An Improved Differential Attack on Full GOST

- extended version, 17 December 2015[*] -

Nicolas T. Courtois

University College London, Gower Street, London, UK

Abstract. GOST 28147-89 is a well-known block cipher and the official encryption standard of the Russian Federation. A 256-bit block cipher considered as an alternative for AES-256 and triple DES, having an amazingly low implementation cost and it is becoming increasingly popular [51, 36]. Until 2010 researchers unanimously agreed that: "despite considerable cryptanalytic efforts spent in the past 20 years, GOST is still not broken", see [51] and in 2010 it was submitted to ISO 18033 to become a worldwide industrial encryption standard.

In 2011 it was suddenly discovered that GOST can be broken and it is insecure on more than one account. There is a substantial variety of recent innovative attacks on GOST [9, 26, 37, 10–12, 31, 21, 19, 20]. We have reflection attacks [37, 26], attacks with double, triple and even quadruple reflections [26, 20], a large variety of self-similarity and black-box reduction attacks [9, 26, 19, 20], some of which do not use any reflections whatsoever [26, 9] and few other. The final key recovery step in various attacks is in many cases a software algebraic attack [26, 9, 20] or/and a Meet-In-The-Middle attack [37, 26, 31, 21]. In differential attacks key bits are guessed and confirmed by the differential properties [54, 10–14, 49] and there have already been quite a few papers about advanced differential attacks on GOST [54, 10–13, 49, 14, 22, 15]. There is also several even more advanced "combination" attacks which combine the complexity reduction approach based on high-level self-similarity of [26, 19, 9] with various advanced differential properties with 2,3 or 4 points, see [26, 20].

In this paper we consider some recent differential attacks on GOST [54, 10–14, 49, 15] and show how to further improve them. We present a single-key attack against full 32-round 256-bit GOST with time complexity of 2^{179} which is substantially faster than any previous single key attack on GOST.

[*] **Note:** This paper is a new extended and updated version of a paper with the same title [27] to appear in LNCS 9100, Springer in March 2016. This paper is substantially expanded compared to the earlier version from 2012. The main attack remains the same and also the same as in the Springer version [27]. The introduction, preliminary study and exploration of underlying cipher structure and properties attacks are however greatly expanded in order to show a bigger picture and provide useful insights. We explain in details the philosophy and the structure of our attacks, and the methodology of how one can find such attacks, cf. Part I, "Discovery of Advanced Differential Attacks on GOST", pages 9- 24. We also cite recent related and follow-up work.

Key Words: Block ciphers, GOST, differential cryptanalysis, sets of differentials, truncated differentials, guess-then-determine, Gaussian distribution, distinguisher attacks.

1 Introduction

This paper is about advanced differential attacks on GOST block cipher [54, 10–12, 22, 53, 49, 52, 13, 14, 24]. The first half of this paper is consecrated to the question of how such attacks can be discovered, cf. Part I. An interesting question is then how reliable some heuristic results are [54, 10–14, 49, 24] which question is not obvious given the fact that GOST is not a Markov cipher [40, 23, 49] and that truncated differentials can be decomposed into some interesting sub-classes, cf. [22, 49] and Section 4.

Remark. This paper does NOT cover the whole spectrum of what differential properties can bring in terms interesting or/and efficient attacks on GOST. In this paper we do not consider multiple key attacks [17, 19, 26, 20, 39] and we do not try to develop or at more advanced "combination" attacks which combine the complexity reduction approach based on high-level self-similarity of [26, 19, 9] with advanced differential properties with 2,3 and 4 points, [26, 20]. This paper is essentially a technical paper on simple yet highly-optimized truncated differential attacks in a single key scenario.

1.1 Background: Differential Cryptanalysis

Differential cryptanalysis (DC) is based on tracking of changes in the differences between two messages as they pass through the consecutive rounds of encryption. It is one of the oldest classical attacks on modern block ciphers, if not the oldest. We refer to [13] for a short historical survey. In cryptographic literature it was first described and analysed by Biham and Shamir [4, 5, 45] and applied to DES algorithm in early 1990s, and has been studied ever since [45, 7]. However, as reported by Coppersmith it was already known by the IBM team which have designed DES around 1974 [30, 28, 6, 29], under the name of T-attack or Tickle attack. Reportedly, Don Coppersmith has explained that defending against DC had been a design goal when DES was designed in 1970s and after discussions with NSA it was decided this technique should be kept confidential, in the context of the "competitive advantage" the United States enjoyed (compared to other countries) in the field of cryptography, cf. [28, 13].

Differential attacks are excessively well known, even for non-specialists. They are always mentioned as one of the most obvious ways to evaluate the security of ciphers. Differential cryptanalysis have effectively guided more or less all cipher designs with very few exceptions for many decades. It may therefore appear very surprising that researchers could have for decades ignored the existence of differential attacks on GOST. On the contrary GOST was quite frequently claimed very secure against such attacks, even in 2012 cf. [53]. It appears quite clearly that the research community have sometimes heavily underestimated the power of DC, which is reflected by many statements which appear in the literature for decades and which are simply far from being true, this including some very recent papers [1] from 2012. In his textbook written in the late 1990s Schneier writes that: "Against differential and linear cryptanalysis, GOST is probably stronger than DES", see [55]. In fact it clearly isn't: even a quick examination will show that one round of GOST is substantially weaker than in DES, and that the diffusion in GOST is very poor and very few key bits are used in each round. GOST is very much unlike in DES where diffusion is excessively good and most of key bits are used in each round. However everybody expected that these definite signs of weakness compared to DES could be compensated by a larger number of 32 rounds cf. [33, 55,

3]. Later in 2000 Russian researchers claimed that breaking GOST with five or more rounds is "very hard" and explain that as few as 7 rounds out of 32 are sufficient to protect GOST against differential cryptanalysis [33]. Needless to say later research have not confirmed at all such very optimistic claims [40, 41, 44, 23, 24] However GOST appears to be quite secure in the standard historical Biham-Shamir formulation of DC with single differences on the full state [4, 33, 54].

Interestingly however Knudsen and other researchers have soon after Biham and Shamir proposed more powerful advanced differential attacks. We have "truncated differential" attacks by Knudsen [46] and other similar attacks (for example attacks with a more general concept of sets of differentials [10]). Such attacks are applied to GOST as early as in 2000 by Seki and Kaneko [54] with some success. In 2011 Courtois and Misztal have found new differential sets for GOST [10] which are substantially better than previously known. They also report that even the original attack of Seki and Kaneko [54] works actually substantially better than predicted, see [11, 12].

More advanced differential attacks are still nevertheless very much underestimated in the cryptographic community. A lot of research about Boolean functions and optimal resistance against DC have been done, in a totally unrealistic settings where small components of ciphers are studied in isolation regardless of how they are connected inside a cipher. This research leads to highly misleading results about types of strong or "optimal" S-boxes which [47, 50–52, 13] are expected to protect ciphers against DC. In contrast in this paper we are going to make a rather very surprising claim that the security of a cipher such as GOST against DC does not depend that much on the S-boxes, but rather on the connections of the cipher, see Section 4 and [53, 13, 14, 49, 22, 15, 23, 24].

We have observed also that many researchers have been influenced by what they have learned from the analysis of a particular cipher such as DES which tends to be quite misleading when we study other ciphers. For example in the most recent paper, again for 2012 and specifically about advanced DC, and which specifically looks at ciphers with small block which is the case for GOST, in Section 1.1. page 3 of [1] we read: *Truncated differentials, [...] in some cases allow to push differential attacks one or two rounds further.* This is yet another example how poorly we have understood differential cryptanalysis. Our recent research on GOST [10–14, 49, 15, 22–24] shows that we can gain not two but much closer to 20 rounds (!) compared to what we initially expected with differential attacks with single differentials [33, 34, 54]. In this paper we continue this line of work. Our goal is to construct interesting statistical distinguishers on say 20 rounds of GOST and transform them into complex attacks on the full 32-round GOST cipher.

1.2 Road Map

This paper is organized as follows:

1. In the Abstract and Introduction section 1, we explained the history of differential cryptanalysis and its advanced variants, and how it relates to other known attacks on GOST and block ciphers. Then in Section 2 we explain the outside-symmetric character of the GOST key schedule and the general principle of splitting GOST into three sections of for example 6+20+6 rounds.
2. Then we have Part I which is consecrated to the difficult combinatorial exploration questions of discovery of "good" differential properties for GOST. These questions are also studied in [54, 10–13, 49, 22].
3. Then in Part II we study the question of constructing a good distinguisher for a larger number of 20 rounds of GOST which is one of the key questions in this paper.
   Additional questions heuristics and results related to parts I and II are also covered in [49, 54, 10–13, 49, 14, 22, 15].
4. In part III we show ho to transform a distinguisher attack on 20+ rounds into an optimized key recovery attack for the full 32 rounds. Earlier and more basic attempts to develop such an attack can be found in [11, 12, 49]. A highly optimized competitive attack is necessarily more complex and requires more stages. We construct our main attack[1] in three stages:
   a. In Section 6 we explain a concept of a sequence of concentric distinguishers and construct one very precise example.
   b. In Section 7 we study the propagation inside GOST in order to be able to construct well chosen subsets of key bits to be guessed at various stages of our attack.
   c. Then finally in Section 8 we describe a full advanced differential attack on 32 rounds of GOST. It is described as a sequence of consecutive steps, in which well-chosen assumptions on the key and data (plaintext) bits are progressively refined with early rejection. It can also be seen as depth-first tree search with multiple levels, such that when no valid solution (full correct 256-bit key) is found in one sub-branch, we backtrack and explore another sub-branch.

