# On zero practical significance of “Key recovery attack on full GOST block cipher with zero time and memory”

Vladimir Rudskoy[†]

[†]*Lomonosov Moscow State University,*

*Faculty of Computational Mathematics and Cybernetics.*

`rudskoy_vladimir@mail.ru`

**Abstract**

In this paper we show that the related key boomerang attack by E. Fleischmann et al. from the paper mentioned in the title does not allow to recover the master key of the GOST block cipher with complexity less than the complexity of the exhaustive search. Next we present modified attacks. Finally we argue that these attacks and the related key approach itself are of extremely limited practical applications and do not represent a fundamental obstacle to practical usage of the block ciphers such as GOST, AES and Kasumi.

## 1 Introduction

Recently there was a host of papers devoted to methods of cryptanalysis, which make use of related keys. In particular, related-key attacks were developed for block ciphers GOST [3], AES [1] and Kasumi [2]. These methods have complexity essentially smaller than the exhaustive search. At the same time, they use very strong assumption of possibilities of the attacker. For practical implementations of the ciphers one can consider such an assumption as extremely improbable.

In [3] the related-key key recovery attack on full GOST block cipher was presented. The authors claim their algorithm allows to recover 8 bits of a master key of the cipher, other bits are recovered by the exhaustive search.

In the present work we investigate in detail the attack from [3]. We show that this attack does not allow to recover the master key of the key of the GOST block cipher with complexity less than the complexity of the exhaustive search. More precisely, we prove it is equivalent to the exhaustive search. Moreover, in [3] the fixed set of S-boxes is considered, and the attack is applicable not to all sets of S-boxes. At the same time, we show how to change the algorithm from [3] in such a way that it will recover 31 bits of a master key regardless of a choice of S-boxes. Then the generalised algorithm will be presented, which allows to recover more bits of a key, and in certain cases fully recover master key, with complexity essentially smaller than the exhaustive search, provided that there are enough related keys.

## 2 Notations and the description of the GOST block cipher

In this paper the following notations are used. For a natural $t$ a set of bit words of length $t$ is denoted by $V_t$. $\oplus$ denotes bitwise XOR of words. Bits are numbered 0 through $t-1$ from right to left: $X = (x_{t-1}, ..., x_0) \in V_t$, $x_i \in \{0, 1\}$, $i = \overline{0, t-1}$. High order bits have greater numbers.

Let $X[i \sim j]$ denote a word consisting of bits of $X$ in positions from $i$-th to $j$-th.

Each word in $V_{32}$ corresponds to a natural number

$$X = (x_{31}, ..., x_0) \mapsto \sum_{i=0}^{31} x_i 2^i.$$

$\boxplus$ denotes addition modulo $2^{32}$ of two words considered as natural numbers.

$L : V_{32} \mapsto V_{32}$ denotes bit rotation of $x \in V_{32}$ by 11 positions to the left.

$$L(x) = L((x_{31}, ..., x_0)) = (x_{20}, x_{19}, ..., x_0, x_{31}, ..., x_{22}, x_{21}).$$

Nonlinear transformation $\Pi : V_{32} \mapsto V_{32}$ consists of 8 parallel 4-bit wide bijective S-boxes $\Pi = (\pi_0, ...\pi_7)$, $\pi_k : V_4 \mapsto V_4,\ k = \overline{0,7}$. The first S-box ($\pi_0$) is applied to the most significant bits, the last S-box ($\pi_7$) is applied to the least significant bits:

$$\Pi(x) = (\pi_0(x_{31}, x_{30}, x_{29}, x_{28}), ..., \pi_7(x_3, x_2, x_1, x_0))$$

In GOST 28147-89 S-boxes are not specified and may be used as a long-term key.

GOST is a block 32-round Feistel cipher. It uses a 64-bit information block and a 256-bit master key. Plaintext block is divided into 32-bit left and right parts $P_0 = L_0||R_0$. The corresponding ciphertext $P_{32} = L_{32}||R_{32}$. Round function is computed according to

$$\begin{cases} L_i = R_{i-1} \\ R_i = L_{i-1} \oplus L(\Pi(R_{i-1} \boxplus k_i)) \end{cases}$$

for $i = \overline{1,31}$ and

$$\begin{cases} R_{32} = R_{31} \\ L_{32} = L_{31} \oplus L(\Pi(R_{31} \boxplus k_{32})), \end{cases}$$

where $k_i$ denotes the $i$-th round subkey. Master key $K$ is divided into 32-bit subkeys $K = (K_1, ..., K_8)$. The key schedule produces round keys as follows:

$$\begin{cases} k_i = K_{(i-1) \mod 8+1}, & i \in \overline{1,24}; \\ k_i = K_{32-i+1}, & i \in \overline{25,32}. \end{cases}$$

for encryption and

$$\begin{cases} k_i = K_i, & i \in \overline{1,8}; \\ k_i = K_{(32-i) \mod 8+1}, & i \in \overline{9,32}. \end{cases}$$

for decryption.

$E(P_0, K)$ denotes an encryption of $P_0$ under key $K$, $E^{-1}(P_{32}, K)$ denotes a decryption.

## 3 The related-key boomerang attack

The boomerang attack was first published in [4]. The attack is an extension to differential cryptanalysis that uses adaptive chosen plaintexts and ciphertexts. To describe this attack we need some definitions.

Let $F : V_n \to V_n$ denote some non-linear transformation of $V_n$.

**Definition 1** *$\alpha \to \beta$, $\alpha, \beta \in V_n$ is called a differential for $F$ if there exist an input pair $(P, P')$ with difference $\alpha$: $P \oplus P' = \alpha$, such that $\beta$ is the output difference,i.e. $F(P) \oplus F(P') = \beta$. The probability $p^F_{\alpha,\beta}$ is related to a differential, $p^F_{\alpha,\beta} = P\{F(x) \oplus F(x \oplus \alpha) = \beta\}$, assuming $x$ being randomly and independently distributed over $V_n$.*

**Definition 2** *A pair $(P, P' = P \oplus \alpha)$ is called a **correct** pair for the differential $\alpha \to \beta$, if it satisfies $F(P) \oplus F(P') = \beta$ and is called a **wrong** pair otherwise.*

We will also need the following proposition [4].

**Proposition 3** *Let $F$ be a bijective transformation, and let $\alpha \to \beta$ be a differential for $F$ with probability $p$. Then, for the inverse transformation $F^{-1}$ the differential $\beta \to \alpha$ also has probability $p$.*

Consider a block cipher $E(X, K) = Y$, where $X \in V_n$ is a plain text, $Y \in V_n$ is a ciphertext, $K \in V_k$ is a secret key. With fixed key, the encryption function is a bijective transformation over $V_n$, therefore statement 3 holds for it.

Suppose $E$ may be represented as a composition of two subciphers $E_0$ and $E_1$, $E = E_0 \circ E_1$.

Let $\alpha \to \beta$ be a differential for $E_0$ with probability $p$ and $\gamma \to \delta$ be a differential for $E_1$ with probability $q$. According to statment 3, the backward differentials $\beta \to \alpha$ for $E_0^{-1}$ and $\gamma \to \delta$ for $E_1^{-1}$ have the same probabilities $p$ and $q$ respectively.

