Sharif University of Technology, Tehran, Iran
{saba_karimani, naghdabadi_z}@ee.sharif.edu

**Abstract**

In this paper we propose a verifiable threshold secret sharing scheme based on learning with errors (LWE) problem. In secret sharing schemes shares are sent through a secure channel. For the sake of consistency between share generation process and share distribution platform, we propose a lattice based secret sharing scheme. To the best of our knowledge, this scheme is the first LWE-based verifiable threshold secret sharing scheme. In this paper, we use Micciancio and Peikert's algorithm to produce a trapdoor. Shares are distributed using an LWE-based public-key cryptosystem. It is shown that the computational security is based on the hardness of LWE problem and one-wayness of Ajtai's function.

**Keywords:** lattice based cryptography, learning with errors (LWE) problem, threshold secret sharing, trapdoor function.

## 1 Introduction

By the appearance of quantum computers, some quantum algorithms have been developed by Shor [1] which threatens the security of the number theoretic cryptosystems. Since then, researchers considered post-quantum cryptographic algorithms because of their resistance against quantum attacks. Lattice based cryptographic schemes are among post-quantum candidates. For compatibility of the share generation process with the cryptographic platform, we introduce a lattice based secret sharing scheme. Also to provide verifiability of the shares and the secret, we exploit the worst-case hardness of shortest vector problem (SVP) as one of the hard lattice problems.

Secret sharing schemes are used in several applications in cryptography such as secure attribute based encryption [2], key management [3] and threshold cryptography [4]. Moreover, secret sharing schemes make it possible to share a secret among a set P of participants in a way that only certain subsets

of them can recover the secret. The set of all authorized subsets of participants is called the access structure denoted by $\Gamma$. Each participant is given a value as his/her share. The secret sharing scheme should be designed in a way that any authorized subset in $\Gamma$ be able to recover the secret and any unauthorized subset cannot get any information about the secret.

In a $(t,\ n)$ threshold secret sharing scheme, the access structure $\Gamma$ is the collection of all t-subsets of P. In other words, any t participants form an authorized set and are able to recover the secret. Most secret sharing schemes consist of two stages. First, using the algorithm, the so-called dealer computes the shares and sends them securely to the participants. At the second stage, at least t participants put their shares into the combiner in order to recover the secret. This notion was first introduced by Shamir and Blakley independently in 1979 [5], [6] which are based on Lagrange interpolator polynomial and linear projective geometry, respectively.

Later, some other secret sharing schemes with extra features such as threshold changeability [7],[8], shares verification [9] and sharing more than a secret at the same time [10],[11] have been proposed.

Notably, Shamir's secret sharing is information theoretically secure but in practice for secure share distribution among participants, a public key cryptography is normally used. But security of classical public key cryptosystems are usually based on hardness of factorization and discrete logarithm problems which are solved by Shor's quantum algorithm. Therefore, Shamir's secret sharing cannot be practically handled securely. In addition to security issues, consistency of the secret sharing algorithm with the lattice based public key infrastructure is also a concern, Therefore, Shamir's secret sharing could be replaced by a lattice based secret sharing scheme.

So far, there has been neither quantum nor classic algorithm for solving hard lattice problems [12]. Consequently, it is supposed that they are secure against cryptographic attacks. Moreover, it is applicable due to its efficient linear computations. Also the security of such schemes can be proven based on worst-case hardness of lattice problems such as shortest vector problem (SVP) and closest vector problem (CVP). In 1996 Ajtai introduced the first lattice based constructions [13]. He proposed a family of one-way collision resistant functions [13], [14].

In [7], Stienfeld has proposed a lattice based construction to change the threshold of any Shamir based secret sharing scheme without need to distribute new shares to the participants. A lattice based $(n,\ n)$ secret sharing scheme was introduced in 2011 by Georgescu [15] whose security relies on the hardness of LWE problem. In 2012 Bansarkhani [16] proposed another

lattice based ($n$, $n$) threshold verifiable secret sharing scheme whose security is based on the approximate SVP. In these schemes, all participants are required to involve in the process of secret reconstruction. Amini et al. [17] in 2014, proposed a lattice based ($t$, $n$) threshold secret sharing scheme with asymptotic security which works only in some special cases and does not enjoy verifiablity. The security of the other schemes is based on the one-wayness of Ajtai's function.

In [11] Pilaram et al. have proposed a lattice based multi-stage ($t$, $n$) threshold secret sharing scheme in which all computations are linear. The scheme shares a number of secrets among participants. The secrets can be recovered independently in any order. Also each participant uses one pseudo-secret share, therefore, by recovering one secret the shares are still unknown for recovering the remaining secrets. The security is based on one-wayness of Ajtai's function [13].

In this paper, to the best of our knowledge, we propose the first lattice based verifiable ($t$, $n$) threshold secret sharing scheme whose security is based on LWE problem. In this scheme each share has four components. Firstly, an algorithm is used to produce a matrix A and its corresponding trapdoor R. Then, the first component of shares is produced using matrix A. The second component is a noise value added to the inner product of the secret with the first component. The third and fourth components are computed from the trapdoor. For recovering the secret the trapdoor is shared among the participants in such a way that any $t$ of them can recover the secret.

The paper is organized as follows: Section 2 provides a brief review of lattices and some hard lattice problems. Section 3 is dedicated to the proposed LWE-based scheme. The security and correctness of the proposed scheme are discussed in Section 4. Section 5 concludes the paper.

## 2 Preliminaries

In this section, some basic concepts of lattices, lattice based cryptography, and secret sharing schemes are introduced.

### 2.1 Notations

In this paper, lower case letters represent column vectors and upper case letters denote matrices. The transpose of a matrix is denoted by $(.)^T$. The finite field of integers modulo $q$ is represented by $\mathbb{Z}_q$. The set of all $m \times n$ matrices with entries in a ring $R$ is denoted by $R^{m \times n}$. The sign $\|.\|$ denotes any arbitrary norm. The notation $< a.b >$ stands for the inner product of

vectors $a$ and $b$. The notation $O(.)$ is used to show the growth of functions.

### 2.2 Lattices

In this paper, by lattice we mean a regular array of points in an m-dimensional real vector space.

**Definition 1.** (18) let $b_1, b_2, ..., b_n$ be n linearly independent vectors in vector space $\mathbb{R}^m$. The lattice $\Lambda$ is defined as the set of all integer linear combinations of $b_1, b_2, ..., b_n$ as follows:

$$\Lambda = L(b_1, b_2, ..., b_n) = \{\sum_{i=1}^{n} x_i b_i; x_i \in \mathbb{Z}\}$$

The set of vectors $\{b_1, b_2, ..., b_n\}$ is called a basis for the lattice and n is called the rank of the lattice.

The security of any lattice based cryptosystem is usually based on NP-Hard problems in lattices such as SVP, CVP and LWE. Shortest vector problem (SVP) is the problem of finding the shortest nonzero vector in a given lattice and Closest vector problem (CVP) is the problem of finding the closest vector of the lattice to the given target point. The learning with errors (LWE) problem [19] is a set of linear equations with random noise values added to each equation is considered.

### 2.3 Lattice based Secret Sharing Scheme

In [11] Pilaram and Eghlidos proposed a lattice based threshold multi-stage secret sharing scheme, which we call PE from now on, for sharing the secrets $s_1, ..., s_m \in \mathbb{Z}_q^t$. First, the dealer announces a public random vector $v \in \mathbb{Z}_q^t$ whose last entry is *1* for which there exist different lattice bases $B_i \in \mathbb{Z}_q^{t \times t}$ such that $s_i = B_i v$ for $i = 1, ..., m$. Then the dealer chooses $n$ random public vectors$\lambda_j \in \mathbb{Z}_q^t$ for$j = 1, ..., n$ such that every $t$ of these vectors are linearly independent. The share of participant $P_i$ is a random vector $c_i \in \{0, 1\}^r$ which is sent by the dealer to$P_i$ through a secure channel. Then the dealer computes the public matrices $A_i$ such that $A_i c_j = B_i \lambda_j$ for $i = 1, ..., m$ and $j = 1, ..., n$ and makes them public. Second, a set of authorized participants $\{j_1, ..., j_t\} \subseteq \{1, ..., n\}$ can recover the matrix $B_i$ corresponding to the secret $s_i$ by computing pseudo-shares $d_{j_k}^i = A_i c_{j_k}$ by $P_{j_k}$ for $k = 1, ..., t$ and then computing $B_i = [d_{j_1}^i, ..., d_{j_t}^i][\lambda_{j_1}, ..., \lambda_{j_t}]^{-1}$. Then, using the public vector $v$, they recover the secret $s_i = B_i v$.

### 2.3.1 Ajtai's hash function

Ajtai in [13] introduced a family of one-way functions called Ajtai's function, which are hard to convert. Ajtai showed that converting the function $f_A(x) = Ax \ (mod\ q)$ for $n, m, q, d \in \mathbb{N}, m > \frac{n \log q}{\log d}, q = O(n^c)$, random $x \in \{0, 1, ..., d-1\}^m$ and uniformly random $A \in \mathbb{Z}_q^{n \times m}$ is equivalent to solving any instance of approximate SVP which is still hard as there is no any classic/quantum algorithm to solve it. Note that the Ajtai's hash function is collision resistant based on the q-ary lattice $\lambda_q^{\perp}(A) = \{x \in \mathbb{Z}^m : Ax = 0(mod q)\}$. In this lattice the matrix $A$ is called as key by Ajtai.

## 3 The Proposed Scheme

In lattice problems, a trapdoor is an intermediator to facilitate the access to a good basis. Micciancio and Peikert have introduced two algorithms in [20] for generating trapdoor and inverting LWE problem with the corresponding trapdoor which we use in our scheme. These algorithms are as follows:

**Algorithm (1)**: An efficient randomized algorithm which on inputs $t \geqslant 1, q \geqslant 2$ and $m = t([\log q] + 2)$, outputs a matrix $A \in \mathbb{Z}_q^{t \times m}$ and a trapdoor $R \in \mathbb{Z}^{2t \times t\lceil \log q \rceil}$ such that $A$ is computationally pseudorandom matrix (PR) under LWE assumption.