[1] The core of this attack also appears in [27] which paper misses however a lot of additional explanations and insights which can only be found here.

2 How To Construct A Differential Attack on GOST?

This paper is about advanced differential attacks on GOST block cipher. In essence this paper describes essentially one such attack on GOST and all the necessary facts and methodology. At the high level the attack exploits the poor key schedule in GOST. At the low level the attack exploits insufficient diffusion and advanced differential properties.

2.1 Key Schedule in GOST

The key structural property of GOST which makes it suitable for cryptanalytic attacks of the specific kind and specific form, is that the last 8 rounds are identical to the fist 8 rounds run in the opposite direction (however this symmetry does not follow for more inner rounds).

rounds	1 8	9 16
keys	$\mathbf{k_0} k_1 k_2 k_3 k_4 k_5 k_6 k_7$	$\mathbf{k_0} k_1 k_2 k_3 k_4 k_5 k_6 k_7$
rounds	17 24	25 32
keys	$\mathbf{k_0} k_1 k_2 k_3 k_4 k_5 k_6 k_7$	$k_7 k_6 k_5 k_4 k_3 k_2 k_1 \mathbf{k_0}$

Fig. 1. Key schedule in GOST is symmetric and the first/last few rounds use the same subset of keys, guessing these keys allows to "peel-off" many rounds from both sides at a low price.

This property has a big impact on security of GOST and was already exploited in numerous attacks on this cipher [9, 26, 37, 10–12, 31, 21, 19, 20]. Only 32 bits of the whole 256-bit key are used in one round and 32 out of 256 is a fairly small proportion equal to 1/8. Moreover for every 32 bits guessed we can remove two full outer rounds, instead of 1 round for a similar cipher without a weak key schedule. Similar the keys used int he first X rounds are the same as used in the last X rounds for every $X = 0 \ldots 8$. We call this property "**outer symmetry**". This property makes GOST vulnerable to differential cryptanalysis, for example earlier research showed that if we guess 192 key bits in just 6 rounds, we can remove 12 full rounds of GOST we can already get an attack faster than brute force, see [11].

In this paper we will exploit this precise and strong symmetry in GOST key schedule further and in more detail. We will look at differential properties which involve sets of differentials which are totally symmetric for the first 8 rounds and the last 8 rounds. Our key guesses will be far more precise than guessing keys for full rounds and specially adapted to this highly symmetric situation. This in order to maximize the number of cases which can be safely discarded due to these partial key guesses.

2.2 Preliminary Remarks

In our later attack which will be described in Sections 6-8 we are going to split GOST into three pieces with 6+20+6 rounds. Early advanced differential attacks were based on a statistical distinguisher which exploited a number of differential propagations to distinguish up to 20 rounds of GOST from a random permutation [11, 12] and needed to guess complete 32-bit keys for several outer rounds in order to fully reconstruct these

internal differentials. The question of constructing further distinguishers on say e.g. 20 rounds of GOST have subsequently been studied in [22, 23, 49]. Their applications have been studied in [27, 22, 23, 49] and in this paper.

Our approach is substantially more detailed. We will guess only some well-chosen key bits which will be used to filter P/C (Plaintext,Ciphertext) pairs used later in the attack. As in [11, 12, 49] the attack runs through many stages with great many filtering/guessing steps, where at each step we reduce the number of cases to consider (the plaintext space, some key bits already guessed and pre-computed relations between all these) and only after this reduction of number of cases we make additional guesses. In contrast to the attack presented in [12] our attack will use symmetric differential sets, which is dictated by the symmetry in the key schedule.

Large parts of this whole process can be viewed as an adaptive Depth-First-Search (DFS) attack on a tree of possibilities which is constructed adaptively depending on the assumptions currently considered as valid. This type of process is very widely used in cryptanalysis.

There is a substantial difficulty in differential attacks where the key size is much larger than the block size as in GOST: there are false positives, differentials which do not propagate but occur naturally, by accident. The key point is that for a very long time the false positives are not eliminated in a differential attack on GOST. We are just dealing with assumptions on internal difference bits in GOST, their consequences and relations between these assumptions but for many many steps none of the steps of the attack is able to see if the inner 20 rounds are 20 rounds of GOST, more rounds of GOST, or maybe just some other permutation. This can only be seen at a much later stage of the attack.

Before we get there we need to study a number of preliminary technical questions.

Part I

Discovery of Advanced Differential Attacks on GOST

3 The Structure of GOST

We will only study the [by far] most popular set of GOST S-boxes known as the "GostR3411_94_TestParamSet" in [36] which was published as early as in 1994. This set of S-boxes is according to Schneier [55] the one which would be used by the Central Bank of the Russian Federation. This is exactly what most researchers call just "the GOST cipher" (without any additional mention) in the cryptographic literature, and many papers ignore the existence of numerous additional variants of the GOST cipher.

Important Remark: This choice of S-boxes greatly affects all the differential probabilities we use in this paper. It is possible to see that similar attacks exist for other variants of GOST It is very extremely naive to believe that they don't exist, cf. [53, 52]. On the contrary. Recent research shows that differential attacks on GOST work quite well across all (known or alternative] sets of S-boxes see [13, 14, 22, 49].

3.1 Notation

In this paper we consider the traditional "Twisted" representation of Feistel ciphers. Each round of GOST looks exactly the same:

$$(L,R) \mapsto (R, L \oplus f_k(R))$$

This is consistent with a majority of works about DES, GOST and other Feistel schemes.

3.2 Internal Connections in GOST

We number the inputs of the S-box Si for $i = 1,2,\ldots,8$ by integers from $4i+1$ to $4i+4$ out of 1..32. The outputs of one S-box Si are numbered according to their final positions after the rotation by 11 positions: for example the inputs of S6 are $20,21,22,23$ and the outputs are $32,1,2,3$.

GOST has 32 rounds such as the one described in Fig. 2 below.

On our picture below the $\boxplus$ denotes the addition modulo 2^{32}. On this picture we do NOT represent the final circular shift by 11 positions modulo 32 which occurs in GOST after the S-boxes. It is represented in a different way, by numbering the output bits of the S-boxes, to see directly where they are connected.

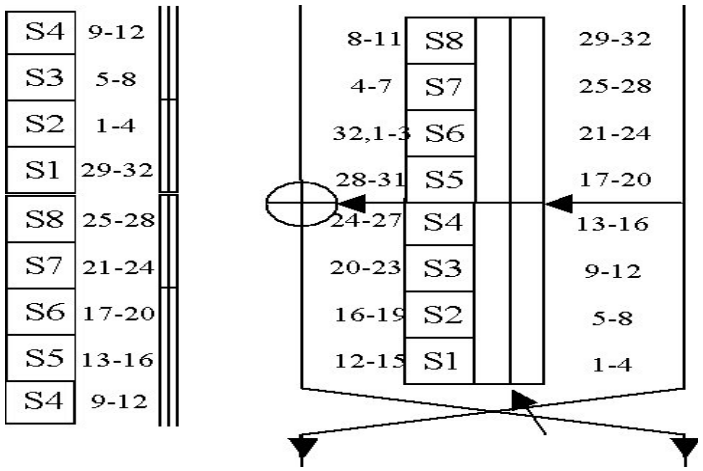

Fig. 2. One Round of GOST And Connections in The Following Round

Given the key scheduling in Fig. 1 we have a complete description of GOST.

At the left margin in Fig. 2 we also show S-box numbers in the next round, which is very helpful, to see which bits are successfully determined in our attacks on GOST. A more detailed explanation of how these bits in the next round depend on the bits in the previous round will be developed later in Fig. 14.