The boomerang attack consists of two steps: the distinguisher step and the key recovery step. During the distinguisher step, an attacker tries to find correct plain text pairs for the differentials $E_0$ and $E_1$. These correct pairs are used

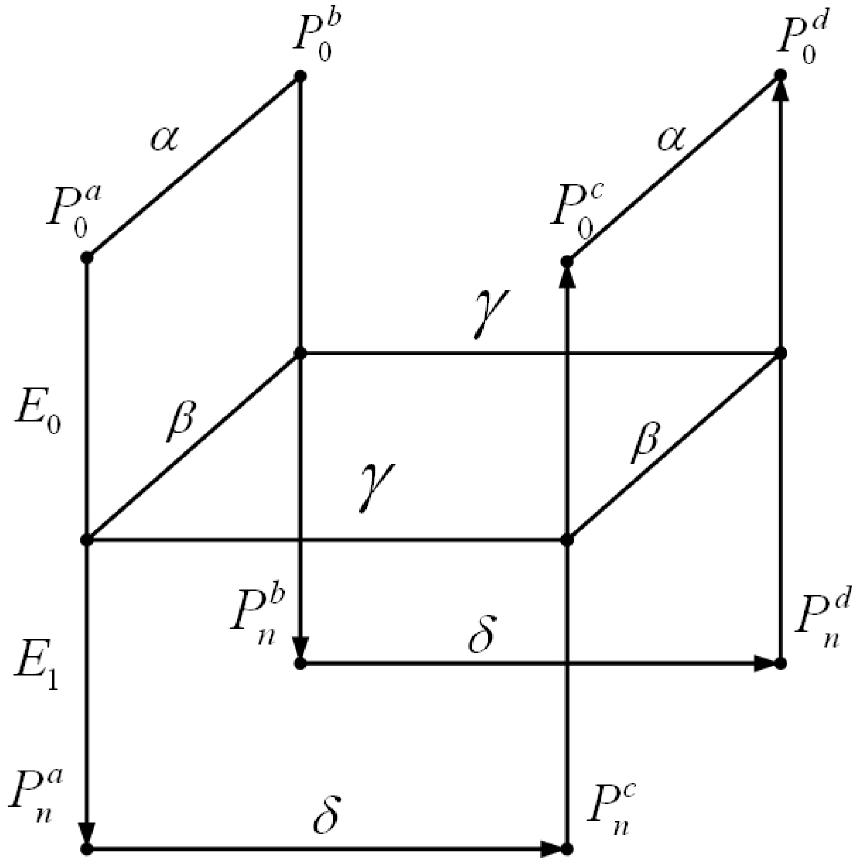


Figure 1: The boomerang attack

afterwards in key recovery step, where one tries to exploit known relations between input and output differences and values of found correct pairs in order to recover some key bits.

The boomerang distinguisher step works as follows (see fig. 1).

- Choose $P_0^a$ and $P_0^b = P_0^a \oplus \alpha$.
- Encrypt the texts: $P_n^a = E(P_0^a, K)$ and $P_n^b = E(P_0^b, K)$.
- Compute the new ciphertexts $P_n^c = P_n^a \oplus \delta$, $P_n^d = P_n^b \oplus \delta$.
- Decrypt the new ciphertexts $P_0^c = E^{-1}(P_n^c, K)$, $P_0^d = E^{-1}(P_n^d, K)$.
- If $P_0^c \oplus P_0^d = \alpha$ store the quartet $(P_0^a, P_0^b, P_0^c, P_0^d)$ in set $\Theta$. We will call these "boomerang quartets".

Here and in the rest of the paper in notation $P_j^i$ the lower index denotes an intermediate output after $j$-th round, 0 denotes plaintext and $n$ denotes ciphertext.

The boomerang key recovery step works as follows. For each found boomerang quartet we suppose that

- $(P_0^a, P_0^b)$ is a correct pair for the differential $\alpha \to \beta$,
- $(P_0^c, P_0^d)$ is a correct pair for the differential $\alpha \to \beta$,
- $(P_n^a, P_n^c)$ is a correct pair for the differential $\delta \to \gamma$,
- $(P_n^b, P_n^d)$ is a correct pair for the differential $\delta \to \gamma$,

and then exploit these assumptions to recover some bits of the master key.

Note that one (or more) pair may not be correct but the quartet still satisfies the condition $P_0^c \oplus P_0^d = \alpha$. We will call this a false boomerang quartet. Note also that the probability of finding a correct quartet is $(pq)^2$ and the probability $P_{false}$ of finding a false quartet can be estimated considering the encryption algorithm as a random substitution. With well-chosen differentials for $E_0$ and $E_1$ one can obtain the probability of finding a correct quartet being much greater than of finding a false one.

In the related-key boomerang attack we suppose that the attacker can encrypt and decrypt each $P_j^i$, $i \in \{a, b, c, d\}$ under the corresponding key $K^i$, where all the keys $K^i$ are unknown to the attacker and related with fixed differences, i.e. $K^i \oplus K^j = \Delta K^{i,j}$ where the differences $\Delta K^{i,j}$ are set by the attacker (or are known).

Formally, in order to mount a boomerang attack under such assumptions we have to consider the bijective transformation $F^* : V_n \times V_k \mapsto V_n \times V_k$, defined as $F^*(X, K) = (E(X, K), K)$. Here and further, saying "related-key differential $\alpha \to \beta$ under related keys $K_1$ and $K_2 = K_1 \oplus \Delta K$" we mean the differential $(\alpha, \Delta K) \to (\beta, \Delta K)$ for $F^*$. Fig. 2 represents the related-key boomerang attack.

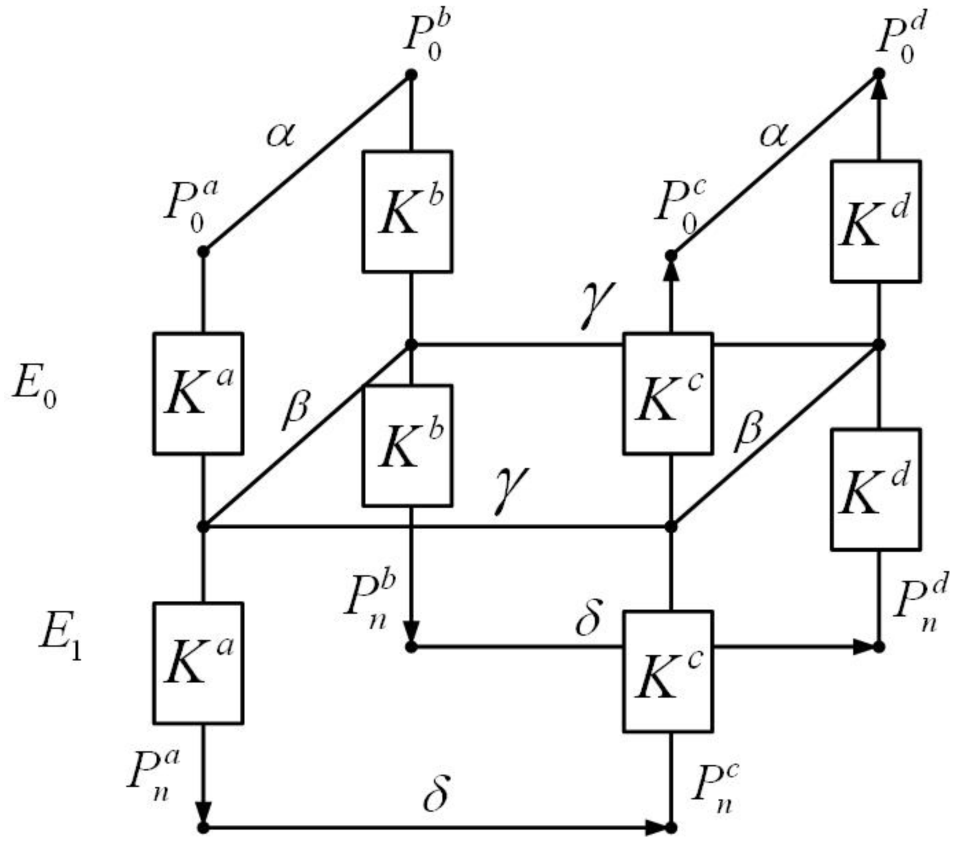


Figure 2: The related-key boomerang attack

# 4 Known results on related-key boomerang attack on the GOST block cipher.

A related-key boomerang attack on the full GOST block cipher was presented in [3]. Here we describe this attack.