**Algorithm (2)**: An efficient algorithm, with overwhelming probability over all random choices, for $s \in \mathbb{Z}_q^m$ and $\|a\| < \frac{q}{o(\sqrt{t \log q})}$ or $e \leftarrow D_{\mathbb{Z}^t, \alpha q}$ for $\frac{1}{\alpha} \geqslant \sqrt{t \log q}.\omega_t$, on inputs a PR matrix $A$, a trapdoor $R$ and a vector $b$ in the form of $b = As + e$, outputs $s$.

In this section we propose a new verifiable LWE-based threshold secret sharing scheme(See pseudocode in Appendix). Using Algorithm (1), the trapdoor $R$ is chosen randomly to produce a matrix $A$. If $A$ is singular, we run Algorithm(1) again. Note that $A$ must be nonsingular to be used in our scheme. The secret s is shared using the corresponding shares that are distributed beforehand. Algorithm (2) is used to reconstruct the secret with the trapdoor $R$ as its input.

Let $n$ and $t$ be the number of participants and the threshold, respectively, such that $n < 2t$. For sharing the secret $s \in \mathbb{Z}_q^{m \times 1}$, the dealer runs Algorithm (1) to obtain the matrix $A$ and the trapdoor matrix $R$. Let the vector $a_i^T$ denote the $i^{th}$ row of $A$, for $i = 1, ..., t$. The dealer chooses the random coefficients $\alpha_j^i \in \mathbb{Z}_q$ for $i = t+1, ..., n$ and $j = 1, ..., t$. The share given to the $i^{th}$ participant is a four tuple $(\tilde{a}_i, \tilde{b}_i, r_i, \tilde{r}_i)$ for $i = 1, ..., n$ . The first

component $\tilde{a_i}$, for $i = t+1, ..., n$ is constructed as follows.

$$\tilde{a_i} = \sum_{j=1}^{t} \alpha_j^i a_j, i = t+1, ..., n \quad (1)$$

Also for $i = 1, ..., t$ we have $\tilde{a_i} = a_i$. For the second part of each share we have $\tilde{b_i} =< \tilde{a_i}.s > +e_i$ in which $s$ is the secret and the noise value $e_i$ comes from a random distribution that satisfies the conditions of Algorithm (1). Also, $A$ is pseudorandom since it is obtained from Algorithm (1). The vector $\tilde{b}$ is produced based on LWE problem.

For constructing the third and fourth share components, the dealer uses a modification of PE secret sharing scheme for one secret. Assume the first $t \times t$ submatrix of $R$ as $\tilde{R_1}$ and the second $t \times t$ submatrix of $R$ started from the $(t+1)^{th}$ row as $\tilde{R_2}$. The dealer publishes the remaining entries of $R$ as $\bar{R}$. Since by Algorithm(1), $R$ is a random matrix, the public values corresponding to $R$ do not leak any information about the secret. The dealer then chooses $n$ public vectors $\lambda_i \in \mathbb{Z}_q^t$ for $i = 1, ..., n$, each $t$ of which are linearly independent. The vectors $\tilde{\lambda_i} \in \mathbb{Z}_q^t$ for $i = 1, ..., n$ are chosen in the same way. The third and the fourth parts $r_i$ and $\tilde{r_i}$ are constructed by computing $r_i = \tilde{R_1}\lambda_i$ and $\tilde{r_i} = \tilde{R_2}\lambda_i$. The dealer also publishes two random matrices $F \in \mathbb{Z}_q^{p \times m}$ and $C \in \mathbb{Z}_q^{p \times t}$ where $p \log q < t < m$ and $q = O(p^c)$ for some constant c. Besides, he publishes the hash values $f_s = Fs, \tilde{f_{i_1}} = F\tilde{a_i}, \tilde{f_{i_2}} = Fb_i$ where $b_i$ is the binary form of $\tilde{b_i}$ in $m$ bits, $\tilde{f_{i_3}} = Cr_i$ and $\tilde{f_{i_4}} = C\tilde{r_i}$ for $i = 1, ..., n$. Since the vectors $s, \tilde{a_i}, r_i, \tilde{r_i}$ and $b_i$ are chosen randomly, the hash values do not leak any information about the hidden values.

In the next step, the dealer sends the corresponding shares to the participants through a secure channel. By receiving the share, the participant $P_i$ uses the public values to verify his/her share. If the share is verified, he/she continues the process, if not, the participant asks the dealer to send his/her share again.

For recovering the secret, the participants $P_{i_1}, P_{i_2}, ..., P_{i_t}$ for $\{i_1, ..., i_t\} \subset \{1, ..., n\}$, make their shares available to the combiner. Using the public values $\tilde{f_{i_1}}, \tilde{f_{i_2}}, \tilde{f_{i_3}}$ and $\tilde{f_{i_4}}$ the combiner verifies the correctness of shares. If the shares of the participants $P_{i_1}, P_{i_2}, ..., P_{i_t}$ were correct, the combiner forms the matrix

$\tilde{A}$ and the vector $\tilde{b}$ as follows:

$$\tilde{A} = \begin{bmatrix} \tilde{a_{i_1}}^T \\ \vdots \\ \tilde{a_{i_t}}^T \end{bmatrix}, \tilde{b} = \begin{bmatrix} \tilde{b_{i_1}} \\ \vdots \\ \tilde{b_{i_t}} \end{bmatrix} \tag{2}$$

Also by using the vectors $r_{i_1}, ..., r_{i_t}$ and the public vectors $\lambda_{i_1}, ..., \lambda_{i_t}$ the matrix $\tilde{R_1}$ is recovered by computing $\tilde{R_1} = [r_{i_1}, ..., r_{i_t}][\lambda_{i_1}, ..., \lambda_{i_t}]^{-1}$. With the same computations we obtain $\tilde{R_2} = [\widetilde{r_{i1}}, ..., \widetilde{r_{it}}][\widetilde{\lambda_{i1}}, ..., \widetilde{\lambda_{it}}]^{-1}$. Then the matrix $\tilde{R}$ is reconstructed as $\tilde{R} = \binom{\tilde{R_1}}{\tilde{R_2}}$. Using $\tilde{R}$ and the public matrix $\bar{R}$, we obtain $R = [\tilde{R}|\bar{R}]$. Furthermore, the combiner runs Algorithm (2) on the equation $\tilde{b} = \tilde{A}s + \tilde{e}$ to find $s$ as the output. Finally, by comparing $Fs$ with the public value $f_s$, everyone can check whether $s$ is correct or not. The complexity of the scheme is of $O(t^3)$.

## 4 Analysis of Security and Correctness

### 4.1 Correctness

The proof of correctness is threefold. First we show that the recovered trapdoor $R$ is a trapdoor for $\tilde{A}$. The correctness of our scheme inherits from PE secret sharing scheme, because we have used PE scheme to generate the corresponding share of $R$.

Secondly, we have to show the recovered $\tilde{A}$ is a key of the q-ary lattice $\lambda_q^{\perp}(A)$. Since the rows of $\tilde{A}$, i.e. $\tilde{a_i}^T$'s, are either rows of A or random linear combinations of them, any $t$ of them are linearly independent, due to the non-singularity of $A$. Both $A$ and $\tilde{A}$ construct the same q-ary lattice that is $\lambda_q^{\perp}(A) = \lambda_q^{\perp}(\tilde{A})$. Hence, they form the same key matrix for it. Third, using Algorithm (2) for $\tilde{b}, \tilde{A}$ and $R$ we can extract $s$ from $\tilde{b} = \tilde{A}s + \tilde{e}$. This is guaranteed by correctness of Algorithm (2) which is proved in [20].

### 4.2 Security

The security proof of the proposed scheme consists of two parts based on the two following theorems.

**Theorem 1.** *In the proposed scheme, any subset of participants of size less than t cannot recover the undisclosed trapdoor R.*

*Proof.* The trapdoor $R$ is shared between $n$ participants by the modified PE algorithm for one secret which is a ($t$, $n$) threshold secret sharing scheme,

where the union of at least $t$ participants is necessary for recovering $R$. Therefore, the proposed scheme owe its security to PE secret sharing scheme. Since by Algorithm(1), R is a random matrix, so the public values corresponding to $R$ do not leak any information about secret and shares. □

By theorem 1, at least $t$ participants are required for recovering $R$ and using it to obtain $s$ from Algorithm (2). Now we prove that with less than $t$ participants, it is computationally impossible to recover $s$.

**Theorem 2.** *In the proposed scheme, any subset of participants of size less than t cannot recover the undisclosed secret s.*

*Proof.* First, since $\tilde{A}$ obtained from Algorithm (1) is pseudorandom, then $\tilde{b}$ has the LWE construction. Here, we reduce the LWE problem into the problem used in our scheme. When less than $t$ shares are given to the combiner, the matrix $\tilde{A}$ and the vector $\tilde{b}$ are not completely constructed. If the secret $s$ could be recovered by pulling less than $t$ shares together, then equivalently the LWE problem would be solved. □

Since the secret $s$ is random, the public value $\tilde{f}_s$ does not leak any information about the secret based on one-wayness of Ajtai's hash function.

## 5 Conclusions

In this paper, we have proposed a new verifiable threshold LWE-based secret sharing scheme. The previously proposed schemes do not satisfy whether the threshold condition or verifiability. In the new scheme, each share is a four tuple whose first two entries hide the secret, based on the hardness of LWE problem, and the other entries share the trapdoor among the participants. Therefore, this scheme enjoys a double layer security, meaning even if the trapdoor is disclosed, the secret is still secure. The secret can be recovered only when an authorized subset of participants put their shares together. Verifiability is satisfied comparing share components with the corresponding public values. We have proved the computational security of the LWE-based scheme is subject to the worst-case hardness of lattice problems which have so far been secure against quantum algorithms.