3.3 Sets of Differentials, Aggregated and Truncated Differentials

GOST seems[2] to be secure in the standard historical Biham-Shamir formulation of differential cryptanalysis (DC) with **single differences on the full state** [4, 33]. However, Knudsen and other researchers have soon proposed more powerful advanced differential attacks, which are called **"truncated differential" attacks** by Knudsen [46]. Similar attacks were applied to GOST as early as in 2000 by Seki and Kaneko [54]. In this paper [54] authors propose a more general formulation: **"sets of differentials"**, and it is clear that "truncated differential" attacks [46] are a special case of attacks with "sets of differentials" cf. [54, 10]. It remains the question of which exact properties are the strongest and most relevant in practical cryptanalytic attacks. This is not a small question and the whole Part I of this paper is consecrated to the study of this paper. It is possible to see that properties presented by Seki and Kaneko in 2000 are not the strongest [54] and until 2011 these questions have not received sufficient attention. Since 2011 many very substantially stronger differential properties have been discovered, cf. [10, 11, 22, 49, 24, 13] and this paper.

Interestingly best known attacks of this type are currently much closer to less general "truncated differential" attacks [46] than to more general "sets of differentials" attacks of [54], which formulation is maybe too general and very hard to study due to combinatorial explosion of the number of possible differentials. Basically "truncated differential" attacks [46] seem to capture well the best advanced differential attacks on GOST, they work extremely well with additional [guess then determine] steps of more complex attacks and little [if any] benefit is achieved by considering more general formulations. One slight generalization is the notion of **affine sets** studied in Section 4.5. Another important intermediate notion is the notion of **"aggregated differentials"** as defined in [10, 11] and in Section 3.4 of this paper. All these notions are useful and we cannot know in advance which attacks on GOST are optimal and how to capture best attacks on GOST without excessive generality which would make such attacks harder to discover and harder to study.

Remark: An alternative way of looking at truncated differential attacks is to emphasize **collisions at the set of "inactive" bits**. See Section 2.3 Collision tests in [32]. See [24] for a detailed explanation how these attacks are related.

[2] At first sight, which does not mean it is secure, see Section 4.11.

3.4 Aggregated and Truncated Differentials in GOST

We consider differences with respect to the popular bitwise XOR operation. Following previous work on this topic [11, 12] we define *an aggregated differential* A, B as the transition where any non-zero difference $a \in A$ will produce an arbitrary non-zero difference $b \in B$ with a certain probability.

In particular, we consider the case when A is a set of all possible non-zero differentials contained within a certain mask. This also is a special case of "Truncated Differentials" [46] which are defined as fixing the difference not on all but a subset of data bits. For example we can assume that active bits are $\{9,10,11,20,21,22,31\} \subset \{0..31\}$. In addition we need to be careful and explicitly exclude all-zero differentials from this set. For example let

$$\Delta = 0x80700700$$

we obtain a set of all differences on 32 bits with between 1 and 7 active bits (but not 0) and where the active bits are contained within the mask 0x80700700. This is exactly the same as saying that active bits are $\{9,10,11,20,21,22,31\}$ and we disallow all-zero differences. This is equivalent to having a set of $2^7 - 1$ differences on 32 bits whcih are allowed, all the other differences are not allowed.

Similarly, the set denoted by $[\Delta, \Delta]$ is a set of difference on 64 bits with up to 14 active bits, where any non-zero difference is allowed, including also differences where the difference is zero in one half, but not the all-zero difference on both halves. We have $|A| = 2^{14} - 1$: there are exactly $2^{14} - 1$ single 64-bit differences in this set of differentials A.

For example the following fact was established in [10–12]:

Fact 3.4.1 *The truncated differential* $[\Delta,\Delta]$ *with uniform sampling of all differences it allows, produces an element of the same aggregated differential set* $[\Delta,\Delta]$ *after 4 rounds of GOST with probability about* $2^{-13.6}$ *on average over all possible keys, where* $\Delta = 0x80700700$ *has 7 active bits.*

For 6 rounds the probability is $2^{-18.7}$ *on average over all possible keys.*

For 8 rounds the probability is $2^{-25.0}$ *on average over all possible keys.*

Remark: Recent research shows that the size of 14 bits is close to optimum, i.e. sets with a different size are less likely to be as good, see Fig. 9 page 23 in Section 4.13 and [24].

Now we look at a one particular differential which we have noticed, arrives with a particularly large probability:

Fact 3.4.2 *The set* $[\Delta,\Delta] = [80700700, 80700700]$ *produces a differential of the form* $[00000700, 80780000]$ *with probability of* $2^{-22.19}$ *for 7 rounds of GOST.*

This was obtained by a computer simulation[3]. We have $|A| = 2^{14} - 1$ and $|B| = 2^8 - 1$. This *an aggregated differential* A, B contains $(2^{14} - 1)(2^8 - 1)$ single differential characteristics.

[3] Assuming that events are governed by a Poisson process, it is possible to know at which moment it is reasonable to stop the simulation see [49, 56, 16]. In this paper we typically report these events with precision of 1 digit, except for probabilities smaller than 2^{30} where we cannot be certain about precision.

4 Propagation of Differentials in GOST

It is possible to see that among truncated differentials of type [80700700,80700700] like for example in Fact 3.4.1 there exist several interesting non-trivial sub-classes. This is related to the notion of *general open sets* which we study in [22, 49].

4.1 General Open Sets

It is a quite peculiar definition based on a certain heuristic idea on how to replace the study of all possible subsets of $2^{64}-1$ or just $2^{14}-1$ possible differences on 64 bits by a smaller number of distinct disjoint sets of differentials in which certain differentials are considered to be similar and are always studied together.

Then we call *general open sets* sets or just *open sets*, special sets of differentials on 64 bits, in which we have removed all non-trivial sub-classes. For example from the set of differentials [80700700, 80700700] we remove all the differentials which represent a greatly simplified pattern with less active bits, for example differentials which are members of [80000000, 00000700] and many other similar sub-sets.

In this notion different bits are grouped into 32 groups of 1 or 3 bits. Each byte in hex is split into 1 higher bit and 3 lower bits corresponding to hex mask either 8 or 7. The key idea is that these sets of 1 or 3 bits behave in a "similar way" in differential attacks due to the GOST connections which we can contemplate on Fig. 3 below. Accordingly, we can define the open set 80700700,80700700 where by convention we will not put square brackets [] anymore, as follows. We will only keep differentials which have essentially simultaneously all major 6 parts of the string of 80700700, 80700700 which remain "active" as opposed to inactive. For example we keep the single differential $0x$80500300,$0x$80200400 in which all the 6 zones of 1 or 3 bits each remain active. We refer to [22, 49] for a formal definition and additional examples.

Overall it is possible to see that we can partition the set of $2^{14}-1$ differences denoted by [80700700, 80700700] which uses 6 groups of 1 or 3 bits each, into 2^6-1 distinct open sets which are **disjoint** sets of differentials of variable sizes. In Fig 6 below we denote these sets by 6-character abbreviations for example 00700000, 80700000 would be abbreviated as _7_87_).

4.2 Transitions Between General Open Sets

At this level of detail, it is already possible to see that not all transitions are possible, for example in one round no differential in one set is possible at all given differential from one specific set at the input. This leads to a certain particular graph of possible transitions which can be used to find efficient differential attacks on GOST, cf. Fig 6 page 18. This behavior is again dictated by the connections inside GOST cf. Fig. 3.

Moreover GOST is a highly predictable cipher as far as propagation of differentials is concerned. We have observed that even after many rounds, for example 8 rounds, only some classes on this graph can be obtained, many are obtained with negligible probability, as shown in Fig. 8 at page 21. Similarly the entropy of the output difference grows slowly and can be quite low after many rounds, see Section 4.6 page 19.