GOST block cipher is treated as a composition of two subciphers $E_0$ and $E_1$, $E = E_0 \circ E_1$, where $E_0$ represents the first 24 rounds and $E_1$ — the last 8 rounds.

Consider four related keys $K^i \in V_{256}$, $K^i = (k_1^i, \ldots, k_8^i)$, $k_j^i \in V_{32}$, $i \in \{a, b, c, d\}$, $j \in \overline{1,8}$, such that

$$\Delta K^* = K^a \oplus K^b = K^c \oplus K^d = (e_{31}, 0, e_{31}, 0, e_{31}, 0, e_{31}, 0),$$

$$\Delta K' = K^a \oplus K^c = K^b \oplus K^d = (e_{31}, 0, 0, 0, 0, 0, 0, 0).$$

Here $e_i \in V_{32}$ denotes a word with all bits excepting $i$-th are zeroes.

Consider a related-key differential $\alpha \to \beta = (0, e_{31}) \to (0, e_{31})$ for $E_0$ and a related-key differential $\gamma \to \delta = (0, 0) \to (e_7, 0)$ for $E_1$. Now we can mount the related-key boomerang distinguisher for GOST according to the description given above:

- Choose a plaintext pair $P_0^a$ and $P_0^b = P_0^a \oplus \alpha$.
- Encrypt $P_0^a$ and $P_0^b$ under related keys $K^a$ and $K^b$ respectively and obtain ciphertexts $P_{32}^a = E(P_0^a, K^a)$ and $P_{32}^b = E(P_0^b, K^b)$.
- Compute new ciphertexts $P_{32}^c = P_{32}^a \oplus \delta$, $P_{32}^d = P_{32}^b \oplus \delta$
- Decrypt $P_{32}^c$ and $P_{32}^d$ under respective related keys $K^c$ and $K^d$.
- Obtain plaintexts $P_0^c = E^{-1}(P_{32}^c, K^c)$, $P_0^d = E^{-1}(P_{32}^d, K^d)$.
- Check if $P_0^c \oplus P_0^d = \alpha$. If true, $(P_0^a, P_0^b, P_0^c, P_0^d)$ is said to form a related-key boomerang quartet.

For each related-key boomerang quartet one can make an assumption:

**Assumption 4**

- $(P_0^a, P_0^b)$ *is a correct pair for the related-key differential* $\alpha \to \beta$ *of* $E_0$.
- $(P_0^c, P_0^d)$ *is a correct pair for the related-key differential* $\alpha \to \beta$ *of* $E_0$.
- $(P_{32}^a, P_{32}^c)$ *is a correct pair for the related-key differential* $\delta \to \gamma$ *of* $E_1^{-1}$
- $(P_{32}^b, P_{32}^d)$ *is a correct pair for the related-key differential* $\delta \to \gamma$ *of* $E_1^{-1}$

Just as for the boomerang attack, we can find a false related-key boomerang quartet, which passes the distinguisher filtering condition, but the assumptions above are false.

Note that the paper [3] considers only one set of S-boxes, a so-called "set of Central Bank of Russian Federation", which is equal to the set of S-boxes from the test case for Russian standard for cryptographic hash function (GOST R 34.11-94). For other sets of S-boxes this attack cannot be applied directly (in some cases). Later we will show that the attack may be applied to any set with a modification depending on S-boxes.

Now we present an algorithm from [3].

**Algorithm 1**

1. *Choose* $2^{5.5}$ *plaintext pairs* $P_0^a$ $P_0^b = P_0^a \oplus (0, e_{31})$.
2. *With a chosen plaintext attack scenario, encrypt the plaintexts under related keys and obtain the ciphertexts* $P_{32}^a = E(P_0^a, K^a)$, $P_{32}^b = E(P_0^b, K^b)$.
3. *Compute the new ciphertexts* $P_{32}^c = P_{32}^a \oplus (e_7, 0)$, $P_{32}^d = P_{32}^b \oplus (e_7, 0)$
4. *With a chosen ciphertext attack scenario, decrypt the new ciphertext under respective related keys* $P_0^c = E^{-1}(P_{32}^c, K^c)$, $P_0^d = E^{-1}(P_{32}^d, K^d)$
5. *If* $P_0^c \oplus P_0^d = (0, e_{31})$. *then store* $(P_0^a, P_0^b, P_0^c, P_0^d)$ *in* $\Theta$.
6. *Guess subkey* $k_1^a$ *at the bit positions 12 to 19. Set the corresponding related subkeys* $k_1^c = k_1^a \oplus e_{31}$, $k_1^b = k_1^a$, $k_1^d = k_1^b \oplus e_{31}$. *Initialize a counter for each bit combination with zero.*
   (a) *For each quartet in* $\Theta$ *partially decrypt* $\overline{P}_{31}^a$, $\overline{P}_{31}^b$, $\overline{P}_{31}^c$, $\overline{P}_{31}^d$.
   (b) *Check if* $\overline{P}_{31}^a \oplus \overline{P}_{31}^c = (0, 0)$ *and* $\overline{P}_{31}^b \oplus \overline{P}_{31}^d = (0, 0)$.
   (c) *If true increase the counter for the used key bits by 1.*
7. *Record the round keys* $k_1^a, k_1^b, k_1^c, k_1^d$ *with the largest counter value.*
8. *For a suggested* $k_1^a$ *do the exhaustive search for the remaining* $256 - 8 = 248$. *If the true 256-bit master key is suggested, output the master key. Otherwise restart the exhaustive search with another* $k_1^a$.

## 5 Analysis of the Attack

First of all we have to determine the probabilities $p$ and $q$ of the related key differentials. Now we will show that the probability of the differential $\alpha \to \beta = (0, e_{31}) \to (0, e_{31})$ for $E_0$ is 1.

Indeed, consider the first two rounds of GOST when encrypting a pair $P_0^a = (L_0^a, R_0^a)$ and $P_0^b = (L_0^b, R_0^b)$, $P_0^a \oplus P_0^b = (0, e_{31})$ under related keys:

$$\begin{cases} \Delta L_1 = \Delta R_0 = e_{31} \\ \Delta R_1 = \mathbf{L}\Pi(R_0^a \boxplus k_1^a) \oplus \mathbf{L}\Pi((R_0^a \oplus e_{31}) \boxplus (k_1^a \oplus e_{31})) = 0 \end{cases}$$

$$\begin{cases} \Delta L_2 = \Delta R_1 = 0 \\ \Delta R_2 = \Delta L_1 \oplus \mathbf{L}\Pi(R_1^a \boxplus k_2^a) \oplus \mathbf{L}\Pi(R_1^a \boxplus k_2^a) = e_{31} \end{cases}$$