For the sake of security, the suggested parameters of LWE public key algorithm are larger than those used in classical cryptographic algorithms. Therefore, in the proposed scheme, the share size should be compatible with the size of the parameters used in the LWE public key algorithm. There is always a tradeoff between efficiency and security. That is the main reason

why LWE-based post-quantum algorithms are not applicable compared to number theoretical algorithms, even though they enjoy provable security.

## References


[1] Shor P. W. Algorithms for quantum computation: Discrete logarithms and factoring, in Proc. 35th Annu. Symp. Found. Comput.Sc., 1994, pp. 124–134.

[2] Waters B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In International Workshop on Public Key Cryptography, pp. 53-70. Springer, Berlin, Heidelberg, 2011.

[3] Harn L. and Changlu L. Authenticated group key transfer protocol based on secret sharing." IEEE transactions on computers 59, no. 6 (2010): 842-846.

[4] Desmedt Y. and Frankel Y. Shared generation of authenticators and signatures. In Annual International Cryptology Conference, pp. 457-469. Springer, Berlin, Heidelberg, 1991.

[5] Shamir A. How to share a secret, Commun. ACM, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[6] Blakley G. R. Safeguarding Cryptographic Keys, in Proc. AFIPS Nat. Comput. Conf., Jun. 1979, vol. 48, pp. 313–317.

[7] Steinfeld R., Huaxiong W., and Pieprzyk P. Lattice-based threshold-changeability for standard Shamir secret-sharing schemes. In International Conference on the Theory and Application of Cryptology and Information Security, pp. 170-186. Springer, Berlin, Heidelberg, 2004.

[8] Pilaram H. and Eghlidos T. A lattice-based changeable threshold multi-secret sharing scheme and its application to threshold cryptography. Scientia Iranica. Transaction D, Computer Science and Engineering, Electrical 24, no. 3 (2017): 1448-1457.

[9] Stadler M. Publicly verifiable secret sharing. In International Conference on the Theory and Applications of Cryptographic Techniques, pp. 190-199. Springer, Berlin, Heidelberg, 1996.

[10] Blundo C., De Santis A., Di Crescenzo G., Gaggia A. G. and Vaccaro U. Multi-secret sharing schemes, in Proc. 14th Annu. Int. Cryptol. Conf. Adv. Cryptol., 1994, pp. 150-163.

[11] Pilaram H., and Eghlidos T. An efficient lattice based multi-stage secret sharing scheme. IEEE Transactions on Dependable and Secure Computing 14, no. 1 (2017): 2-8.

[12] Bernstein D., Buchmann J. and Dahmen E. Post-Quantum Cryptography. New York, NY, USA: Springer [Online]. Available: `http://books.google.com/books?id=VB5981O47NAC`. 2009.

[13] Ajtai M. Generating hard instances of lattice problems (extended abstract), in Proc. 28th Annu. ACM Symp. Theory Comput., 1996, pp. 99-108.

[14] Goldreich O., Goldwasser S. and Halevi S. Collision-Free Hashing from Lattice Problems. IACR Cryptology ePrint Archive 1996: 9. 1996.

[15] Georgescu A., A LWE-based secret sharing scheme, IJCA Special Issue Netw. Security Cryptography, vol. NSC, no. 3, pp. 27-29, Dec. 2011.

[16] El Bansarkhani R. and Meziani M. An efficient lattice-based secret sharing construction, in Proc. 6th Inf. Security Theory Practice: Security, Privacy Trust Comput. Syst. Ambient Intell. Ecosyst., 2012, pp. 160–168.

[17] Amini Khorasgani H., Asaad S., Eghlidos T. and Aref M.R., A lattice-based threshold secret sharing scheme. In Information Security and Cryptology (ISCISC), 2014 11th International ISC Conference on, pp. 173-179. IEEE, 2014.

[18] Micciancio D. and Goldwasser S. Complexity of Lattice Problems: A Cryptographic Perspective. ser. Milken Institute Series on Financial Innovation and Economic Growth. New York, NY, USA: Springer. 2002.

[19] Regev O. On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM) 56, no. 6:34. 2009.

[20] Micciancio D. and Peikert C. Trapdoors for lattices: Simpler, tighter, faster, smaller. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 700-718. Springer, Berlin, Heidelberg, 2012.

## Appendix

## A Pseudocode

The dealer computes the shares as follows:
*Set parameters:*
*n: the number of participants;*
*t: the threshold such that* $n < 2t$*;*
*1. Run Algorithm (1) on inputs* $t \geqslant 1, q \geqslant 2, m = t\lceil \log q \rceil$ *output A,R.*
$A = \begin{bmatrix} a_1^T \\ \vdots \\ a_t^T \end{bmatrix}, R = \begin{bmatrix} \widetilde{R_1} & \bar{R} \\ \widetilde{R_2} & \end{bmatrix}$ *and make* $\bar{R}$ *public;*
*2. Choose integers* $\alpha_j^i \in \mathbb{Z}_q$ *for* $t+1 \leqslant i \leqslant n, 1 \leqslant j \leqslant t$*, uniformly at random;*
*3. Set* $\tilde{a}_i = a_i$ *for* $1 \leqslant i \leqslant t$*;*
*4. Set* $\tilde{a}_i = \sum_{j=1}^{t} \alpha_j^i \alpha_j$ *for* $t+1 \leqslant i \leqslant n$*;*
*5. Choose* $e_i$ *from a random distribution that satisfies the conditions of Algorithm (1);*
*6. Set* $\tilde{b}_i =< \tilde{a}_i.s > +e_i$ *for* $1 \leqslant i \leqslant n$*;*
*7. Choose* $\lambda_i, \widetilde{\lambda_i} \in \mathbb{Z}_q^t$ *randomly with uniform distribution and make them public;*
*8. Set* $r_i = \widetilde{R_1}\lambda_i, \widetilde{r_i} = \widetilde{R_2}\widetilde{\lambda_i}$ *for* $1 \leqslant i \leqslant n$*;*
*9. Choose* $F \in \mathbb{Z}_q^{p \times m}, C \in \mathbb{Z}_q^{p \times t}$ *randomly and make them public;*
*10. Set* $f_s = Fs, \widetilde{f_{i_1}} = F\tilde{a}_i, \widetilde{f_{i_2}} = Fb_i$ *where* $b_i$ *is the binary form of* $\tilde{b}_i, \widetilde{f_{i_3}} = Cr_i, \widetilde{f_{i_4}} = C\tilde{r}_i$ *for* $1 \leqslant i \leqslant n$ *and make them public;*
*11. Send the share* $(\widetilde{a_i}, \widetilde{b_i}, r_i, \widetilde{r_i})$ *to the participant* $P_i$ *for* $1 \leqslant i \leqslant n$*.*

Share Verification:
*1. Each participant verifies his share by comparing* $F\tilde{a}_1$ *with* $\widetilde{f_{i_1}}$*,* $Fb_i$ *with* $\widetilde{f_{i_2}}$*,* $Cr_i$ *with* $\widetilde{f_{i_3}}$ *and* $C\tilde{r}_i$ *with* $\widetilde{f_{i_4}}$*.*
*2. If shares are correctly verified, continue.*
*3. Else, ask the dealer to resend the shares.*

Secret Recovery:
When the participants $\{P_{i_1}, P_{i_2}, ..., P_{i_t}\}$ get together and put their shares into the combiner, it runs the algorithm as follows:
*1. Set* $\tilde{R_1} = [r_{i_1}, ..., r_{i_t}][\lambda_{i_1}, ..., \lambda_{i_t}]^{-1}$*;*
*2. Set* $\tilde{R_2} = [\widetilde{r}_{i1}, ..., \widetilde{r}_{it}][\widetilde{\lambda}_{i1}, ..., \widetilde{\lambda}_{it}]^{-1}$*;*

*3. Set* $R = \begin{bmatrix} \widetilde{R_1} & \bar{R} \\ \widetilde{R_2} & \end{bmatrix}$;

*4. Set* $\tilde{A} = \begin{bmatrix} \widetilde{a_{i_1}}^T \\ \vdots \\ \widetilde{a_{i_t}}^T \end{bmatrix}, \tilde{b} = \begin{bmatrix} \widetilde{b_{i_1}} \\ \vdots \\ \widetilde{b_{i_t}} \end{bmatrix}$

*5. Run Algorithm (2) on input* $(\tilde{A}, R, \tilde{b})$ *to obtain the secret* $s$.

# Probabilistic aspects and Applications

# Data Recovering for a Neural Network-based Biometric Authentication Scheme

Vladimir Mironkin[1] and Dmitry Bogdanov[2]

[1]National Research University Higher School of Economics, Moscow, Russia
mironkin.v@mail.ru
[2]TVP Laboratory, Moscow, Russia
bogdanovds@rambler.ru

**Abstract**

The scheme for protection of neural network biometric containers using cryptographic algorithms is studied. The method of recovering information defining the neural network is proposed. The inconsistency of combining password and neural network biometric protection systems is shown.

**Keywords:** neural network, password, biometric authentication system, protection of neural network container.

## 1 Introduction

Biometry is a very attractive tool for user authentication in information systems. The classic way to implement biometric authentication schemes is comparing the vector of biometric parameters with the existing template and granting access rights to a user according to a given criterion. However, this method has a significant disadvantage because it requires the need to keep biometric template secretly, which makes it difficult to implement such schemes in portable devices [5].

In [8] a concept of fuzzy extractors was introduced. Fuzzy extractors do not require the storage of confidential data on devices. At the registration phase fuzzy extractors generate a secret vector and a public vector (a so called helper) from a person's biometric parameters. When a user authenticates a helper together with uploaded biometric parameters could be transformed into a secret vector, which could be used for user authentication.