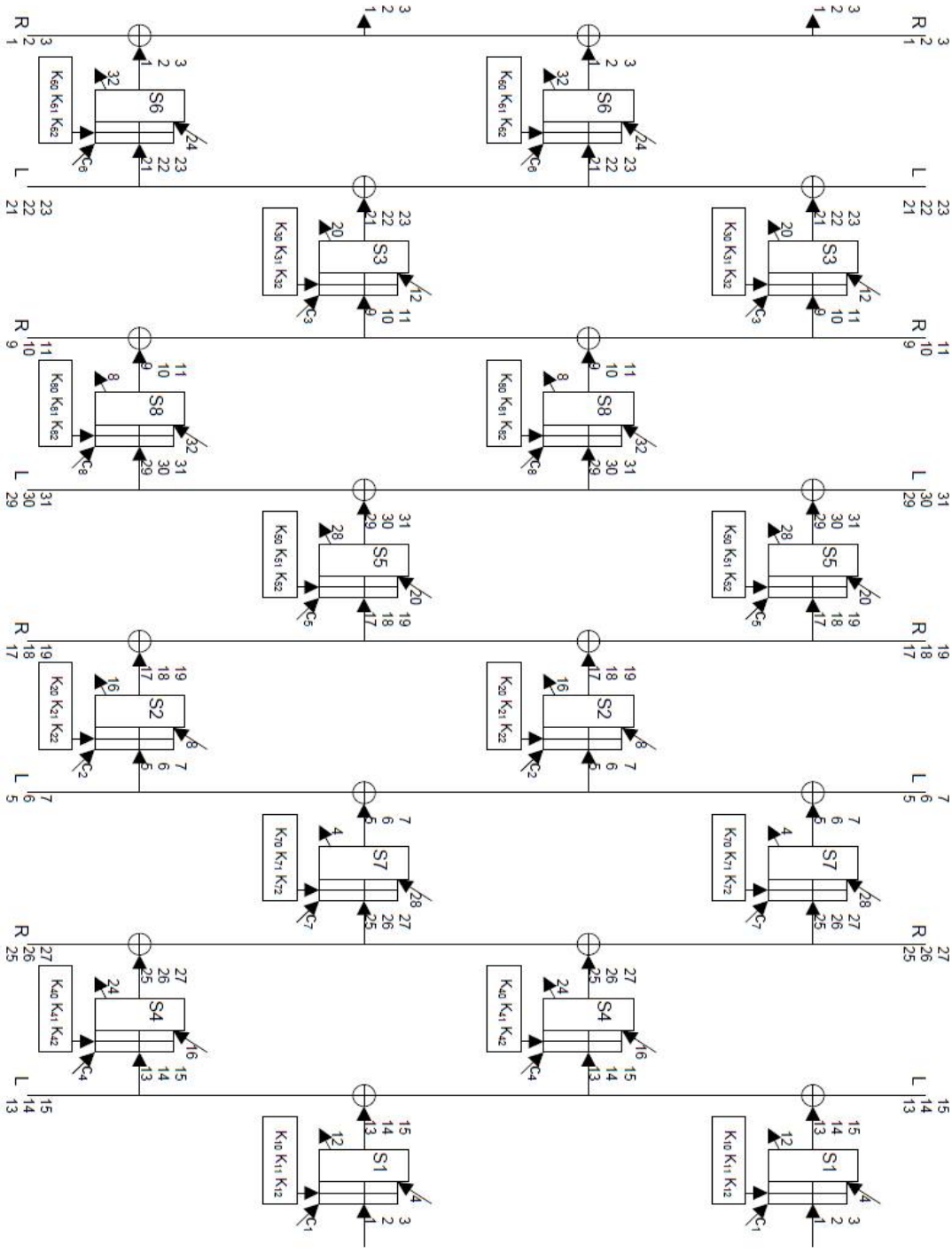

Fig. 3. Connections between half of S-boxes in 4 consecutive rounds

4.3 On Important Classes of Aggregated Differentials in GOST

It is possible to see that many very good differential attacks on GOST are related to the important structure which exists within GOST. Basically, half of GOST S-boxes are very closely connected, and loosely connected with the other half, which is shown in Fig. 3. It is like two distinct nearly-bijective block ciphers applied in parallel with relatively few connections or leakage between the two internal states.

Remark. Moreover these 2 sub-ciphers use distinct keys for large numbers of rounds up to 24.

In this respect some very good attacks on GOST known to us can be classified in several major categories which we call 8+0, 5+1, and 3+3.

Differential Attacks of Type 3+3: The meaning of 3+3 is that the attack uses ONLY 3 fixed S-boxes from Fig. 3, for example S3,S6,S8, and another fixed 3 S-boxes from the complement of Fig. 3, and no other S-boxes. Moreover all these S-boxes are connected in a cyclical structure such that non-zero differences circulate within the structure, and do not leave it. The structure must be closed and circular because the GOST S-boxes are bijective and non-zero differentials cannot be canceled totally. This is shown in Fig. 4. The key interesting differentials which we exploit in this paper are of the type 3+3.

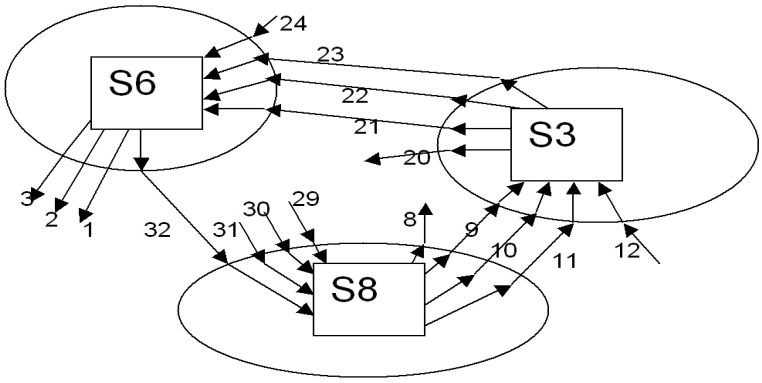

Fig. 4. Connections between S-boxes used in our 3+3 attack

Differential Attacks of Type 1+5: Other non-trivial interesting classes of differentials exist in GOST. We give one example of a differential set of type 1+5. It is as follows:
$[07070008, 08007070]$. This set of $2^{14} - 1$ single differentials propagates for 8 rounds with an average probability of $2^{-26.0}$ for 8 rounds of GOST. Here active S-boxes are, just one S-box S7 on one Fig. 3 and on the other copy of Fig. 3, and in order of connections we have S-boxes S14725 where the perturbation can circulate in a closed loop. with relatively high probabilities.

Differential Attacks of Type 4+4: They can also be seen as an overlapping of two 3+3 attacks, see Fig. 5

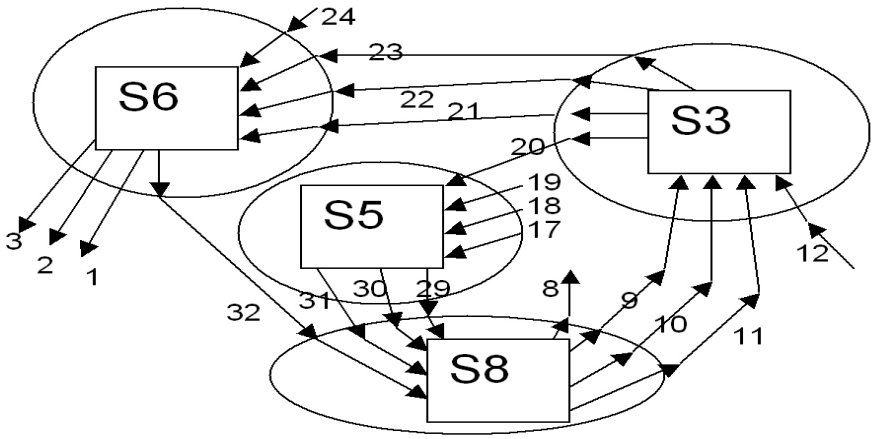

Fig. 5. Extended Version: of Fig. 4 with 4 S-boxes

Differential Attacks of Type 8+0: These are the original attacks of Seki and Kaneko from [54]. These attacks are the oldest and not the best ones, but clearly already depend on our partitioning of all GOST S-boxes into two sets.

Remark: All these properties are very strong and implications of such properties go far beyond the present paper. These are basically high profile structural properties which necessarily lead to many interesting [not yet published] attacks on reduced round GOST. However it is not clear if such attacks can compete with the main result of this paper precisely for 32 rounds and given the current GOST key schedule.