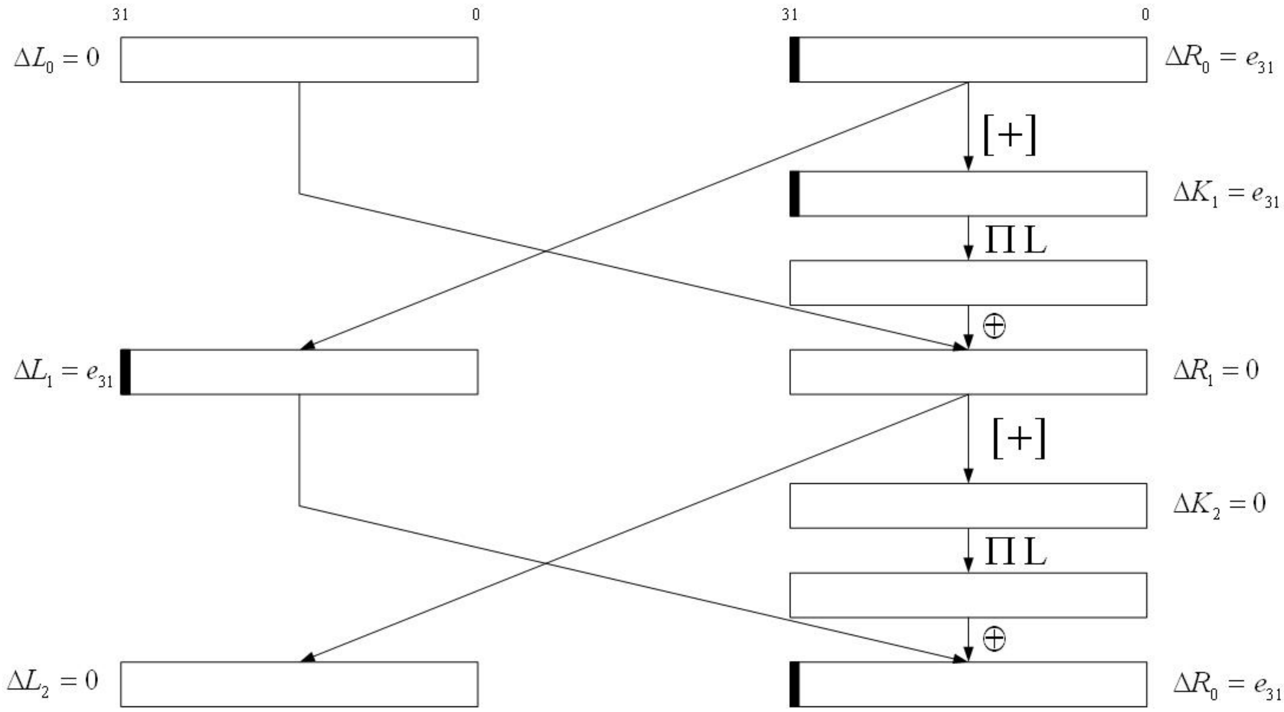


Figure 3: Related-key differential for $E_0$

Fig. 3 explains the related-key differential for $E_0$. Here rectangles represent corresponding data blocks step-by-step. Filled regions mark bit positions with non-zero differences.

Thus, the probability of the related-key differential $(0, e_{31}) \to (0, e_{31})$ for two rounds of GOST is 1. Obviously, the probability of this differential remains the same for 8 rounds and consequently for 24 rounds.

Now we move on to related-key differential $(e_7, 0) \to (0, 0)$ for $E_1^{-1}$. In [3] it is stated that the probability of this differential is $2^{-2}$ which is not true to the fact.

In [3] the mentioned differential is supposed to take place in the first round of $E_1^{-1}$ passing the zero output difference through the remaining 7 rounds. With a correct input pair we have

$$L_{32}^a \oplus \mathbf{L}\Pi(R_{32}^a \boxplus k_1^a) = L_{32}^c \oplus \mathbf{L}\Pi(R_{32}^c \boxplus k_1^c).$$

Taking into account that $L_{32}^c = L_{32}^a \oplus e_7$, $R_{32}^a = R_{32}^c$, $k_1^c = k_1^a \oplus e_{31}$ this is equivalent to

$$\mathbf{L}\Pi(R_{32}^a \boxplus k_1^a) \oplus \mathbf{L}\Pi(R_{32}^a \boxplus (k_1^a \oplus e_{31})) = L_{32}^a \oplus L_{32}^a = e_7,$$

or, just the same

$$\Pi(R_{32}^a \boxplus k_1^a) \oplus \Pi((R_{32}^a \boxplus k_1^a) \oplus e_{31}) = e_{28}.$$

Let $\pi : V_4 \to V_4$ denotes the S-box on high order bits, $\overline{X}$ denotes high order 4-bit subword of $X$, Using this notation the last equation is equivalent to $\pi(\overline{R_{32}^a \boxplus k_1^a}) \oplus \pi((\overline{R_{32}^a \boxplus k_1^a}) \oplus e_3) = e_0$, so, in other words, the differential $e_3 \to e_0$ for $\pi$ is considered. Hence the probability of the related-key differential $(e_7, 0) \to (0, 0)$ for $E_1^{-1}$ equals the probability of the differential $e_3 \to e_0$ for the S-box $\pi$.

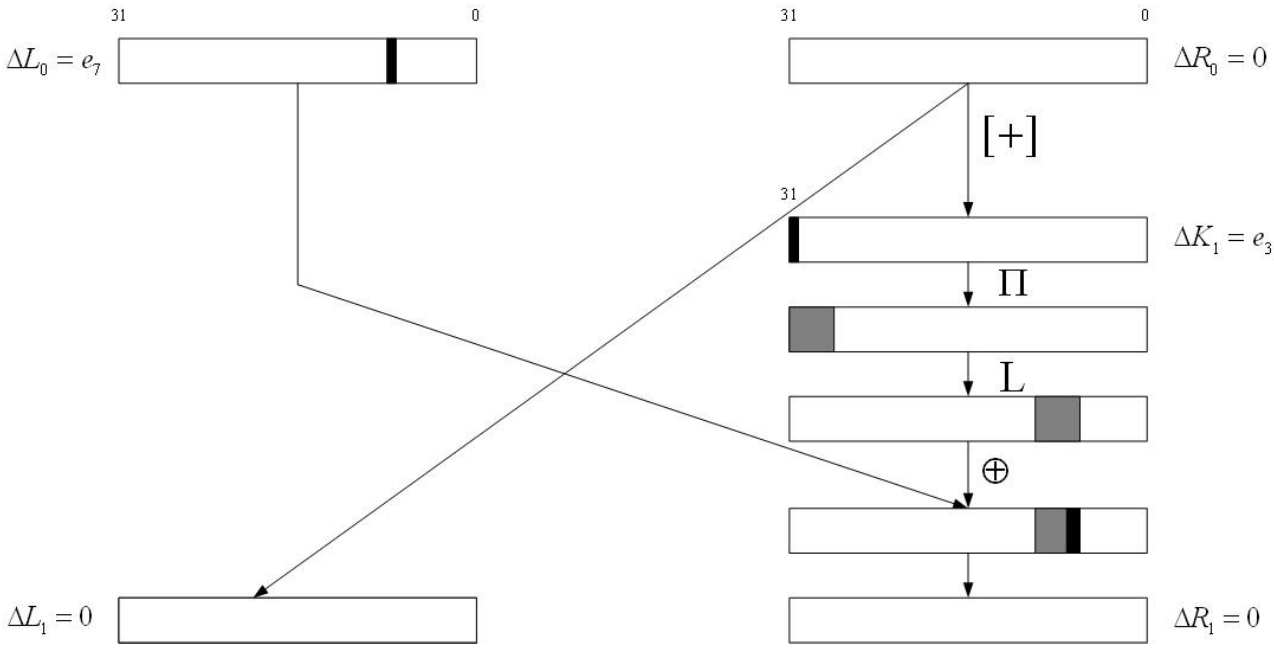


Figure 4: Related-key differential for $E_1^{-1}$

Fig. 4 explains the related-key differential for $E_1^{-1}$. As before, rectangles represent corresponding data blocks step-by-step. Regions filled black mark bit positions with non-zero differences, and regions filled grey mark bit positions with *possibly* non-zero differences.

First we note that, in contradiction to [3], the probability of the differential for the S-box is not $2^{-2}$ but $2^{-3}$.

Secondly, the attack described above depends on S-boxes. In cases when different S-box set is used, the probability may vary. Moreover, it can be equal to zero or one, and in this case the attack is not applicable.

In the former case there aren't any true boomerang quartets. For some S-boxes this problem can be solved. In order to do this, we have to explore the probabilities of differentials of a sort $e_3 \to \omega$ and choose one with non-zero (and not equal to one) probability among them. After that we should consider the related-key differential $(\mathbf{L}[\omega, 0, ..., 0], 0) \to (0, 0)$ for $E_1^{-1}$.