In [8] it was proved in some formal model that the secret information does not leak through the helper. Nevertheless, it was disproved in [9] by showing that in some scenarios such information could leak. That is way, obtaining secure fuzzy extractors is an open problem. In [10] a good overview of the

discussed area could be found. In [6] a neural network-based approach to construction of similar schemes is described.

However, further researches have shown that a neural network transformation compromise can cause the effectively recovering of a secret key information [5]. As a result, in order to eliminate this weakness the scheme of a neural network container protection [1] using cryptographic algorithms was proposed in 2017 [7]. However, the results of this work show that the proposed scheme [7] does not provide the stated protection.

## 2 A neural network-based biometric scheme

Consider a neural network converting biometric data (fingerprint, retina, handwritten signature, etc.) into some binary sequence used for data access, or information encryption.

**Definition 1.** *A neuron is a weighted summation of input parameters* $x_0, \ldots, x_{n-1}$*, where* $n \in \mathbb{N}$*. An output value of the neuron is calculated by the formula* $y_i = Z\left(\sum_{i=1}^{n} w_i x_i\right)$*, where* $w_i$ *– a weight coefficient of the neuron,* $Z$ *– Heaviside step function.*

Let's introduce the following notations:

| | |
|---|---|
| $N$ | a number of input biometric parameters, $N \in \mathbb{N}$; |
| $\bar{x}$ | a vector of input biometric parameters, $\bar{x} = (x_0, \ldots, x_{N-1})$; |
| $m$ | a number of neurons in the neural network, $m \in \mathbb{N}$; |
| $n$ | a number of inputs of each neuron, $n \in \mathbb{N}$; |
| $x_{i\sim j}$ | a vector consisting of the $i$-th, ..., the $j$-th bits of a binary representation of $x$, $j > i$; |
| $V_l$ | a set of all binary vectors of a length $l \in \mathbb{N} \bigcup \{0\}$; |
| $V^*$ | a set of all binary vectors of a finite length, $V^* = \bigcup\limits_{l \geqslant 0} V_l$; |
| $h : V^* \to V_l$ | a hash function converting vectors of arbitrary finite length to vectors of a length $l$; |
| $\bar{c}$ | a key sequence corresponding to a legitimate user, $\bar{c} = (c_0, \ldots, c_{m-1}), c_i \in \{0, 1\}$; |
| $\bar{u}$ | a corresponding table consisting of the neuron inputs defined by external inputs of the network; |
| $u_i$ | the $i$-th row of the table $\bar{u}$, $i \in \overline{0, m-1}$; |

| | |
|---|---|
| $\bar{w}$ | a weight table in which the $i$-th row indicates what weights of the biometric parameters $x_{u_i}$ the $i$-th neuron has; |
| $w_i$ | the $i$-th row of the table $\bar{w}$, $i \in \overline{0, m-1}$; |
| $s$ | a fixed parameter "salt", $s \in V_t$, $t \in \mathbb{N}$; |
| $p$ | a password, $p \in V_r$, $r \in \mathbb{N}$; |
| $]v[$ | $]v[ = \min\{n \in \mathbb{Z} : n \geqslant v\}$ for $v \in \mathbb{R}$; |
| $a\|\|b$ | a concatenation of vectors $a, b \in V^*$. |

Let's assume the neural network consists of $m$ neurons, each of them has $n$ inputs. Suppose also the biometric data is encoded by parameters $x_0, \ldots, x_{N-1}$.

**Note 1.** *The process of transformation of the corresponding and the weight tables (see Fig. 1) at the stage of a neural network training [2] is organized in such way that for a legitimate biometric sample at the output of the $i$-th neuron the $i$-th bit of the key sequence is calculated by the formula $c_i = y_i = Z\left(\sum\limits_{i=1}^{n} w_i x_{u_i}\right)$ and for any other biometric samples the corresponding bit is equiprobable.*

The general scheme of a key sequence generation based on the input parameters $\bar{x}$ is shown in Fig. 1.

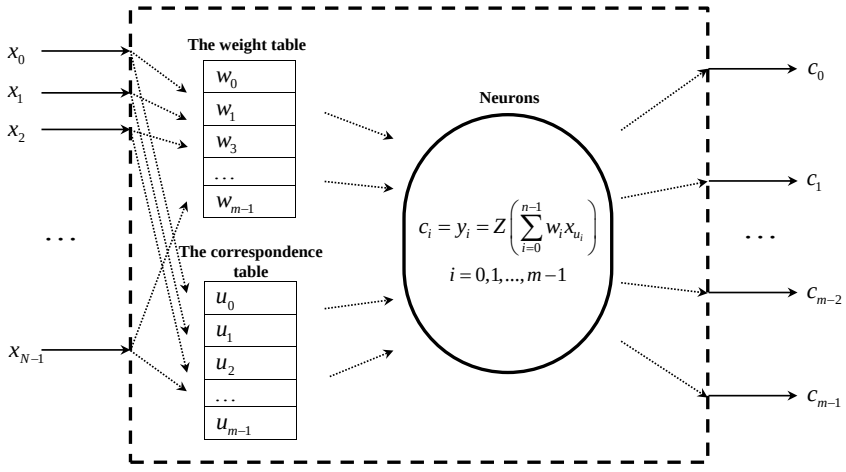



Figure 1: The generation of the key sequence $\bar{c}$

Let's describe a purpose of some functional elements of the neural network [7]:

1. The binary vector of a length $]\log_2(N)[$ is used to encode rows of $\bar{u}$. If $\log_2(N) \notin \mathbb{N}$, a calculation of a number of used biometric parameter is

determined by the relation $\alpha' = \alpha \,(\bmod N)$, where $\alpha$ – an element of a row of $\bar{u}$ and $\alpha'$ – a number of a biometric parameter.

2. The set of integer values of the range $[-2^{d-1}, 2^{d-1}-1]$ is used to encode $w_i, i \in \overline{1,n}$, where $d \geqslant 8$.
3. The binary sequence $\{\gamma_i\}_{i=0}^{m}$ is used to encrypt the corresponding and the weight tables. Components of this sequence are formed for each neuron separately by the hash function $h$, defined in [3]:

$$\gamma_i = \left(h_i^{(1)} || h\left(h_i^{(1)}\right) || \ldots || h^{k-1}\left(h_i^{(1)}\right)\right)_{0 \sim n(b+d)-1}, \tag{1}$$

where $k = \left]\frac{n(b+d)}{l}\right[$ and the components $\gamma_i$ are defined as follows:

$$h_0^{(1)} = h\left(s \| p \| 0\right), \tag{2}$$

$$h_i^{(1)} = h\left(s \| p \| i \| c_0, \ldots, c_{i-1}\right), i > 0. \tag{3}$$

The network encryption is based on a summation of $\gamma_i$ and a concatenation of the corresponding rows of the tables $\bar{u}$ and $\bar{w}$:

$$E_i = (u_i \| w_i) \oplus \gamma_i.$$

**Note 2.** *According to the scheme [7] we store the values $E_i$, $i = \overline{0, m-1}$ in a neural network biometric container instead of the tables $\bar{u}$ and $\bar{w}$.*

The rows of the corresponding and the weight tables are successively decrypted according to the following rule:

1. For $i = 0$ we calculate the rows

$$u_0 = (E_0 \oplus \gamma_0)_{0 \sim bn-1}, \tag{4}$$

$$w_0 = (E_0 \oplus \gamma_0)_{bn \sim n(b+d)-1},$$

and form the bit $c_0$, where $\gamma_0$ is determined from the relations (1) and (2).

2. For $i \in \overline{1, m-1}$ we calculate the next rows using the bit sequence $(c_0, \ldots, c_{i-1})$ formed on the previous steps:

$$u_i = (E_i \oplus \gamma_i)_{0 \sim bn-1}, \tag{5}$$

$$w_i = (E_i \oplus \gamma_i)_{bn \sim n(b+d)-1},$$

where $\gamma_i$ is determined from the relations (1) and (3).

## 3 A recovering of the key bits

In this section we propose a method of a recovering of the password, the key sequence $\bar{c}$ and the encrypted tables $\bar{u}$ and $\bar{w}$ without using any input biometric data.

**Note 3.** *According to the standard [2] a neural network training satisfies the following conditions:*

1. *Any four successive rows of the table $\bar{u}$ consist of $4n$ different elements;*
2. *$\forall i, j \in \overline{0, N-1}$ a number of usages of $x_i$ differs from a number of usages of $x_j$ by no more than 2 in the table $\bar{u}$.*

### 3.1 A recovery algorithm

For rows of the table $\bar{u}$ let's define the following event

$$A_{i\sim j} = \left\{ \{u_i, \ldots, u_{jn-1}\} \bigcap \{u_{jn}, \ldots, u_{(j+1)n-1}\} = \varnothing \right\},$$

where $0 \leqslant i < j \leqslant m-1$. Then the next proposition is true.