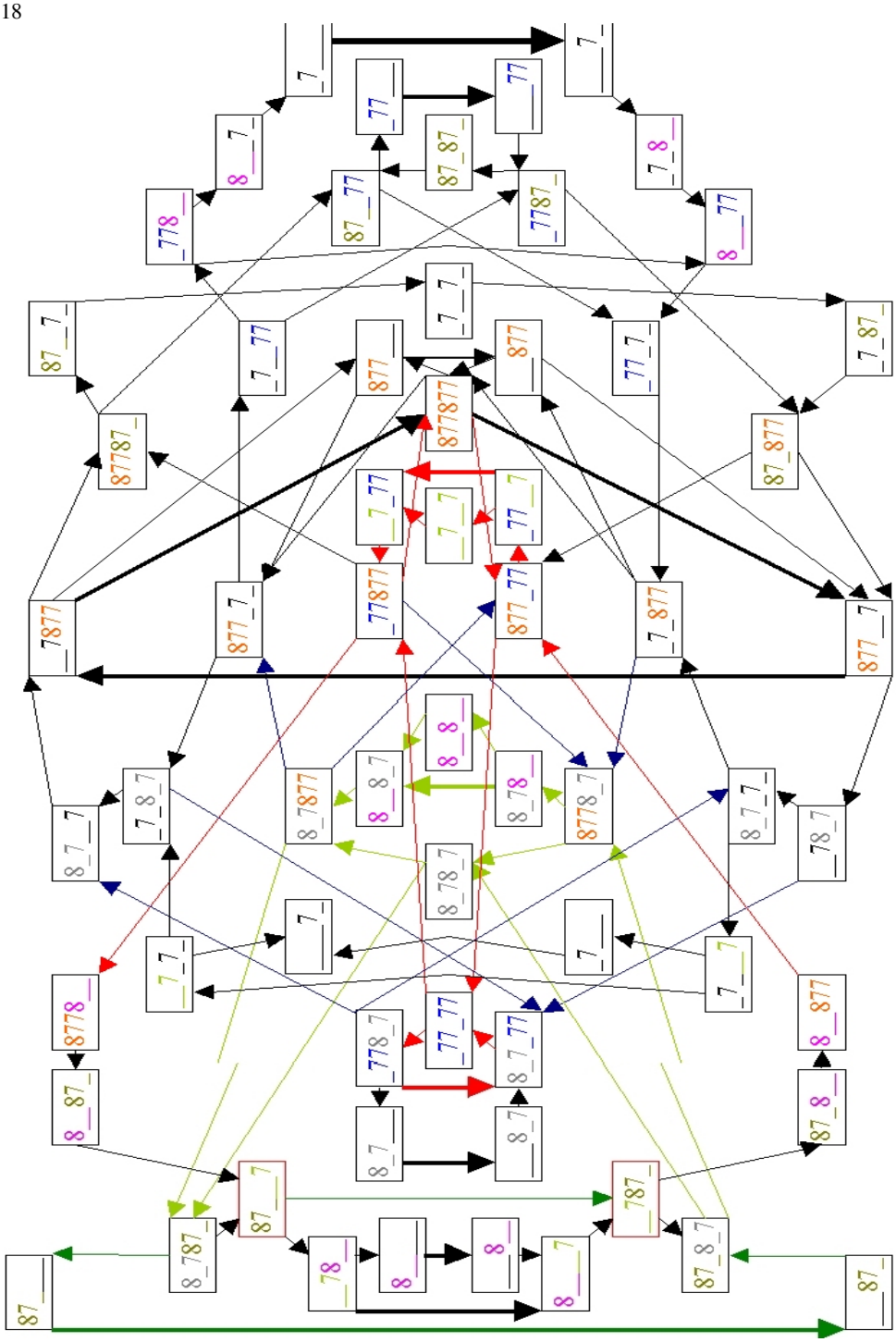

Fig. 6. Transitions between so called “general open” sets of differentials in GOST

4.4 How Much Attacks Depend on S-boxes

The key point we want to make here is that these classes of differentials do NOT depend that much on the S-boxes. They depend much more on the structure of the cipher and the exact connections inside. This is especially true for sets which include many differentials, such as studied in this paper where individual very strong differentials will matter less.

A common misconception is that the security of a cipher such as GOST against differential cryptanalysis (DC) depends extensively on the S-boxes, and some "optimal" S-boxes would make it more secure, see [47, 51]. Researchers in the academia study S-boxes in isolation [47, 50] and propose new ciphers [47, 51] or try to tweak/improve the security of known ciphers [47, 41, 52]. However when S-boxes are connected together, everything changes(!). We are **not certain if it is possible at all** to make a cipher such as GOST secure against DC by changing only the S-boxes [47, 51, 52], which idea was discussed during the ISO standardization process of GOST. The question of how far one can go with advanced differential attacks such as studied in this paper remains widely open, see [53]. We conjecture that the security of GOST against advanced forms of DC depends "essentially" on the connections on the cipher, and though some S-boxes make it weaker against DC, none will make it really very strong.

Remark. This conjecture is disputed in [53]. It is totally incorrect to claim that the attacks do not work for the new replacement S-boxes they propose [53, 52], on the contrary. We refer to [13] and also [14, 49, 22, 24, 15, 23, 49] and to the following sections for a more detailed discussion of this topic. Numerous examples of attacks with different sets of S-boxes can be found below starting from Section 4.12.

4.5 Affine Sets

We take another look at our example of alternative differential set of type 1+5 which is [07070008,08007070]. It had $2^{14}-1$ single differentials propagates for 8 rounds with an average probability of $2^{-26.0}$ for 8 rounds of GOST.

Interestingly, this set can be seen as a linear space of dimension 14 without the zero point. Quite remarkably, if we shift this set by a constant, so that it becomes an affine set for example with the exact shift of 0x00000002,0x00000000 (where we need to exclude another point which is still 0 after the shift by our affine constant) we obtain an affine iterative set of differentials which also contains $2^{14}-1$ single differentials and yet it propagates for 8 rounds with an average probability of $2^{-24.9}$ for 8 rounds of GOST instead of $2^{-26.0}$, and which is strictly better even than in our best "traditional" 3+3 set from Fact 3.4.1. However we have not been able to find such properties for other numbers of rounds.

4.6 Propagation, Order From Chaos

The propagation of differentials can be seen as a system in which the initial low entropy differential pattern slowly dissipates, cf. also [23, 22]. However it does NOT have to be so if we consider the output entropy relative to specific output differentials. Our experiments show that the output entropy will almost always be substantially lower then expected. For example the exact distribution of the $2^{14}-1$ output single differentials in Fact 3.4.1 is going to be far from uniform which reduces the entropy. We have even

observed something quite remarkable which is like order emerging from chaos and is not obvious to explain. We give here one example.

As in Fact 3.4.1 we consider the input differential (Δ,Δ) and the output differential (Δ,Δ) after 6 rounds. On average over possible $2^{14}-1$ input differentials it propagates with probability $2^{-18.6}$. This probability however hides the fact that among the $2^{14}-1$ input differentials there are few for which the propagation is much easier, because some S-boxes are going to be inactive, for example differentials of form $[80000700,00700000]$. Therefore it is interesting to look at the "hardest" case, which is the majority of the $2^{14}-1$ cases, where we specifically forbid the difference under any of 8's and 7's to be non-zero at then input. In this case the propagation for 6 round occurs with probability $2^{-23.8}$ instead of $2^{-18.6}$. We call this sort of set an "open set".

Now the amazing property we have observed is that out of these $2^{-23.8}$ cases with differences on 14 bits, an unexpectedly large proportion of cases, about 40%, or $2^{-25.0}$, has only 3 active bits out of 14, and falls within $[00000000,00000700]$ after 6 rounds. We have done everything to obtain input differences as chaotic possible just because of the sheer complexity of 6 rounds of encryption. Yet the output we have seen they are **either** highly complex and random (when outside of $[80700700,0x80700700]$), or nearly constant (for 40% of the remaining time).

Another example which illustrates on how entropy of differentials in GOST increases slowly within a certain framework can be found in Fig 4.9 below.

4.7 Another Example

We consider the following transition: from $[80700000,00000700]$ at the input to $[80700700,80700700]$ after 4 rounds. This happens with probability $2^{-8.40}$. Quite interestingly, inside these $2^{-8.40}$ cases the output set of type $[80000000,00000700]$ happens about half of the time and this set is smaller than our input set of differentials.

It is possible that these properties or low entropy (or emerging order) are due to "rigidity" of our advanced differentials sets: only some patterns are possible.

4.8 Interpretation of Our Examples

One again we see that perturbations in GOST either dissipate totally in some sort of chaos or they remain constrained within very specific patterns. All these examples suggest that interesting non-trivial attacks with fairly small sets, will exist for larger numbers of rounds of GOST. This will be sued in this paper for example in later Fact 5.0.1, however very clearly other very highly regular structural perturbations can also propagate over large numbers of round in GOST and we have not yet discovered all the possibilities for developing quite strong attacks on GOST.

4.9 How Quick Is Diffusion of GOST?

Following [22] it is possible to analyse the propagation of differential and show that the entropy seems ot increase linearly with the number of rounds at a rather slow rate. This is shown on Fig 4.9 below which we borrow from [22]:

Round	Entropy
0	0.0
1	0.0
2	2.81
3	5.61
4	5.72
5	8.19
6	10.92
7	12.31

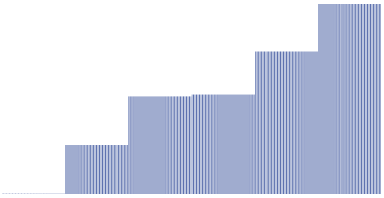

Fig. 7. A relative differential entropy estimation and plot after 1-7 rounds of GOST starting from a simple input set 8000000000000000 when staying within the framework of Fig. 6.

4.10 Propagation For 8 Rounds

The following figure which we borrowed from [22] show that only some general open sets within the framework of Fig. 6 are likely to be obtained after 8 rounds. We are still very from from "good" diffusion for 8 rounds of GOST.