The latter case is the worst one: the input difference $e_3$ always produces the same output difference and hence all key bit combinations will pass the filtering conditions. Therefore the attack will be equivalent to exhaustive key search. However such S-boxes have bad cryptographic properties, making the whole cipher vulnerable to differential and linear attacks and hence such S-boxes are unlikely to be used in practice.

In the rest of the paper we consider the same S-box set as in [3] for simplicity sake. For other S-box sets the attack can be easily adjusted (if possible), using the previous discussion, and will have the same complexity.

So, the probability of finding a boomerang quartet is $(pq)^2 = (1 \cdot 2^{-3})^2 = 2^{-6}$, the probability of finding a false quartet is supposed to be severely less.

Now we move to the analysis of the algorithm.

If the term "partial description" (the step 6(a)) means computing several bits (where possible), then the step 7 does not reject any false bit combination.

Now we compute value of $P_{31}^i$ in bit positions where it is possible ,i.e. where we possess all the necessary information. Obviously, $L_{31}^i = R_{32}^i$. Next, we are able to compute $(k_1^i \boxplus R_{32}^i)[12 \sim 19]$, discarding possibly non-zero incoming carry bit, and consequently compute $R_{31}^i[23 \sim 30]$. It is easy to check that $k_1^i[12 \sim$

$19] = k_1^j[12 \sim 19]$ and $R_{32}^i[12 \sim 19] = R_{32}^j[12 \sim 19]$ for all $i, j \in \{a, b, c, d\}$. Hence, the condition of the step 6(b) holds for all guessed bit combinations. So in step 7 we choose all bit combinations and in step 8 we find the master key by full 256 bit exhaustive search. Fig. 5 explains the discussion above. As

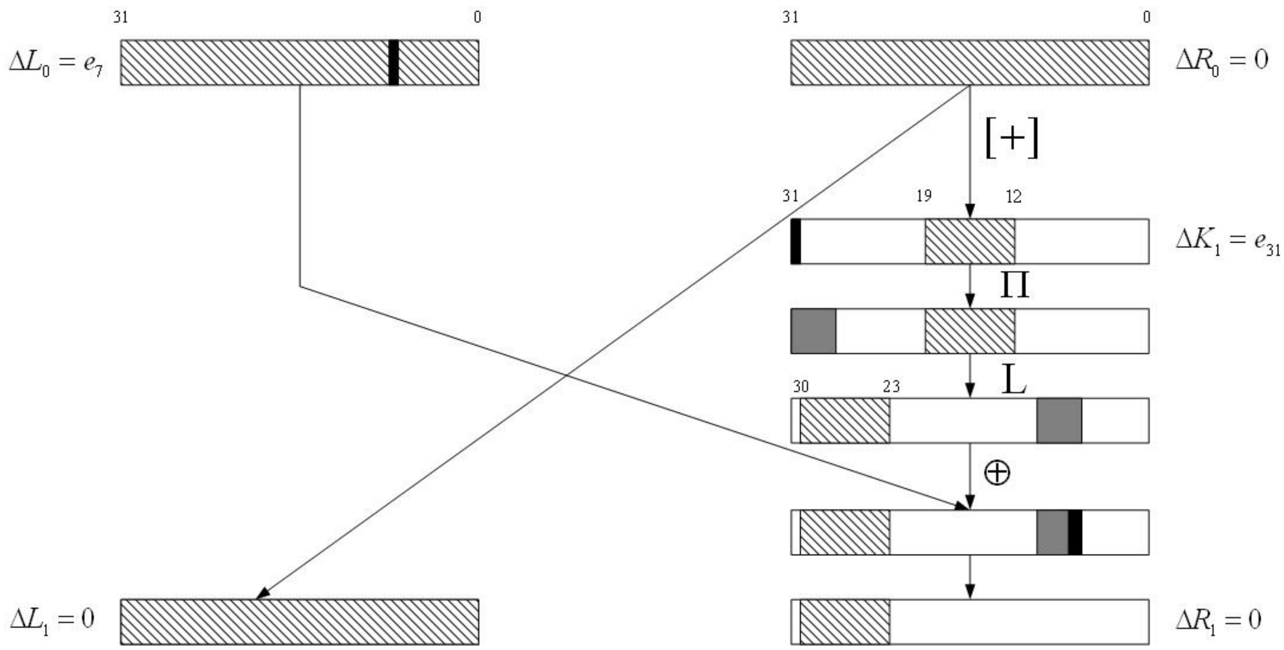


Figure 5: Partial decryption

before, regions filled black mark bit positions with non-zero differences and regions filled grey mark bit positions with *possibly* non-zero differences. Dash patterned regions mark bits which values we are able to compute by partial encryption.

## 6 Subkey recovery related-key boomerang attack on the GOST block cipher

In the previous section we have shown that the "attack" presented by [3] in fact is equivalent to the exhaustive search. Still, with proper modifications the introduced related-key differentials are worth being concerned. In this section we introduce a related-key boomerang attack on GOST, which exploits these related-key differentials and recovers up to 31 bits of a master key.

Let $k_1^i$ denote a true subkey and $\widehat{k_1^i}$ denote a false subkey. Consider false subkey filtering conditions: $P_{31}^a \oplus P_{31}^c \stackrel{?}{=} (0,0)$ and $P_{31}^b \oplus P_{31}^d \stackrel{?}{=} (0,0)$. Obviously, both equations hold for the true subkey (simply by a boomerang quartet construction). A non-zero difference can appear only if two inputs of the high order S-box form a **wrong** pair for the differential $e_3 \to e_0$, i.e.

$$\pi((P_{32}^a \boxplus k_1^a)[28 \sim 31]) \oplus \pi((P_{32}^a \boxplus k_1^a)[28 \sim 31] \oplus e_3) = e_0, \tag{1}$$

but

$$\pi((P_{32}^a \boxplus \widehat{k_1^a})[28 \sim 31]) \oplus \pi((P_{32}^a \boxplus \widehat{k_1^a})[28 \sim 31] \oplus e_3) \neq e_0. \tag{2}$$

In other words, the last condition means that the high order 4-bit subword of the sum of the ciphertext and the true key differs from that of the sum of the ciphertext and the false key. It is possible if 4 high order bits of the true and false keys differ and also when the carry bits upcoming to the 28-th position of the sum differ for the true and false keys. Hence, the partial encryption must be performed on high order bits.

Note, that the keys $k$ and $k \oplus e_{31}$ cannot be distinguished by this condition.

The next algorithm, which is actually a modification of algorithm 1, recovers 3 bits of $k_1^a$.

**Algorithm 2**

1. *Choose $2^{10}$ plaintext pairs $P_0^a$ and $P_0^b = P_0^a \oplus (0, e_{31})$.*
2. *Encrypt under related keys and obtain the ciphertexts $P_{32}^a = E(P_0^a, K^a)$, $P_{32}^b = E(P_0^b, K^b)$.*
3. *Compute the new ciphertexts $P_{32}^c = P_{32}^a \oplus (e_7, 0)$, $P_{32}^d = P_{32}^b \oplus (e_7, 0)$*
4. *Decrypt under related keys $P_0^c = E^{-1}(P_{32}^c, K^c)$, $P_0^d = E^{-1}(P_{32}^d, K^d)$*
5. *If $P_0^c \oplus P_0^d = (0, e_{31})$ store the quartet $(P_0^a, P_0^b, P_0^c, P_0^d)$ in $\Theta$.*
6. *Guess subkey $k_1^a$ at positions 28 to 31. For each guessed bit combination initialize a counter with zero. Compute related subkeys $k_1^c = k_1^a \oplus e_{31}$, $k_1^b = k_1^a$, $k_1^d = k_1^b \oplus e_{31}$ at the same positions*

(a) *For each quartet in* $\Theta$ *compute* $\overline{P}_{31}^{a}$, $\overline{P}_{31}^{b}$, $\overline{P}_{31}^{c}$, $\overline{P}_{31}^{d}$ *at corresponding bit positions*

(b) *Check if* $\overline{P}_{31}^{a} \oplus \overline{P}_{31}^{c} \stackrel{?}{=} (0,0)$ *and* $\overline{P}_{31}^{b} \oplus \overline{P}_{31}^{d} \stackrel{?}{=} (0,0)$.