**Proposition 1.** *Let $\bar{c} \in V_m$ is a sequence formed by the neural network [7] with $4n < N$. Let $\bar{c}' \in V_j$, $j \in \overline{1, m-1}$ is such sequence that $c'_i = c_i$, $i = \overline{0, j-2}$ for $j > 1$. Then*

$$\mathbf{P}\left\{A_{\max(0,j-3)\sim j}\right\} = \begin{cases} 1, & c'_{j-1} = c_{j-1}, \\ \frac{(N-\min(3,j)n)^{[n]}}{2^{nb}}, & c'_{j-1} \neq c_{j-1}. \end{cases}$$

*Proof.* According to item 2 of note 3 the result is obvious in the case $c'_{j-1} = c_{j-1}$. Let $c'_{j-1} = c_{j-1} \oplus 1$. Then the value $h\left(s\|p\|j+1\|c_0, \ldots, c_{j-1}, c'_j\right)$ for $j > 1$ and the value $h\left(s\|p\|1\|c'_0\right)$ for $j = 1$ are distributed equiprobable on the set of images of the hash function $h$ and therefore $u'_j$ also has an equiprobable distribution on the set of rows. Thus, according to [2] a number of possible variants of values for the rows $u'_j$ is $\left(2^b\right)^n$. Then for $4n < N$ we obtain

$$\mathbf{P}\{A_{0\sim 1}\} = \frac{A^n_{N-n}}{(2^b)^n} = \frac{(N-n)^{[n]}}{2^{nb}}, \; \mathbf{P}\{A_{0\sim 2}\} = \frac{A^n_{N-2n}}{(2^b)^n} = \frac{(N-2n)^{[n]}}{2^{nb}},$$

$$\mathbf{P}\{A_{j-3\sim j}\} = \frac{A^n_{N-3n}}{(2^b)^n} = \frac{(N-3n)^{[n]}}{2^{nb}}, \; j \geqslant 3. \; \square$$

**Example 1.** *For the biometric autentification scheme [4] with the parameters $N = 416$, $n = 16$*

$$\mathbf{P}\{A_{0\sim 1}\} \approx 1.4 \cdot 10^{-2}, \; \mathbf{P}\{A_{0\sim 2}\} \approx 7.3 \cdot 10^{-3}, \; \mathbf{P}\{A_{j-3\sim j}\} \approx 3.6 \cdot 10^{-3}, \; j \geqslant 3.$$

*For the scheme [7] with the parameters* $N = 416$, $n = 32$

$$\mathbf{P}\{A_{0\sim 1}\} \approx 2.7 \cdot 10^{-5},\ \mathbf{P}\{A_{0\sim 2}\} \approx 1.5 \cdot 10^{-6},\ \mathbf{P}\{A_{j-3\sim j}\} \approx 5,9 \cdot 10^{-8},\ j \geqslant 3.$$

Let's formulate an algorithm of password and key sequence recovering based on the proposition 1.

---

Input: an empty set.

Output: $p$, $(c_0, c_1, \ldots, c_{m-2})$.

1. Set $C_0 = \cdots = C_{m-2} = \varnothing$.
2. Randomly select $p \in V_r$ and calculate the row $u_0$ according to the rule (4).
3. Guess the pair of values $c_0 \in \{0, 1\}$ and calculate the rows $u_1$ according to the rule (5). If $\bigcap\limits_{j=0}^{1} u_j = \varnothing$, supplement $C_0$ by the corresponding bit $c_0$. If $C_0 = \varnothing$, go to step 2, otherwise – to step 4.
4. Guess the pair of values $c_1 \in \{0, 1\}$ and calculate the row $u_2$. If $\bigcap\limits_{j=0}^{2} u_j = \varnothing$, supplement $C_1$ by the corresponding bit $c_1$. If $C_1 = \varnothing$, go to step 2, otherwise – to step 5.
5. Set $i = 3$.
6. If $i \geqslant m$, the algorithm finishes its work, otherwise we guess the pair of values $c_{i-1} \in \{0, 1\}$ and calculate the row $u_i$. If $\bigcap\limits_{j=i-3}^{i} u_j = \varnothing$, supplement $C_{i-1}$ by the corresponding bit $c_{i-1}$.
   If $C_{i-1} = \varnothing$, go to step 2, otherwise set $i = i + 1$ and go to step 6.

---

**Note 4.** *Additionally we can organize a rejection of the bits* $c_i$ *using the features of the coding of the input parameters and item 2 of note 3.*

The algorithm 1 forms the set of possible key sequences:

$$C = \left\{(c_0, c_1, \ldots, c_{m-1})\,|c_i \in C_i,\ i = \overline{0, m-2}, c_{m-1} \in \{0, 1\}\right\}.$$

**Note 5.** *The results of [5] allow to define the last bit* $c_{m-1}$ *and the vector of input biometric parameters* $x_0, \ldots, x_{N-1}$.

### 3.2 Characteristics of the recovery algorithm

We can consider the tree for the algorithm 1 vertices of which correspond to constructed sequences. A root $u_0$ of the tree is formed as a result of password $p$ search. Other vertices are formed successively from layer to layer. The set $C$ contains only sequences corresponding to branches of a length $m - 1$.

Let $q_i = \mathbf{P}\left\{A_{\max(0,i-3)\sim i}\right\}$, $i \in \overline{1, m-1}$. It should be noted that $q_i = q_3$ for $i \geqslant 3$.

Further we consider subtrees corresponding to cases of true (Fig. 2) and false (Fig. 3) values $p$. The subtree vertices correspond to the values of the testing bits $c_0, c_1, \ldots, c_{m-2}$. At the same time, the vertices of the $i$-th layer are formed with probabilities $q_i$ at the $i$-th step of the algorithm 1, respectively.

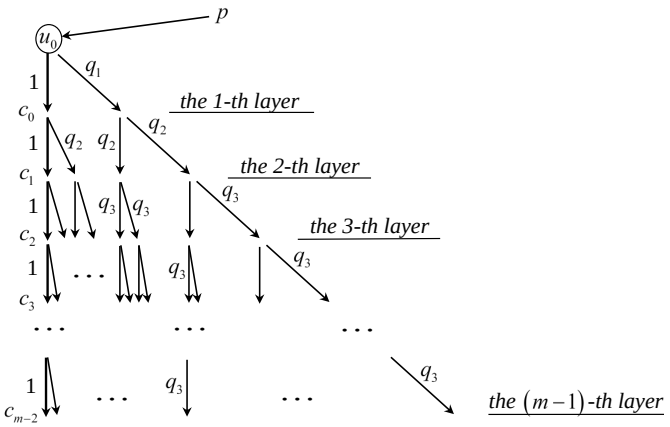



Figure 2: The scheme of the algorithm 1 for a true $p$

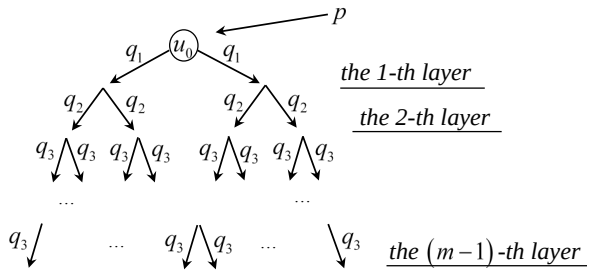



Figure 3: The scheme of the algorithm 1 for a false $p$

Let's assume the success of the algorithm 1 consists of the forming of single desired values of the password $p$ and the sequence $c_0, c_1, \ldots, c_{m-2}$.

Let $P_1$ is the probability of the success of the algorithm 1. Then by the law of total probability we obtain

$$P_1 = \tfrac{1}{2^r}\mathbf{P}\left\{success|p\ is\ true\right\} + \left(1 - \tfrac{1}{2^r}\right)\mathbf{P}\left\{success|p\ is\ false\right\}.$$

For a true $p$ the desired sequence is in the set $C$ with probability 1. In this case the success of the algorithm 1 consists of a rejection of all other variants of the sequence $c_0, c_1, \ldots, c_{m-2}$. It means the $(m-1)$-th layer of this subtree doesn't have any vertices corresponding to the false sequences $c_0, c_1, \ldots, c_{m-2}$.

Let $B_i$, $i \in \overline{1, m-1}$ is the event that for the true $p$ the $i$-th layer of the tree has some vertices. Then $\mathbf{P}\left\{success|p\ is\ true\right\} = 1 - \mathbf{P}\left\{B_{m-1}\right\}$.

Further let $P^{(i)}$ is the probability of the binary tree of a height $i$ (Fig. 4), vertices which are formed with a probability $q_3$.

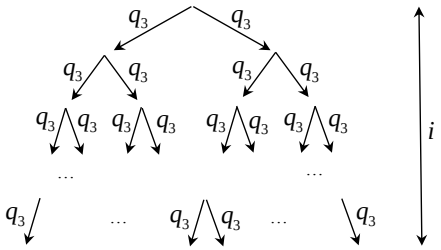



Figure 4: The binary tree of a height $i$ with a probability $q_3$

In this case the following relations are true:

$$\begin{cases} P^{(0)} = 1, \\ P^{(1)} = 2q_3 - q_3^2, \\ P^{(i)} = \left(2\left(q_3 - q_3^2\right) + q_3^2 P^{(i-1)}\right) P^{(i-1)}, i \geqslant 2. \end{cases}$$

Therefore,

$$\begin{gathered} \mathbf{P}\{B_1\} = q_1; \\ \mathbf{P}\{B_2\} = q_2 + q_1\left(2q_2 - q_2^2\right); \\ \mathbf{P}\{B_i\} = q_3 \sum_{j=0}^{i-2} P^{(j)} + \left(q_2 + 2q_1\left(q_2 - q_2^2\right) + q_1 q_2^2 P^{(i-1)}\right) P^{(i-1)}, i \geqslant 2. \end{gathered} \tag{6}$$

For a false $p$ the success of the algorithm 1 consists of a rejection of all generated sequences $c_0, c_1, \ldots, c_{m-1}$.

Let $D_i$, $i \in \overline{1, m-1}$ is the event that for a false $p$ the $i$-th layer of the corresponding subtree has some vertices. In this case $\mathbf{P}\{success | p\ is\ false\} = 1 - \mathbf{P}\{D_{m-1}\}$, where

$$\begin{gathered} \mathbf{P}\{D_1\} = 2q_1 - q_1^2; \\ \mathbf{P}\{D_2\} = 2\left(q_1 - q_1^2\right)\left(2q_2 - q_2^2\right) + q_1^2\left(2q_2 - q_2^2\right)^2; \\ \mathbf{P}\{D_i\} = 2\left(q_1 - q_1^2\right)\left(2\left(q_2 - q_2^2\right) P^{(i-1)} + q_2^2\left(P^{(i-1)}\right)^2\right) + \\ + q_1^2\left(2\left(q_2 - q_2^2\right) P^{(i-1)} + q_2^2\left(P^{(i-1)}\right)^2\right)^2, i \geqslant 2. \end{gathered} \tag{7}$$

Therefore, we obtain the equality for the probability of the success:

$$P_1 = \frac{1}{2^r}\left(1 - \mathbf{P}\left\{D_{m-1}\right\}\right) + \left(1 - \frac{1}{2^r}\right)\left(1 - \mathbf{P}\left\{B_{m-1}\right\}\right),$$

where $\mathbf{P}\left\{B_{m-1}\right\}$, $\mathbf{P}\left\{D_{m-1}\right\}$ are determined from the relations (6) and (7), respectively.