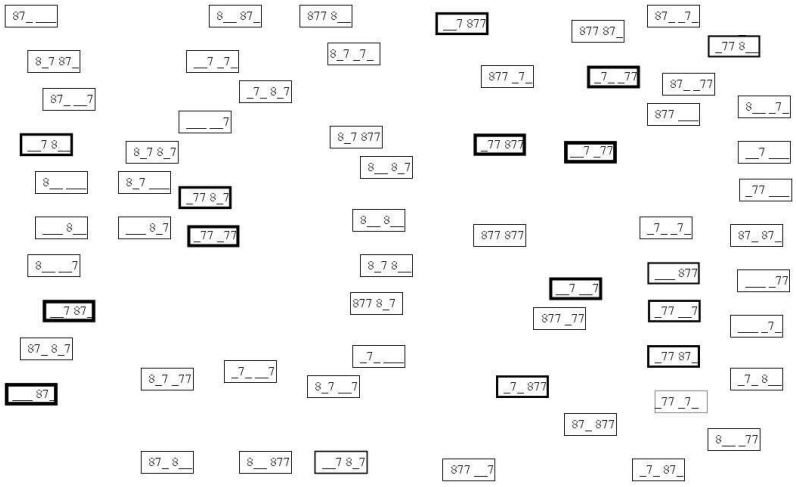

Fig. 8. Propagation of $(80000000, 00000000)$ after 8R of GOST

4.11 Attacks with Single Differentials

We predict that GOST also has some quite strong attacks with single differentials, which however are NOT at all obtained compositions of traditional single-differential characteristics, yet lead to efficient attacks for larger numbers of rounds of GOST. This is simply an inevitable consequence of the existence of strong truncated differentials for large numbers of rounds which is the main property which leads to efficient attacks in this paper. It is easy to see that truncated differential can transition into single differential on both sides for larger number of rounds. This is already seen in Section 4.9 with a single differential on one side.

4.12 Discovery Of New Truncated Differential Properties

Interesting sets of truncated differentials for GOST can be found by trial and error and hill climbing by a basic evolutionary algorithm as follows.

1. We select a set of say $X = 17$ bits at random.
   Some starting points for this algorithm have also been guessed or constructed by hand following heuristics such as in Section 4.3.
2. We use a variant of method called method of "Structures" by Biham and Shamir [5]. More precisely we fix the 64-17 bits and consider all possible plaintexts with such fixed 64-17 bits, and look at how many ciphertext also share the same 64-17 bits. This method gives a quadratic speedup in measuring the propagation probability.
3. We keep a population of some 1000 best sets and mix new sets of 14 bits generated with older sets of 14 in which we flip a few bits. We sample from each distribution half of the time.

More detailed variant of this basic method are studied in [49, 23]. In Table 1 below and on the following pages we show some results obtained with this algorithm through extensive computer simulations (done with several CPUs running for months) by Nicolas Courtois, Theodosis Mourouzis and Guangyan Song.

a		S-box Set Name	Truncated differential set S	P([S]→[S]) 8R	10R	12R
21	0	GostR3411_94_TestParamSet	78780000 F0070783	$2^{-26.6}$	$2^{-37.3}$	2^{-43}
21	1	GostR3411_94_CryptoProParamSet	00070790 787880F0	$2^{-25.8}$	2^{-37}	$2^{-41.6}$
21	2	Gost28147_TestParamSet	80040707 E0787E00	$2^{-24.1}$	$2^{-34.5}$	$2^{-39.7}$
21	3	Gost28147_CryptoProParamSetA	78780060 80070787	$2^{-25.4}$	$2^{-32.9}$	2^{-44}
21	4	Gost28147_CryptoProParamSetB	2460301A 0088C757	2^{-}	2^{-43}	2^{-45}
21	5	Gost28147_CryptoProParamSetC	80008787 7078F800	$2^{-26.1}$	2^{-41}	2^{-44}
21	6	Gost28147_CryptoProParamSetD	78780070 800F0782	$2^{-27.4}$	$2^{-35.4}$	
21	7	GostR3411_94_SberbankHash	80000787 F2787800	$2^{-26.1}$	2	2^{-43}
21	8	GOST ISO 18033-3 proposal	80020787 F0787800	$2^{-24.8}$	$2^{-34.5}$	2^{-42}
21	9	GOST-P proposal	F0027060 73400784	2^{-38}	2^{-41}	2^{-43}

Table 1. Some truncated differential properties with 21 active bits

4.13 On Optimal Size in Truncated Differential Attacks

Recent research shows that truncated differential with 14 bits are the best, and that sets with smaller or bigger size are typically at least slightly worse. This is shown on Fig. 9 and studied in more detail in [24].

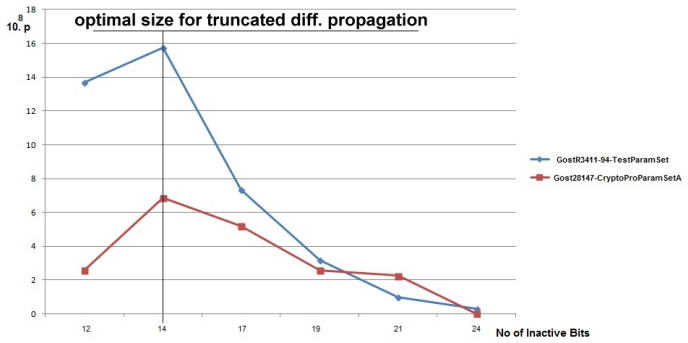

Fig. 9. For each set of GOST S-boxes there exists $D \approx 14$ which maximizes the propagation of truncated differential properties for 8 rounds.

a		S-box Set Name	Truncated differential set S	P([S]→[S]) 8R	10R	12R
14	0	GostR3411_94_TestParamSet	80700700 80700700	$2^{-25.0}$	$\mathbf{2^{-31.8}}$	$\mathbf{2^{-38.6}}$
14	1	GostR3411_94_CryptoProParamSet	00030780 703A0010	$2^{-22.8}$	$2^{-34.5}$	
14	2	Gost28147_TestParamSet	60707800 00000507	$2^{-22.2}$	$2^{-35.7}$	
14	3	Gost28147_CryptoProParamSetA	70780000 80030780	$2^{-23.8}$	$2^{-35.0}$	2^{-41}
14	4	Gost28147_CryptoProParamSetB	C0707000 00000707	$2^{-23.0}$	$2^{-37.3}$	
14	5	Gost28147_CryptoProParamSetC	03070780 70000030	$2^{-25.9}$	2^{-37}	2^{-42}
14	6	Gost28147_CryptoProParamSetD	D0707000 80000207	$2^{-24.7}$	2^{-38}	$2^{-41.5}$
14	7	GostR3411_94_SberbankHash	80080207 80707800	$2^{-22.4}$	$2^{-36.0}$	2^{-42}
14	8	GOST ISO 18033-3 proposal	80000707 20707000	$2^{-22.7}$	2^{-40}	
14	8	GOST ISO 18033-3 proposal	80000707 C0706000	$2^{-25.6}$	2^{-36}	2^{-46}
14	9	GOST-P proposal	50703800 00000707	$2^{-25.2}$	2^{-41}	

Table 2. Some attack sets with 14 active bits

We refer to [13, 26, 22, 49, 23, 24] for more properties of this type. Discovery of new sets is a non-trivial heuristic search task. It shows that the concept of "truncated differentials" is too general, it mixes too many vastly different probabilities to enable straightforward precise explorations. The final result is not always what we expect.

4.14 Can Truncated Differentials Be Joined and Probabilities Multiplied?

This is a provocative question as most of the time we do just that and probabilities are multiplied. Most of the time there are just small discrepancies. The final result is sometimes worse, and also quite frequently it is better than expected. This could be because of existence of additional middle states not captured by a given composition.

Numerous examples which we have studied show that the discovery of new sets is a non-trivial task and not every heuristic works. Even though this heuristic was necessary in the past to construct attacks on larger numbers of rounds of GOST, we need to remark that one **cannot** just combine arbitrary "truncated differentials". More precisely,combining such properties and multiplying the probabilities is not reliable and cannot be trusted to be correct in general. In general it appears that truncated differentials which work well for X rounds will NOT be those which work best of Y rounds, and vice versa, and in general not even close. For each number of rounds, a different dedicated attack need to be developed. This can be seen in Table 2. Additional examples can be found in Section 1 page 160 in [23] and many more can be found [13, 22, 49, 23, 24] and many more are also found in different Tables with results displayed in this section.

We recall from the beginning of Section 4 that a better heuristic methodology and important refinement of "truncated differentials" is proposed in [22, 49].

4.15 Open vs. Closed Sets of Differentials

The paper [22] introduces an important notion of so called "General Open Sets" which are a specific way to partition "truncated differentials" into disjoint sets of differentials which are very similar. This leads to more refined attacks than traditional truncated cryptanalysis [46] and allows us to see that there is hidden very substantial complexity even inside one single set of differentials, cf. Fig. 6 page 18 and Fig. 8 page 21.

A useful notion of a "closure" of these "open sets" is defined in [22]. Closed sets are basically ordinary truncated differentials, while open sets are their subsets which remove significant special cases which heuristically make a big difference given the internal structure of GOST. If we denote $[7007070070070700]$ a closed set, the corresponding open set may be denoted by just 7007070070070700 without square brackets, this notation is used for example in [22].