(c) *If true, increase the counter by 1.*

7. *Record bit combinations* $k_1^a$ *with the highest counter.*

**Remark 5** *For each recorded* $k_1^a$ *recover the remaining* $256 - 4 = 252$ *bits by the exhaustive search or using any other attack. If the right key is found then stop. Otherwise choose another* $k_1^a$ *and repeat the exhaustive search or attack.*

**Remark 6** *The algorithm discards all but two bit combinations, since the subkeys* $k$ *and* $k \oplus e_{31}$ *are indistinguishable by conditions (1) and (2).*

**Remark 7** *The number of plaintext pairs to be chosen* ($2^{10}$) *is dictated by the tendency to provide the high rate of success. See section 8 for details.*

The attack can be improved to recover 31 bits of $k_1^a$. Indeed, suppose that for some $m \leq 27$ we have $k_1^i[m+1 \sim 31] = \hat{k}_1^i[m+1 \sim 31]$ and $k_1^i[m] \neq \hat{k}_1^i[m]$. If there exists such a related-key boomerang quartet that the plaintext $P_{32}^a = (L_{32}^a, R_{32,m}^a)$ with an arbitrary $L_{32}^a$ and

$$R_{32,m}^a[j] = \begin{cases} \{0,1\}_R, & j = 28, 31; \\ k_1^a[j] \oplus 1, & j = \overline{m+1, 27}; \\ 1, & j = m; \\ 0, & j = \overline{0, m-1}; \end{cases}$$

then 4 high order bits of the sum of $k_1^a$ and $\hat{k}_1^a$ with this plaintext are different. Here $\{0,1\}_R$ denotes a random element of $\{0,1\}$, so there are $2^4$ possible different values of $R_{32,m}^a$.

It follows from the previous statement that we have to, informally speaking, “throw a boomerang” backwards, starting not with chosen plaintexts, but with chosen ciphertexts (see fig. 6).

The next algorithm recovers 31 key bits of GOST.

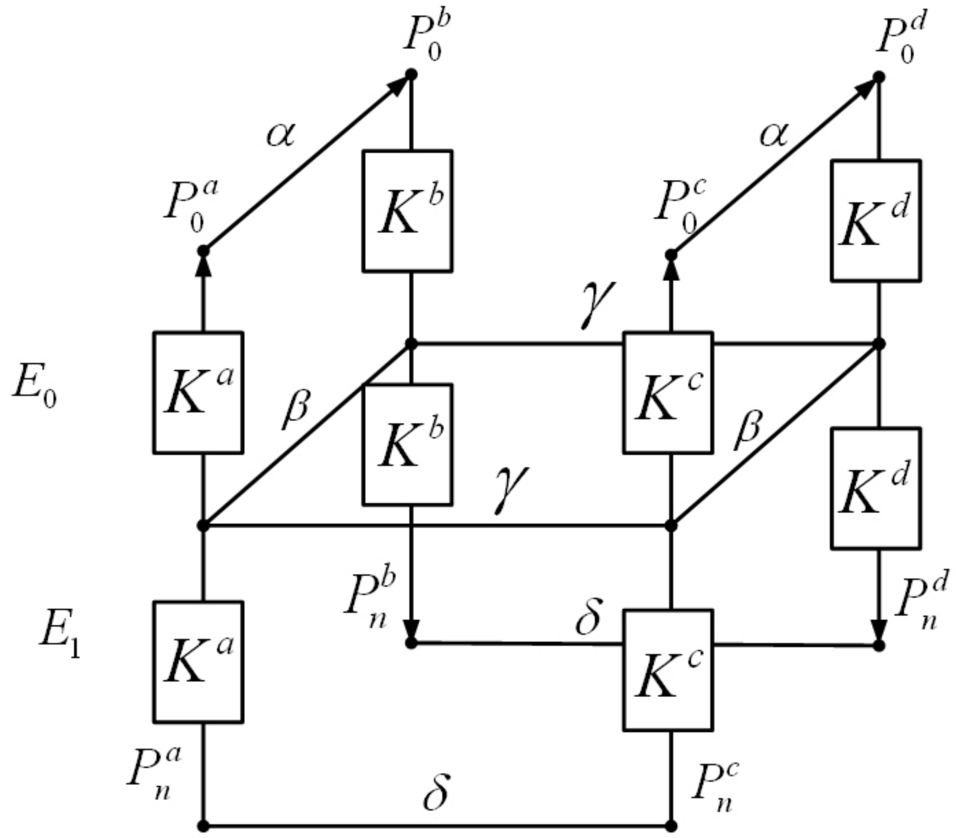


Figure 6: The inverse related-key boomerang attack.

**Algorithm 3**

1. *Recover* $k_1^a[28 \sim 30]$*, using algorithm 2. Set* $k_1^a[31] = 0$.
2. *Initialize a counter* $\xi$ *with 27.*
3. *Choose* $2^{10}$ *ciphertext pairs* $P_{32}^a$ $P_{32}^c = P_{32}^a \oplus (e_7, 0)$*, with all* $2^4$ *possible values of* $R_{32}^a$ *of a kind* $R_{32,\xi}^a$*, defined above, and* $L_{32}^a$ *takes* $2^6$ *different arbitrary values.*
4. *Decrypt under related keys to obtain* $P_0^a = E^{-1}(P_{32}^a, K^a)$, $P_0^c = E^{-1}(P_{32}^c, K^c)$.
5. *Compute the new plaintexts* $P_0^b = P_0^a \oplus (0, e_{31})$, $P_0^d = P_0^c \oplus (0, e_{31})$.
6. *Encrypt under related keys to obtain* $P_{32}^b = E(P_0^b, K^b)$, $P_{32}^d = E(P_0^d, K^d)$.
7. *Check if* $P_{32}^b \oplus P_{32}^d \stackrel{?}{=} (e_7, 0)$*. If true, store the quartet* $(P_{32}^a, P_{32}^b, P_{32}^c, P_{32}^d)$ *in* $\Theta$.

8. *Consider two subkeys* $k_1^{a,(0)}, k_1^{a,(1)}$

$$k_1^{a,(l)}[j] = \begin{cases} k_1^a[j], & j = \overline{\xi + 1, 31}; \\ l, & j = \xi; \\ 0, & j = \overline{0, \xi - 1}; \end{cases} .$$

9. *Compute the related keys* $k_1^{c,(l)} = k_1^{a,(l)} \oplus e_{31}$.

10. *For each subkey* $k_1^{a,(l)}$ *initialize a counter with zero and*

    (a) *For each quartet in* $\Theta$ *decrypt* $P_{31}^a, P_{31}^b$.

    (b) *If* $P_{31}^a \oplus P_{31}^c \stackrel{?}{=} (0, 0)$ *increase the counter by 1.*

11. *Choose one of the* $k_1^{a,(0)}, k_1^{a,(1)}$ *with the largest counter value.*

12. *Set* $k_1^a[\xi] = k_1^{a,(l)}[\xi]$, *for the chosen* $l$.