**Note 6.** *The algorithm 1 doesn't require an additional rejection of false variants of the sequences $c_0, c_1, \ldots, c_{m-2}$ with probability $P_1$.*

The approximate values of $P_1$ for the schemes [4], [7] for some password length values in the case $m = 256$ are presented in table 2.

| $r$ | **4** | **6** | **8** | **10** | **12** |
|---|---|---|---|---|---|
| $P_1$ for [4] | $1 - 2.3 \cdot 10^{-4}$ | $1 - 5.7 \cdot 10^{-5}$ | $1 - 1.4 \cdot 10^{-5}$ | $1 - 3.5 \cdot 10^{-6}$ | $1 - 8.9 \cdot 10^{-7}$ |
| $P_1$ for [7] | $1 - 3.7 \cdot 10^{-9}$ | $1 - 9.2 \cdot 10^{-10}$ | $1 - 2.3 \cdot 10^{-10}$ | $1 - 5.8 \cdot 10^{-11}$ | $1 - 1.4 \cdot 10^{-11}$ |

Table 2: The values of $P_1$ for the schemes [4], [7].

Let $T_1$ is the complexity of the algorithm 1. Then the following statements are true:

– In the case of a true $p$ the minimum of $T_1$ is $2\,(m-1)$ hash function evaluation with the probability

$$\mathbf{P}\left\{T_1 = 2\,(m-1)\right\} = (1-p_1)\,(1-p_2)\,(1-p_3)^{m-3}.$$

– In the case of a false $p$ the value $\mathbf{P}\left\{T_1 = 2k\right\}$ is inversely proportional to $k$. Herewith,

$$\begin{aligned}\mathbf{P}\left\{T_1 \leqslant 6\right\} &= \mathbf{P}\left\{T_1 = 2\right\} + \mathbf{P}\left\{T_1 = 4\right\} + \mathbf{P}\left\{T_1 = 6\right\} = \\ &= (1-q_1)^2 + 2\left(q_1 - q_1^2\right)\left((1-q_2)^2 + 2\left(q_2 - q_2^2\right)(1-q_3)^2\right) + \\ &\qquad + q_1^2(1-q_2)^4.\end{aligned}$$

For example, for the scheme [4] $\mathbf{P}\left\{T_1 \leqslant 6\right\} \approx 0.999985$, and for the scheme [7] $\mathbf{P}\left\{T_1 \leqslant 6\right\} \approx 1 - 7.9 \cdot 10^{-14}$.

In the worst case a number of checks of $p$ in the algorithm 1 is $2^r$. Thus, using the independence of checks of bits $c_0, c_1, \ldots, c_{m-2}$ for different values of $p$ we obtain

$$\begin{aligned}\mathbf{P}\left\{T_1 \leqslant 2m + 3\left(2^{r+1} - 1\right)\right\} &> (1-p_1)\,(1-p_2)\,(1-p_3)^{m-3} \times \\ &\times \left[(1-q_1)^2 + 2\left(q_1 - {q_1}^2\right)\left((1-q_2)^2 + 2\left(q_2 - {q_2}^2\right)(1-q_3)^2\right) + q_1^2(1-q_2)^4\right]^{2^r-1}.\end{aligned}$$

The approximate values of the probability $p_{\min}(m,r) = \mathbf{P}\left\{T_1 \leqslant 2m + 3\left(2^{r+1}-1\right)\right\}$ for the schemes [4], [7] in the case of the maximum number of steps of the algorithm 1 are presented in table 3.

| $r$ | **4** | **6** | **8** | **10** | **12** | **14** | **16** |
|---|---|---|---|---|---|---|---|
| $p_{\min}(m,r)$ for [4] | 0.3901 | 0.3898 | 0.3887 | 0.3842 | 0.3669 | 0.3047 | 0.1451 |
| $p_{\min}(m,r)$ for [7] | 0.9999 | 0.9999 | 0.9999 | 0.9999 | 0.9999 | 0.9999 | 0.9999 |

Table 3: The probability $p_{\min}(m,r)$ for the schemes [4], [7]

## Conclusions

The proposed method of the key information recovering does not require a knowledge of the biometric data and allows to restore all parameters determining the neural network, including the biometric template. The obtained results showed the key sequence recovering is equivalent to the password recovering.

It should be noted the protection scheme of a neural network container [7] regardless of the used encryption rules (5) does not provide any protection to the proposed method. Therefore, protection schemes like [7] are unsafe.

## References


[1] GOST R 52633.0-2006. Information protection. Information protection technology. Requirements to the means of high-reliability biometric authentication [in Russian].

[2] GOST R 52633.5-2011. Information protection. Information protection technology. The neural net biometry-code convertor automatic training [in Russian].

[3] GOST R 34.11-2012. Information technology. Cryptographic data security. Hash function.

[4] Efimov O. V., Funtikov V. A., Jazov U. K. The neuronet converter "biometry-code" structure and interconnections choice strategy, Neurocomputers: development, application, v. 6, 2009, pp. 14-16 [in Russian].

[5] Marshalko G. B. On the security of a neural network-based biometric authentication scheme, Math. Asp. of Cryptogr., 5:2 (2014), 87-98.

[6] Nazarov I. G., Efimov O. V., Yazov Y. K. The package of national standards, ensuring biometric and neural network protection of mass circulated personal data privacy, Neurocomputers: development, application, v. 3, 2012, pp. 9-16 [in Russian].

[7] Technical specification (project). Neural network container protection using cryptographic algorithms. http://dissov.pnzgu.ru/files/dissov.pnzgu.ru/2017/tech/kolchugina/spisok_trudov_veduschey_org.pdf [in Russian].

[8] Dodis Y., Reyzin L., Smith A. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. Lecture Notes in Computer Scince, v. 3072, 2004, pp. 523-540.

[9] Boyen X., Reusable fuzzy extractors, CCS'04 Proceedings of the 11th ACM conference on Computer and communications security, pp. 82-91.

[10] Kevenaa T., Skoric B., Tuyls P., Security with Noisy Data: On Private Biometrics, Secure Key Storage and Anti-Counterfeiting Springer Science and Business Media, 2007, 300 p.

# Testing the NIST Statistical Test Suite on Artificial Pseudorandom Sequences

Andrey Zubkov and Aleksandr Serov

Steklov Mathematical Institute of the Russian Academy of Sciences, Moscow, Russia
{zubkov, serov}@mi.ras.ru

**Abstract**

We discuss the results of experiments with the well-known NIST Statistical Test Suite designed for testing the hypothesis on the uniformity and independence of binary sequence elements. In particular, we investigate conditions on the parameters of piecewise merging of two linear recurrent sequences under which such combined sequences successfully pass all tests of the NIST package.

**Keywords:** Bernoulli sequence, random equiprobable sequence, statistical tests, pseudo-random number sequence generators.

## 1 Introduction

Generators of random and pseudo-random sequences are used in many fields of science and technology, including the cryptography. The most strict conditions on the quality of generated sequences are used in cryptography: to ensure the information security it is necessary for the generated sequences to be indistinguishable (or to be difficult to distinguish) from the equiprobable Bernoulli sequences. So the development and investigation of methods to test the closeness of the binary sequence properties to the properties of the equiprobable Bernoulli sequence is an actual problem.

There are two basic types of generators used to generate random sequences: random number sequence generators (RNG) and pseudo-random number sequence generators (PRNG).

The random number generators of the first type (see, for example, [7]) use some nondeterministic sources of randomness (see, for example, [9]) to generate the intermediate nondeterministic sequence that serves as the input of a deterministic device transforming it into resulting irreproducible sequence (if there is no such device and the intermediate sequence is used as the output, then the generator is classified as nondeterministic).

One of cryptographic properties that an output sequence of the RNG should have is unpredictability. But physical sources of randomness may generate sequences which are not equiprobable and are predictable to some extent.This deficiency of physical sources may be reduced by combining outputs from sources of various types. Nevertheless RNG combining several sources of randomness may produce output sequences having properties that differ from that of a sequence with independent equiprobable elements. From the other hand, physical RNG which generates random sequences with high cryptographic properties may be quite slow.

The PRNGs use the randomness sources only during the initialization phase to generate the initial state of the PRNG. The output sequence of the PRNG is a deterministic function of its initial state and parameters chosen by user, so its randomness is limited by the randomness of the initial state. It's curious that some statistical properties of the output sequence of a cryptographically secure PRNG may be higher than that of a random sequence generated by physical source of randomness.

## 2 Statistical test packages

In practice the testing of statistical properties of random sequences (the property is the higher the closer the characteristics of the tested sequence are to the characteristics of random equiprobable sequence) begins with the application of some statistical test packages. There are several popular packages of statistical tests which are distributed with open source codes (e. g. TESTU01 see [8], DIEHARD see [3, 4], NIST see [6], SPRNG see [5]), or with closed source codes (e. g. Crypt-X http://www.isrc.qut.edu.au/resource/cryptx/). These packages allow to perform the analysis and testing of random sequences and have significant intersections in the sets of tests.

## 3 Main results

From the statistical test packages listed above, the NIST statistical tests package was selected as one of the most popular, fully documented and actively used for generator certifications.