Closed sets are usually just slightly bigger than some underlying open sets but their behavior is substantially harder to account for. In results such as in Table 2 and 1 we present transitions for closed sets. This what attackers need to use in final attacks.

Now in order to study and reliably predict the behavior of truncated differentials for larger numbers of rounds (e.g. 20) where a computer simulation is too slow to give a precise result, we study open sets for which transition probabilities are expected to be less variable depending on additional factors such as non-uniform sampling of differences in input or output sets of differences. In addition closed to open and open to close transitions can be studied and are already more reliable than closed to closed. All this will be combined below in Section 4.17, cf. also [22].

4.16 Open to Open Transitions

In this section we give some examples of open to open transitions of various sizes which again is a useful building block to find interesting attacks on larger numbers of rounds and to overcome problems outlined in Section 4.14 and in order to implement search/interpolation strategies such as described in [22] and below in Section 4.17 and in Fig 10.

Unlike for other Tables, these results are for 4 rounds. Results of type open $\rightarrow$ open are usually weaker than closed $\rightarrow$ closed [i.e. ordinary truncated differentials] which can be seen as a union of a certain number of open $\rightarrow$ open transitions.

	S-box Set Name	Best set S	P(4R) S$\rightarrow$S
0	GostR3411_94_TestParamSet	70707070 00000707 $\rightarrow$ 70707070 00000707	$2^{-25.0}$
1	GostR3411_94_CryptoProParamSet	70007070 07000707 $\rightarrow$ 70007070 07000707	$2^{-25.8}$
2	Gost28147_TestParamSet	07000078 07007070 $\rightarrow$ 07000078 07007070	$2^{-25.4}$
3	Gost28147_CryptoProParamSetA	70700700 80000700 $\rightarrow$ 70700700 80000700	$2^{-23.0}$
4	Gost28147_CryptoProParamSetB	F0707070 00070007 $\rightarrow$ F0707070 00070007	$2^{-25.0}$
5	Gost28147_CryptoProParamSetC	07070780 70000070 $\rightarrow$ 07070780 70000070	$2^{-23.5}$
6	Gost28147_CryptoProParamSetD	F0070070 08070080 $\rightarrow$ F0070070 08070080	$2^{-24.0}$
7	GostR3411_94_SberbankHash	70070070 00070080 $\rightarrow$ 70070070 00070080	$2^{-26.6}$
8	GOST ISO 18033-3 proposal	0008070F 70000800 $\rightarrow$ 0008070F 70000800	$2^{-26.6}$
9	GOST-P proposal	70707800 00070707 $\rightarrow$ 70707800 00070707	$2^{-27.8}$

Table 3. Some invariant sets open $\rightarrow$ open for 4 rounds of GOST

4.17 Putting It All Together

Following [22, 49] one can construct closed sets for n rounds by decomposing a block cipher in 3 pieces as shown in Fig. 10 and by using open set on the inside and closed sets on the outside, so we study transitions closed $\rightarrow$ open $\rightarrow$ open $\rightarrow$ closed which at the end are just traditional truncated differentials with a precise upper bound on their probability [if it is bigger, it is not a problem, the attacks get easier] . Then we sum all the probabilities which are multiplied along each possible path as shown on Fig. 10. This summation is correct because all middle sets of differentials are disjoint [it is possible to show that our open sets are always disjoint].

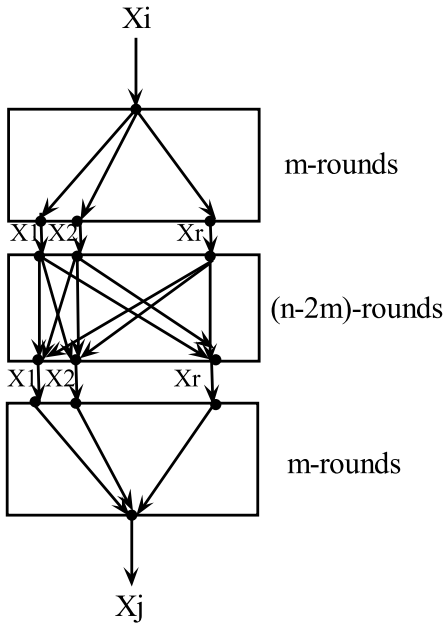

Fig. 10. A construction of a distinguisher for n rounds by splitting in 3 blocks

In order to compute this sum we have created a large databases of various transitions obtained through lengthy simulations, and added things together following Fig. 10. Our final objective is to obtain some final result such as later Fact 5.0.1 page 30. A number of similar cases needed to be checked and compared to find this result. In this paper the final optimization has only been done for one [default] set of GOST S-boxes. For the sake of simplicity we do not provide the full summation and will provide a much simpler independent justification for Fact 5.0.1. However being able to compute the final probability both ways has been an important sanity check in our work.

Part II

Constructing Distinguishers for 20 Rounds of GOST

5 Our Main Distinguisher For 20 Rounds

Our goal is to design an attack on the full 32-round GOST. We will use the same methodology as in [11, 12] guess some key bits and use a distinguisher and we also design a distinguisher for 20 rounds. However our new distinguisher is symmetric and the attack is more complicated. Our attacks operate with distinguishers which count certain individual quite rare events of a certain type. This can be modeled by the so called *Law of Small Numbers* also known simply as a Poisson process [8, 16, 49].

A crucial question is how a differential attack on GOST can cope with false positives. The expected average has frequently two distinct components. There are differentials which occur due to propagation of small Hamming weight differentials for 20 rounds of GOST. Then there are also some other which occur "by accident" for an arbitrary permutation on 64 bis but without very strong differences in the middle of the encryption process. Such additional differences occur not only for a Random Permutation (RP) but also **almost always**, with overwhelming probability $1 - \varepsilon$ for ANY permutation, such as several rounds of GOST block cipher or any other bijection.

In addition we need to quantify precisely the interaction between these two components. Possibly there is some intersection of two sets of events which have different root causes. This is essential if we want to reliably distinguish between 20 rounds of GOST and some other permutation. Finally if overall the number of observed events deviates from an expected average, this can be modeled by the Gauss error function [12, 56, 49, 16].

Fact 5.0.1 *We look at the combination of a non-zero input difference of type* $[80780000, 00000700]$ *and a non-zero output difference of type* $[00000700, 80780000]$ *for 20 rounds of GOST.*

For a typical permutation on 64-bits (does NOT have to be a random permutation, can be GOST with more rounds) we expect that there are 2^{15} *pairs* P_i, P_j *with such differences. The distribution of this number can be approximated by a Gaussian with a standard deviation of* $2^{7.5}$.

For 20 rounds of GOST and for a given random GOST key, there exists two disjoint sets of $2^{15} + 2^{13.9}$ *such pairs* P_i, P_j.

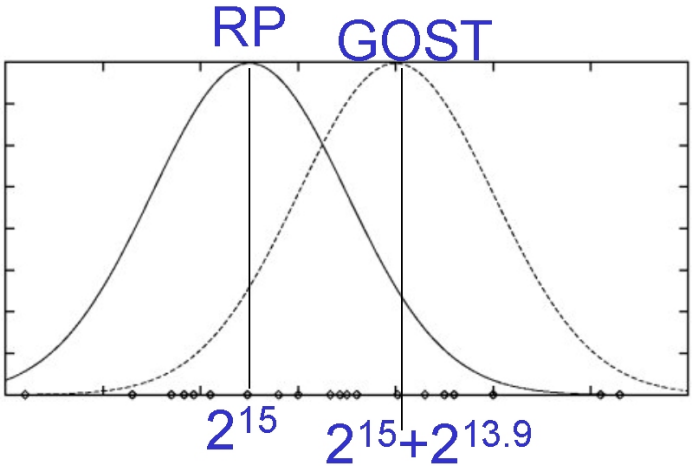

Fig. 11. Signal vs. Noise Differential Distinguisher for 20 Rounds of GOST

These are two entirely disjoint sets of pairs, which can be distinguished by the fact that $2^{13.9}$ *pairs will have the difference* $0x80700700, 0x80700700$ *after 6 rounds from the beginning AND 6 rounds from the end, and none of the* 2^{15} *will have such internal differences.*

The distribution of the sum can be approximated by a Gaussian with an average of about $2^{15} + 2^{13.9}$ *and the standard deviation of* $2^{7.8}$.
This distinguisher works at $2^{6.1}$ *standard deviations.*

Justification: For any permutation, we observe that every single combination of an input differential on 64 bits, and on an output differential on 64 bits, is expected to occur about 0.5 times on average. Indeed we have 2^{127} pairs and about 2^{128} possible sets of two differentials. Now we have $(2^8-1)(2^8-1)$ possibilities of type: $[80780000, 00000700)]to$ (after some permutation OR 20 inner rounds of GOST) $\rightarrow [00000700, 80780000]$. Overall we expect to obtain $0.5 \cdot 2^{8+8} = 2^{15}$ pairs P_i, P_j for a given GOST key, with any of these $2^8 - 2^3 + 1$ differences.