13. *If* $\xi = 0$, *then the subkey is recovered. Otherwise decrease counter* $\xi$ *by 1 and go to step 3.*

**Remark 8** *With found* $k_1^a$ *recover the remaining* $256 - 32 = 224$ *bits by the exhaustive search or any other attack. If the right key is recovered output the master key. Otherwise set* $\overline{k_1^a} = k_1^a \oplus e_{31}$ *and recover the remaining bits.*

**Remark 9** *As before, the number of plaintext pairs to be chosen* ($2^{10}$) *is dictated by the tendency to provide the high rate of success. See section 8 for details.*

# 7 Generalization of the attack

Now we present a generalized attack. Once the subkey $k_1^a$ is recovered, GOST is reduced to 30 rounds, since we are able to perform transformations of the first and the last round using the recovered subkey. In order to recover subsequent subkeys we will use different quartet of related keys, namely, to recover $k_{t+1}^a$ assuming the subkeys $k_1^a, ..., k_t^a$ are known, we use the following quartet:

$$\Delta K^* = K^a \oplus K^b = K^c \oplus K^d = (e_{31}, 0, e_{31}, 0, e_{31}, 0, e_{31}, 0),$$

$$\Delta K' = K^a \oplus K^c = K^b \oplus K^d = (0, ..., \underset{t+1}{e_{31}}, 0, ..., 0) \tag{3}$$

(In (3) all the 32-bit subwords of $\Delta K'$ are zero except $(t+1)$-th that is equal to $e_{31}$).

Using these subkeys we set up attacks similar to the algorithms 2 and 3:

**Algorithm 4**

1. *Choose* $2^{10}$ *plaintext pairs* $P_0^a$ *and* $P_0^b = P_0^a \oplus (0, e_{31})$.
2. *Encrypt under related keys and obtain the ciphertexts* $P_{32}^a = E(P_0^a, K^a)$, $P_{32}^b = E(P_0^b, K^b)$.
3. *With known* $k_i^j,\ j \in \{a, b, c, d\},\ i \in \overline{1, t}$ *compute* $P_{32-t}^a, P_{32-t}^b$
4. *Compute the new intermediate texts* $P_{32-t}^c = P_{32-t}^a \oplus (e_7, 0)$, $P_{32-t}^d = P_{32-t}^b \oplus (e_7, 0)$
5. *With known* $k_i^j,\ j \in \{a, b, c, d\},\ i \in \overline{1, t}$ *compute* $P_{32}^c, P_{32}^d$
6. *Decrypt under related keys:* $P_0^c = E^{-1}(P_{32}^c, K^c)$, $P_0^d = E^{-1}(P_{32}^d, K^d)$
7. *If* $P_0^c \oplus P_0^d = (0, e_{31})$ *store the quartet* $(P_0^a, P_0^b, P_0^c, P_0^d)$ *in* $\Theta$.
8. *Guess subkey* $k_{t+1}^a$ *at positions 28 to 31. For each guessed bit combination initialize a counter with zero. Compute related subkeys* $k_{t+1}^c = k_{t+1}^a \oplus e_{31}$, $k_{t+1}^b = k_{t+1}^a$, $k_{t+1}^d = k_{t+1}^b \oplus e_{31}$ *at the same positions.*
   - (a) *For each quartet in* $\Theta$ *using* $k_i^j,\ j \in \{a, b, c, d\},\ i \in \overline{1, t}$ *compute* $P_{32-t}^a, P_{32-t}^b$
   - (b) *Compute* $\overline{P}_{32-t-1}^a, \overline{P}_{32-t-1}^b, \overline{P}_{32-t-1}^c, \overline{P}_{32-t-1}^d$ *at corresponding positions*
   - (c) *Check if* $\overline{P}_{32-t-1}^a \oplus \overline{P}_{32-t-1}^c \stackrel{?}{=} (0, 0)$ *and* $\overline{P}_{32-t-1}^b \oplus \overline{P}_{32-t-1}^d \stackrel{?}{=} (0, 0)$.
   - (d) *If true, increase the counter by 1.*
9. *Record bit combinations* $k_{t+1}^a$ *with the largest counter value.*

Define $R^a_{32-t,m}$ as

$$R^a_{32-t,m}[j] = \begin{cases} \{0,1\}_R, & j = 28, 31; \\ k^a_{t+1}[j] \oplus 1, & j = \overline{m+1, 27}; \\ 1, & j = m; \\ 0, & j = \overline{0, m-1}; \end{cases}$$

**Algorithm 5**

1. *Recover* $k^a_{t+1}[28 \sim 31]$ *using algorithm 4. Set* $k^a_{t+1}[31] = 0$.
2. *Initialize a counter* $\xi$ *with 27.*
3. *Choose* $2^{10}$ *intermediate text pairs* $P^a_{32-t}$ $P^c_{32-t} = P^a_{32-t} \oplus (e_7, 0)$, *with all* $2^4$ *possible values of* $R^a_{32-t}$ *of a kind* $R^a_{32-t,\xi}$, *and* $L^a_{32-t}$ *takes* $2^6$ *different arbitrary values.*
4. *Using* $k^j_i$, $j \in \{a, b, c, d\}$, $i \in \overline{1, t}$ *compute* $P^a_{32}, P^c_{32}$
5. *Decrypt under related keys to obtain* $P^a_0 = E^{-1}(P^a_{32}, K^a)$, $P^c_0 = E^{-1}(P^c_{32}, K^c)$.
6. *Compute the new plaintexts* $P^b_0 = P^a_0 \oplus (0, e_{31})$, $P^d_0 = P^c_0 \oplus (0, e_{31})$.
7. *Encrypt under related keys to obtain* $P^b_{32} = E(P^b_0, K^b)$, $P^d_{32} = E(P^d_0, K^d)$.
8. *Using* $k^j_i$, $j \in \{a, b, c, d\}$, $i \in \overline{1, t}$ *compute* $P^b_{32-t}, P^d_{32-t}$
9. *Check if* $P^b_{32-t} \oplus P^d_{32-t} \stackrel{?}{=} (e_7, 0)$. *If true, store* $(P^a_{32-t}, P^b_{32-t}, P^c_{32-t}, P^d_{32-t})$ *in* $\Theta$.
10. *Consider two subkeys* $k^{a,(0)}_{t+1}, k^{a,(1)}_{t+1}$:

$$k^{a,(l)}_{t+1}[j] = \begin{cases} k^a_{t+1}[j], & j = \overline{\xi+1, 31}; \\ l, & j = \xi; \\ 0, & j = \overline{0, \xi-1}; \end{cases}.$$

11. *Compute the related subkeys* $k^{c,(l)}_{t+1} = k^{a,(l)}_{t+1} \oplus e_{31}$.

12. *For each key* $k_{t+1}^{a,(l)}$ *initialize a counter with zero and*
    (a) *For each quartet in* $\Theta$ *using* $k_i^j$, $j \in \{a,b,c,d\}$, $i \in \overline{1,t}$ *compute* $P_{32-t}^a, P_{32-t}^b$
    (b) *Compute* $P_{32-t-1}^a$, $P_{32-t-1}^b$
    (c) *Check if* $P_{32-t-1}^a \oplus P_{32-t-1}^c \stackrel{?}{=} (0,0)$
    (d) *If true, increase the counter by 1.*
13. *Choose one of the* $k_{t+1}^{a,(0)}, k_{t+1}^{a,(1)}$ *with the highest counter.*
14. *Set* $k_{t+1}^a[i] = k_{t+1}^{a,(l)}[i]$, *for the chosen* $l$.
15. *If* $\xi = 0$, *then output the subkey. Otherwise decrease counter* $\xi$ *by 1 and go to step 3.*

There are two important observations regarding the generalized attacks. The first one deals with the false quartets. Algorithms 2-5 find the correct keys if, roughly speaking, the true quartets outnumber the false ones. Computer simulations show that true quartets appear with probability of about $2^{-6}$, as predicted. At the same time, the number of false quartets increases as the number of rounds reduces. Experiments show that differential trails for all found false quartets follow the boomerang structure, with the same trails for $E_0$ ($\alpha \to \beta = (0, e_{31}) \to (0, e_{31})$), but different trails for $E_1^{-1}$.