The NIST Statistical Test Suite consists of 15 tests «developed for the randomness testing of the binary sequences» (word-for-word from the manual). These 15 tests are listed in Table 1.

| Number | Test Name |
|---|---|
| 1 | Frequency |
| 2 | Block Frequency |
| 3 | Runs |
| 4 | Longest Run |
| 5 | Binary Matrix Rank |
| 6 | Discrete Fourier Transform |
| 7 | Non-overlapping Template Matching |
| 8 | Overlapping Template Matching |
| 9 | Universal |
| 10 | Linear Complexity |
| 11 | Serial |
| 12 | Approximate Entropy |
| 13 | Cumulative Sums |
| 14 | Random Excursions |
| 15 | Random Excursions Variant |

Table 1: List of NIST Statistical Tests

For testing the sequence it is divided into several sufficiently long blocks and for each statistical test a set of P-values corresponding to these blocks are produced. The sequence is considered as *accepted by the test* if the corresponding P-values look like independent random variables with the uniform distribution on $[0, 1]$, in particular, exceed the fixed significance level $\alpha$, and is considered as *rejected by the test* otherwise.

The test parameters we have used are listed in Table 2.

| Test Name | Block Length |
|---|---|
| Block Frequency | 128 |
| Longest Run | 10000 |
| Binary Matrix Rank | 1024 |
| Non-overlapping Template Matching | 9 |
| Overlapping Template Matching | 9 |
| Universal (Initialization Steps) | 11 (20480) |
| Linear Complexity | 500 |
| Serial | 16 |
| Approximate Entropy | 10 |

Table 2: Parameters used for NIST Statistical Test Suite

The critical values of statistics in the NIST Statistical Test Suite were computed by means of limit theorems, and it was recommended that analysed sequences should have sufficiently large lengths. All segments sequences that we have tested were of $2^{25} - 1$ bit length.

## 4 Description of our experiments

In this section we describe types of pseudo-random sequences tested by the NIST Test Suite and results of tests. The significance level $\alpha = 0.01$ determining the rule of acceptance/rejection of the hypothesis was selected by default.

### 4.1 Testing pseudo-random sequences generated by linear shift registers of the maximal period

Pseudo-random sequences of the maximal period were generated by the linear shift registers with feedbacks given by the following primitive polynomials of degrees 25 and 27 over GF(2):

$$\begin{aligned} f(x) &= x^{25} + x^3 + 1, \\ g(x) &= x^{27} + x^5 + x^2 + x + 1, \\ h(x) &= x^{27} + x^{19} + x^{18} + x^{17} + x^{11} + x^6 + 1, \\ m(x) &= x^{27} + x^{26} + x^{25} + x^{24} + x^{23} + x^{22} + x^{21} + x^{20} + x^{19} + x^{17} \\ &\quad + x^{15} + x^{13} + x^{11} + x^9 + x^7 + x^5 + x^3 + x + 1. \end{aligned}$$

Initial states of all linear shift registers were chosen to have only one nonzero bit, namely the most significant one. The value $33,554,431 = 2^{25} - 1$ was selected as the length of the segments of tested sequences, thus each of the registers with the polynomials $g(x)$, $h(x)$ and $m(x)$ was used to perform four tests on disjoint segments of an output sequence of length $2^{27} - 1$. These sequences were chosen to simplify the problem of detecting their nonrandomness.

The segments of the pseudorandom sequences obtained by the linear shift registers with the $g(x)$, $h(x)$ and $m(x)$ polynomials have successfully passed all the tests from the NIST Test Suite except for The Binary Matrix Rank Test (ranks of binary matrices of the size $32 \times 32$), The Discrete Fourier Transform (Spectral) Test and The Linear Complexity Test, where the $P$-values were less than $10^{-6}$, with the significance level $\alpha = 0.01$. The full-period sequence corresponding to the polynomial $f(x)$, in addition to the listed tests, did not pass the Tests for the Longest Run-of-Ones in a Block ($P$-value $6 \cdot 10^{-6}$) and Maurer's «Universal Statistical» Test ($P$-value $1.91 \cdot 10^{-4}$). (Results of testing are collected in Table 3 below.)

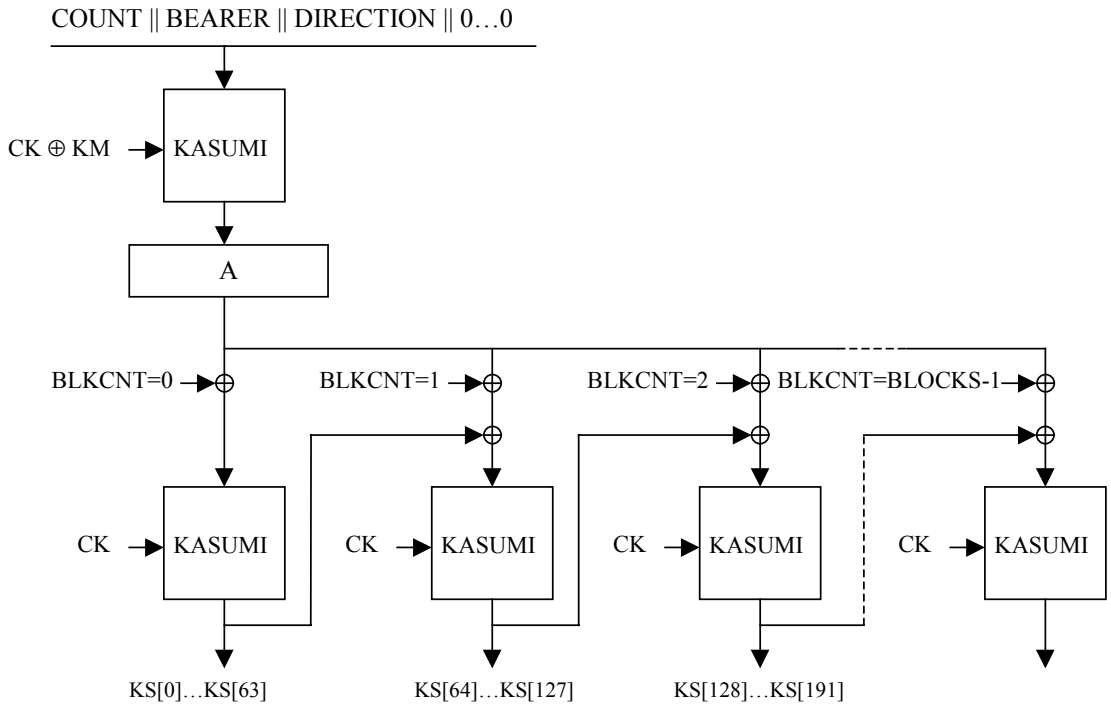



Figure 1: The generator of the key stream **A8**

## 4.2 Testing pseudo-random sequences generated by the linear shift registers with additive noise

The additive noise applied to the output sequences of linear shift registers given by the primitive polynomials $f(x)$, $g(x)$, $h(x)$ and $m(x)$ over GF(2) was produced by means of KASUMI block cipher used in the A8 algorithm. The detailed description of A8 algorithm may be found in [10].

The values of noise added to bits of the original sequence were determined by the corresponding 64-bit output KS values of the KASUMI block cipher (see Fig. 1). KASUMI has 128-bit key CK which is used in the 8-round Feistel scheme. For example, to obtain the $i$-th noise bit taking value 1 with probability $\frac{1}{4}$ we compare the output 64-bit KS value of the KASUMI encryption algorithm corresponding to the value BLKCNT $= i$ with $2^{62}$.

The only test from the NIST Test Suite that detects nonrandomness in the disjoint $2^{25} - 1$ bit segments of output sequences of linear shift registers with polynomials $f(x)$, $g(x)$, $h(x)$, $m(x)$ perturbed by the Bernoulli noise sequence with parameter $\frac{1}{4}$, turned out to be The Discrete Fourier Transform (Spectral) Test; the corresponding $P$-values were smaller $10^{-6}$.

The increasing of the noise parameter of the Bernoulli sequence from $\frac{1}{4}$ to $\frac{3}{8}$ results in sequences which pass **all tests** from the NIST Test Suite (with the exception of some sequences corresponding to the polynomial $f(x)$).

### 4.3 Testing of filtered output sequences of linear shift registers of the maximal period

To filter output sequences of linear shift registers given by primitive polynomials $f(x)$, $g(x)$, $h(x)$ and $m(x)$ over GF(2) we use the balanced Boolean function corresponding to the most significant bit of the nonlinear substitution SubBytes $S = (s_1, s_2, \ldots, s_8)\colon \mathrm{GF}(2)^8 \to \mathrm{GF}(2)^8$ of the AES symmetric block cipher algorithm. The values of the first six arguments of $s_1$ were determined by bits 1, 3, 9, 13, 17, 21 for all registers, the remaining 2 arguments were determined by bits 23, 25 of the register with polynomial $f(x)$, and by bits 25, 27 of for other registers.

Filtered sequences failed to pass a number of tests of NIST Test Suite (see Table 3), and corresponding $P$-values in many cases were smaller than $10^{-6}$.

Changing the set of arguments of the filter function $s_1$ from the mentioned above to $\{1, 2, 3, 5, 8, 12, 17, 23\}$, $\{1, 3, 9, 14, 17, 21, 22, 24\}$ and $\{1, 3, 9, 14, 17, 21, 24, 26\}$ had no significant impact on the experimental results, see Table 3.

### 4.4 Testing of pseudo-random sequences obtained by merging of outputs of two linear shift registers of maximal periods

We have considered two types of pseudo-random sequences consisted of segments of two binary recurrent sequences generated by linear shift registers with feedbacks defined by two primitive polynomials.