As discussed above, the differential for $E_1^{-1}$ used in true quartets splits into 1R characteristic $(e_7, 0) \to (0,0)$ with probability $2^{-3}$ and $(0,0) \to (0,0)$ characteristic the for remaining rounds with probability 1 and therefore the overall probability does not depend on the number of rounds in $E_1^{-1}$.

Differentials for $E_1^{-1}$ used in false quartets can as well be split into 1R characteristic $(e_7, 0) \to \xi$ and $\xi \to \chi$ for remaining rounds with some *nonzero* $\xi$ and $\chi$. Obviously, the diffusion of a nonzero difference $\xi$ and hence the probability of finding a false boomerang quartet strongly depends on the number of rounds in $E_1^{-1}$. Experiments show that if $E_1^{-1}$ is reduced to 3 (or more) rounds, then algorithms 4 and 5 recover the subkey properly. Unfurtunately, further reduction dramatically increases the number of false quartets which in turn makes the algorithms as “effective” as simple guessing.

The second observation is that algorithms 3 and 5 recover two possible subkey values: $k^a_{t+1}$ and $k^a_{t+1}\oplus e_{31}$, assuming that $k^a_i,\ i \in \overline{1,t}$ are chosen correctly.

Hence, we can build a binary subkey tree of height 6 and width $2^6$. Each node at layer $t$ corresponds to $k^a_t$ or $k^a_t\oplus e_{31}$, where the value of $k^a_t$ is computed by algorithm 3 or 5, assuming the keys $k^a_i,\ i \in \overline{1,t-1}$, corresponding to the nodes in path from this node to the root, are determined correctly. Each path from a leaf to the root of the constructed tree corresponds to a set of six subkeys, one of which is the true set. Total number of such sets is $2^6$. For each of the sets one can bruteforce the two remaining subkeys or use other cryptanalytic attacks to recover them.

## 8 Complexity and success rate of the presented attack

Here we explain our choice of data quantity and estimate time complexity and success rate of the algorithms.

Claim the success rate of our attack being near 1, namely $\tau = 1 - 10^{-4}$.

We say that the algorithm 2,3,4 or 5 executed *successfully*, if it output only two possible bit combinations for algorithms 2 and 4 and only two possible subkeys for algorithms 3 and 5.

Let $n$ denote the number of chosen plain- or ciphertext pairs given on input. We estimate the success rate of the algorithms 2 and 4 as

$$P_1(n) = \big(1 - (1 - (pq)^2)^n\big) \cdot (1 - 2^{-n}) = \big(1 - (1 - 2^{-6})^n\big) \cdot (1 - 2^{-n}).$$

The success rate of the algorithms 3 and 5 is estimated as follows:

$$P_2(n) = \big(1 - (1 - (pq)^2)^n\big)^{28} = \big(1 - (1 - 2^{-6})^n\big)^{28}.$$

The success rate of recovering one subkey is $P_1(n)P_2(n)$, and success rate for six subkeys is $(P_1(n)P_2(n))^6$. One can check that given $n = N = 2^{10}$ implies

$$(P_1(N)P_2(N))^6 > \tau.$$

That is why we chose data quantity of $N = 2^{10}$ pairs of text for algorithms 2, 3, 4, 5.

Time complexity of algorithm 2 is $C_1 = 2^{10}\cdot(4+\frac{4}{32})$ encryptions/decryptions. The complexity of algorithm 3 is $C_2 = 28 \cdot 2^{10} \cdot (4 + \frac{2}{32}) + C_1$ encryptions/decryptions. Time complexity of algorithms 4 and 5 is $C_3(t) = 2^{10} \cdot 2^t \cdot (4 + \frac{4}{32} + t\frac{4}{32})$ and $C_4(t) = 28 \cdot 2^{10} \cdot 2^t \cdot (4 + \frac{2}{32} + t\frac{4}{32}) + C_3(t)$ respectively. Note that $C_1 = C_3(0)$ and $C_2 = C_4(0)$.

Thus, we recover 192 bits of the master key with the presented related-key boomerang attack with time complexity

$$\sum_{t=0}^{5} 2^t \cdot C_4(t) \approx 2^{28}$$

encryptions with success rate over $\tau = 1 - 10^{-4}$. The attack uses 14 related keys with fixed pairwise differences set by an attacker.

Using bruteforce attack for remaining subkeys results in total time complexity of

$$2^{28} + 2^6 \cdot 2^{32} \cdot 2^{32} \approx 2^{71}$$

encryptions.

**Remark 10** *One can easy see, that probability of any differential for any bijective 4-bit wide S-box is either equal to zero or is grater or equal to* $2^{-3}$. *Thus, for an arbitrary S-box set, time and data complexity of the modified attack is the same, while the success rate can be equal or higher.*

The standard GOST 28147-89 does not specify algorithms for master key generation and usage, implying a “natural” assumption on keys being randomly and independently chosen. In given assumption, the overall success rate of the attack is extremely low.

We suppose that the attacker can encrypt and decrypt some texts under four arbitrary and independently chosen secret keys. The probability that these four keys form a quartet suitable for applying the partial key recovery related-key boomerang attack (algorithms 2 and 3) is

$$P = 1 \cdot \frac{1}{2^{256}} \cdot \frac{1}{2^{256}} \cdot \frac{1}{2^{256}} = 2^{-256 \cdot 3} = 2^{-768}.$$

At the same time, one can simply "guess" a key with probability $2^{-256} \gg 2^{-768}$. The probability that 14 arbitrary and independently chosen keys form a specific set suitable for applying the general attack is

$$P = 2^{-256 \cdot 13} = 2^{-3328}.$$

One can study different attack scenarios, but in general, given the assumption of zero-knowledge about the keys for the attacker, the practical application of the related-key boomerang attack is doubtful, since it is based on a extremely rare event, often, as in previous example, much more infrequent than successful key-guessing.

# 9 Conclusion

In this paper we present the full key recovery attack on GOST block cipher that uses related-key boomerang technique. Previously, such kind of attack was presented in [3]. It considered a fixed set of S-boxes and recovered 8 bits of the master key. We show that, due to an error, the attack from [3] actually reduces to the exhaustive search.

Then we develop related-key boomerang attack that works for most sets of S-boxes and uses related-key differential depending on the S-boxes. Using the backward boomerang technique with appropriate choice of ciphertexts, our attack recovers 31 bits of the master key. Next we present a generalized attack that recovers 192 bits of the master key with practical data and time complexity.

Note that within the related-key model, practical attacks on variety of block ciphers, including AES [1] and Kasumi [2] were developed. At the same time, the most problematic aspect of these attacks is the reliance on related keys, which is not universally accepted as a practical attack model. Certainly, the possibility of such attacks has to be examined in each specific implementation and mode of operation of a cipher, and new block ciphers have to be designed to resist such attacks. However, any related-key attack on a block cipher does not represent a fundamental obstacle to practical usage of the cipher.

## References

[1] Biryukov A., Khovratovich D. Related-key Cryptanalysis of the Full AES-192 and AES-256. — Cryptology ePrint Archive, Report 2009/317, http://eprint.iacr.org/2009/317.pdf. — 2009.

[2] Dunkelman O., Keller N., Shamir A. A Practial-Time Attack on the A5/3 Cryptosystem Used in Third Generation GSM Telephony. — Cryptology ePrint Archive, Report 2010/013, http://eprint.iacr.org/2010/013.pdf. — 2010.

[3] Fleischmann Ewan, Gorski Michael, Huehne Jan-Hendrik, Lucks Stefan. Key recovery attack on full GOST block cipher with zero time and memory. — Published as ISO/IEC JTC 1/SC 27 N8229. — 2009.

[4] Wagner David. The boomerang attack// Knudsen Lars R., editor, FSE/ Lecture Notes in Computer Science. — v. 1636, Springer. — 1999. — 156–170.