A) The output sequence of the first register $\{x_1, x_2, \ldots\}$ corresponding to the polynomial $f(x)$ was divided into adjacent segments of $L_1 = 25$ bits, the output sequence of the second register $\{y_1, y_2, \ldots\}$ corresponding to the polynomial $g(x)$ was similarly divided into segments of $L_2 = 27$ bits. Further, the tested sequence $\{z_1, z_2, \ldots\}$ of the first type was constructed by merging of obtained segments of two sequences:

$$\{z_k\}_{k=0}^{2^{L_1}-1+\left[\frac{2^{L_1}-1}{L_1}\right]L_2} = \{x_1, \ldots, x_{L_1}, y_1, \ldots, y_{L_2}, x_{L_1+1}, \ldots, x_{2L_1}, y_{L_2+1}, \ldots\}.$$

B) The output register sequences were divided into adjacent segments of a variable lengths according to the following rule:

- the first segment of the first register output sequence consists of $L_1 = L_1^* = 25$ sequential output bits,
- the first segment of the second register output sequence consists of

$$L_2^* = 16 + 2^3 x_{L_1^*-3} + 2^2 x_{L_1^*-2} + 2x_{L_1^*-1} + x_{L_1^*}$$

  bits (a fixed value 16 was increased by the integer formed by the last 4 bits of the already constructed sequence),
- the second segment of the first register consists of

$$L_1^* = 16 + 2^3 y_{L_2^*-3} + 2^2 y_{L_2^*-2} + 2y_{L_2^*-1} + y_{L_2^*}$$

  bits (a fixed value 16 was increased by the integer formed by the last 4 bits of the already constructed sequence),
- and so on.

Consequently the tested sequence $\{w_1, w_2, \ldots\}$ of the second type had the form

$$\{x_1, \ldots, x_{L_1}, y_1, \ldots, y_{L_2^*}, x_{L_1+1}, \ldots, x_{L_1+L_1^*}, y_{L_2^*+1}, \ldots\}.$$

The testing of these sequences shows that the first type sequences $\{z_1, z_2, \ldots\}$ had passed all tests with the exception of Discrete Fourier Transform (Spectral) Test: for this test $P$-values were smaller than $10^{-6}$, while the second type sequence had passed all the tests with $P$-values being as a rule essentially larger than $\alpha = 0.01$. The second type sequences with

$$L_2^* = 32 + 2^5 x_{L_1^*-5} + 2^4 x_{L_1^*-4} + 2^3 x_{L_1^*-3} + 2^2 x_{L_1^*-2} + 2x_{L_1^*-1} + x_{L_1^*}$$
$$L_1^* = 32 + 2^5 y_{L_2^*-5} + 2^4 y_{L_2^*-4} + 2^3 y_{L_2^*-3} + 2^2 y_{L_2^*-2} + 2y_{L_2^*-1} + y_{L_2^*}$$

did not pass only the Overlapping Template Matching test.

The second type sequences with polynomial $f(x)$ of the first register $\{x_1, x_2, \ldots\}$ replaced by the polynomial

$$F(x) = 1 + x^2 + x^5 + x^6 + x^8 + x^{10} + x^{13} + x^{14} + x^{16} + x^{18} + x^{21} + x^{22} + x^{24} + x^{26} + x^{29} + x^{30} + x^{32}$$

successfully passed all the tests. Increasing the mean values of $L_1^*$ and $L_2^*$ up to 128 didn't change the result, but after additional replacement of the feedback polynomial $f(x)$ of the first register by the polynomial $h(x)$ some sequences had failed to pass the Overlapping Template Matching test, while all other tests as before were passed.

### 4.5 Testing of the pseudo-random sequence generated by AES

The tested pseudorandom sequence was obtained by iterative application of the AES block cipher algorithm to the zero plain text with a 128-bit key in the cipher block chaining mode with initialization 128-bit vector all bits of which are nonzero except for the 7 lower bits; the key bits were fixed by zero and did not change during the iterative calculations. Each byte of the encrypted sequence was replaced by the corresponding bit depending on the byte value.

Four non-overlapping segments of the length $2^{25} - 1$ bits of the initial sequence of the length $2^{27} - 4$ bits passed all the tests from the NIST Test Suite in the aggregate, except for The Serial Test for the second segment only, where the $P$-value of one of two statistics turned out to be slightly less than the significance level $\alpha = 0.01$, namely 0.008415.

| Testing | LSR | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LSR output | $f$ | + | + | + | − | − | − | + | + | − | − | − | + | + | + | + |
| | $g$ | + | + | + | + | − | − | + | + | + | − | + | + | + | + | + |
| | $h$ | + | + | + | + | − | − | + | + | + | − | ± | + | + | + | + |
| | $m$ | + | + | + | + | − | − | + | + | + | − | + | + | + | + | + |
| LSR + additive noise with parameter 1/4 | $f$ | + | + | + | + | + | − | + | + | + | + | + | + | + | + | + |
| | $g$ | + | + | + | + | + | − | + | + | + | + | + | + | + | + | + |
| | $h$ | + | + | + | + | + | − | + | + | + | + | + | + | + | + | + |
| | $m$ | + | + | + | − | + | − | + | + | + | + | + | + | + | + | + |
| with parameter 3/8 | $f$ | + | + | + | + | + | ± | + | + | + | + | + | + | + | − | − |
| | $g$ | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| | $h$ | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| | $m$ | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| 1, 3, 9, 13, 17, 21,23, 25 | $f$ | + | − | − | − | + | − | − | − | − | + | − | − | + | + | + |
| filtered LSR arguments of $f_1$ 1, 3, 9, 13, 17, 21,25, 27 | $g$ | + | − | + | + | + | − | − | + | − | + | − | − | + | + | + |
| | $h$ | + | − | + | + | + | + | − | + | − | + | − | − | + | + | + |
| | $m$ | + | − | + | + | + | − | − | + | − | + | − | − | + | + | + |
| 1, 3, 9, 14, 17, 21,22, 24 | $f$ | + | + | + | + | + | − | − | − | + | + | − | − | + | + | + |
| filtered LSR arguments of $f_1$ 1, 3, 9, 14, 17, 21,23, 26 | $g$ | + | + | + | + | + | − | − | + | + | + | − | − | + | + | + |
| | $h$ | + | − | + | + | + | ± | − | − | + | + | − | − | + | + | + |
| | $m$ | + | − | + | + | + | − | − | − | + | + | − | − | + | + | + |
| merging of segments with average length | $f, g,$ 24 | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| | $f, g,$ 64 | + | + | + | + | + | + | + | − | + | + | + | + | + | + | + |
| | $f, g,$ 96 | + | + | + | + | + | + | + | − | + | + | + | + | + | + | + |
| | $F, g,$ 64 | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| | $F, g,$ 128 | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| | $h, g,$ 128 | + | + | + | + | + | + | + | ± | + | + | + | + | + | + | + |
| fixed length 25, 27 | $f, g$ | + | + | + | + | + | − | + | + | + | + | + | + | + | + | + |
| PRS | AES | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| PRS obtained by shrinking | $g, f$ | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| | $g, h$ | + | + | + | + | + | + | + | + | + | + | − | + | + | + | + |

Table 3: Experimental results

### 4.6 Testing the output sequence of shrinking generators composed of two linear shift registers

Two tested sequences were obtained by extracting from the output sequence of the first linear shift register (with feedback polynomial $g(x)$) all bits corresponding to the nonzero bits in the output sequence of the second linear shift register (with feedback polynomial $f(x)$ for the first type test sequence and the polynomial $h(x)$ for the second). In initial states of all linear shift registers all bits were zero except the higher-order bit.

The first type sequence passed all the tests from the NIST Test Suite with the significance level $\alpha = 0.01$. The second type sequence passed all the tests except for the Serial Test: for this test $P$-values turned out to be smaller than $10^{-6}$. Maybe this is the consequence of coincidence of orders of the source and control sequences.

Table 3 shows the results of almost all performed experiments.

## 5 Conclusions

The set of experiments with different non-random pseudo-random sequences showed that the NIST Test Suite may detect some deviations of properties of analyzed sequences from that of ideal Bernoulli sequences, but may fail to detect non-randomness of deterministic sequences with not very complex artificial irregularities (see subsection 4.4).

## References


[1] Knuth D. The Art of Computer Programming, volume 2: Seminumerical Algorithms, Addison-Wesley, 688 pp. 1969.

[2] Maurer U. A universal statistical test for random bit generators. J. Cryptology, 5 (2), pp. 89–105. 1992.

[3] `http://stat.fsu.edu/pub/diehard/`.

[4] Marsaglia G. DIEHARD: a battery of tests of randomness. `http://stat.fsu.edu/~geo/diehard.html`. 1996.

[5] Mascagni M., Srinivasan A. Algorithm 806: SPRNG: A scalable library for pseudorandom number generation. ACM Trans. Math. Soft., 26, pp. 436–461. 2000.

[6] Rukhin A., Soto J., Nechvatal J., Smid M., Barker E., Leigh S., Levenson M., Vangel M., Banks D., Heckert A., Dray J., Vo S. A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications. NIST Special Publication 800-22 Revision 1a, 27 April 2010.

[7] Barker E., Kelsey J. Recommendation for random bit generator (RBG) constructions DRAFT NIST SP 800-90C. http://csrc.nist.gov/publications/drafts/800-90/draft-sp800-90c.pdf. August 2012.

[8] L'Ecuyer P., Simard R. TestU01. D'epartement d'Informatique et de Recherche Op'erationnelleUniversit'e de Montr'eal, 214 pp., http://simul.iro.umontreal.ca/testu01/guideshorttestu01.pdf. 2013.

[9] Barker E., Kelsey J. Recommendation for the entropy sources used for random bit generation. NIST DRAFT Special Publication 800-90B. http://csrc.nist.gov/publications/PubsDrafts.html\#800-90B. January 27, 2016.

[10] 3GPP 3rd Generation Partnership Project; Technical Specification Group Services and system Aspects; Security related network functions (Release 9). http://www.3gpp.org/ftp/Specs/archive/.

[11] Universal Mobile Telecommunications System (UMTS); Specification of the 3GPP confidentiality and integrity algorithms; Document 2: Kasumi specification. http://www.etsi.org/website/document/algorithms.