For the actual 20 rounds of GOST the situation is yet more complex. We need to distinguish between pairs which occur "by accident" and those which occur due to "propagation". We are going to develop a precise argument showing that both sets are entirely disjoint and their numbers can be added. In order to do this we are going to give

a precise meaning to the word “propagation” in this precise 20 rounds case: we say that the differential ”propagates” if it goes through two additional differences in the middle as follows:

```
0x80780000 0x00000700
     (7 Rounds)
0x80700700 0x80700700
     (6 Rounds)
0x80700700 0x80700700
     (7 Rounds)
0x00000700 0x80780000
```

Fig. 12. ”Propagation” For 20 rounds With Specific Middle Differentials

Following Fact 3.4.1 and given $2^{64+14-1}$ pairs with the initial difference, we have $2^{77-18.7} = 2^{58.3}$ pairs for the middle 6 rounds.

Then following Fact 3.4.2 the propagation in the next 7 rounds occurs with probability $2^{-22.2}$ on average over GOST keys. Since this is a permutation, the same propagation can be applied backwards in the preceding 7 rounds. Overall, we expect that $2^{58.3-44.4} = 2^{13.9}$ pairs survive.

Now we are going to show that typically, none of these $2^{13.9}$ pairs P_i, P_j is a member of the set of 2^{15} established beforehand. This can be established as follows: for any of the 2^{15} cases which occur naturally at random, we have a non-zero input differential $[80780000, 00000700]$. Then, a computer simulation shows that a differential of type $[80700700, 8070070]$ CAN occur at 7 rounds from the beginning (as in Fig. 12 which is 6+7 rounds from the beginning in GOST) but only with probability of $2^{-16.2}$. Similarly it can also occur 7 rounds from the end, but only with probability of $2^{-16.2}$. Overall we expect that only about $2^{15-16.2-16.2} = 2^{-17}$ pairs P_i, P_j on average will have the “propagation” characteristics according to Fig. 12. Therefore, the two sets are entirely disjoint with a very high probability.

To summarize, we expect to get always a mix of $2^{15} + 2^{13.9}$ cases, which are unlikely to have an intersection, just subject to the standard deviation for each set. Because we are dealing with a sum of a very large number of almost totally independent events, and exactly in the same way as in [11, 12], and due to the Central Limit Theorem [56] these numbers are expected to follow a Gaussian distribution and the standard deviation is expected to be equal exactly to the square root of their expected average number which will be about $2^{15.55}$ for 20 rounds and about $2^{15.0}$ for other permutations.

5.1 Related Research: Distinguishers For 20 Rounds of GOST

We can compare this result to a distinguisher which is used in [12]: where the input difference is $[80000000, 00000000]$, and the output difference is of the form $[00000100, 80600200]$. In the case of the actual 20 rounds of GOST we expect to get $2^4 + 2^5$ such differences, while 2^4 would occur naturally at random for some other permutation.

Other examples of distinguishers for 20 rounds can be found in Section 5 of [22] and in Section 8 of [49]. In [22, 49] we study a more general and more precise methodology for construction of advanced differential attacks for 20 rounds of GOST for various sets of S-boxes. This is also outlined in Part I of this paper, cf. Fig. 10 page 26 and the observations about the predicted vs. actual behavior of attacks with open and closed sets in Section 4.15 page 24. The method outlined in Fig. 10 is actually more precise than in our justification of Fact 5.0.1 and should be seen as a sanity check w.r.t. to potential problems outlined in Section 4.14. In addition as already shown in Section 4.13 we expect that similar attacks exist for other S-boxes, see [13, 14, 22, 49].

Part III

Our Main Attack With Concentric Distinguishers

6 Concentric Distinguishers

So far we have constructed one very good distinguisher for 20 rounds of GOST, cf. Fact 5.0.1. Now the question is as follows. In the similar way as in [12, 49] we want to avoid the necessity to examine all possibilities for the key in the first 6 rounds, and just apply the distinguisher. We want to progressively reduce the key size and the data space on the way, and for this to build a sequence of **concentric distinguishers** for 22, 24 and more rounds which allow early rejection of many cases, so that we are going to examine 20 rounds of GOST with some assumptions on the key and some subset of data much less frequently. This is expected to lead to really efficient attacks on full 32 rounds of GOST. One very simple example of such attack was already described in [12]. More complex distinguishers for 20 rounds have been constructed and studied in [22, 49]. In this paper we study similar and also more complex distinguishers.

6.1 Extending with Additional Weakly Constrained Rounds

We start with our distinguisher property of Fact 5.0.1. This property is going to be extended with a "weakly constrained" differential propagation which occurs with quite a high probability for 6 more rounds on each side.

We also need a model to account for what is going to happen when our assumptions are wrong. Therefore we are going to compare what happens with GOST split as 6+20+6 rounds to a situation which involves a random permutation (RP) as follows. We look at combination of 6 rounds of GOST, some permutation, and 6 rounds of GOST with the same keys in the backwards direction, as in GOST. This is illustrated in Fig. 13 which accounts for both sort of situations. It can represent the full 32-round GOST with 20 rounds in the middle, and it could also be a situation which we wrongly assumed to be the full 32-round GOST and the middle permutation is not exactly a random permutation however it is not at all what we assumed in our attack and we can expect that it might behave as a random permutation for the properties we study.

This leads to the following property which is the core property in our later attack on full 32-round GOST.

Definition 6.1.1 (Alpha Property) *We say that a pair of encryptions for the full 32-round GOST (or for a combination of 6 rounds of GOST, some permutation, and 6 rounds of GOST with reversed keys) has the Alpha Property if the following whole configuration of sets of differentials simultaneously holds:*

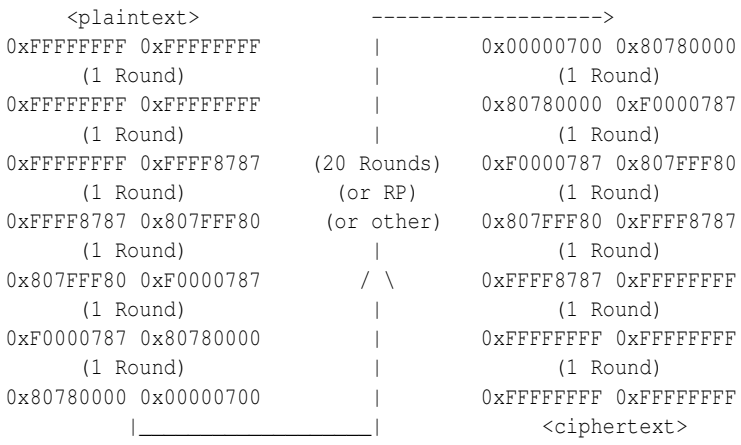

Fig. 13. The Alpha Property

We note that this property is perfectly symmetric (encryption/decryption). They are identical if we look at the cipher in the backwards (decryption) direction, which is very helpful for the attacker given the key scheduling of GOST.

6.2 Alpha Property: GOST vs. Random Permutation

In a similar way as before, a key problem in our distinguisher is that unhappily the Alpha property can occur also "by accident", not at all for the reasons we expect. This question needs to be formulated more precisely, as this property is about differentials also inside GOST, and therefore we cannot just compare GOST to a random permutation. The right question which we need to ask is as follows: in our composition of 6 rounds of GOST, some permutation and then the same 6 rounds in the decryption mode, can we have a fully consistent situation with all the differences which we have in the property Alpha on the outer 2x6 rounds, similar as in Fig. 13.

We have the following result:

Fact 6.2.1 *For the full 32-round GOST and on average over the GOST keys, there exists* $2^{13.0} + 2^{11.9}$ *distinct pairs of plaintexts* $P_i \neq P_j$ *which have the Alpha property.*

If we replace the inner 20 rounds by a random permutation or with GOST with more rounds, we expect only about $2^{13.0}$ *distinct pairs with a standard deviation of* $2^{6.5}$.

Justification: We apply Fact 5.0.1 and obtain $2^{15} + 2^{13.9}$ pairs for the inner 20 rounds with two disjoint sets as explained before. Then it is easy to verify, by a computer simulation, that this provokes the 6 difference sets in the following 6 rounds, simultaneously, with probability as large as $2^{-0.98}$, which is due to slow diffusion in GOST. The same applies in the first 6 rounds. Overall we obtain about $2^{13.9-2.0} \approx 2^{11.9}$ pairs with propagation, and a disjoint set (because subsets of disjoint sets from Fact 5.0.1) with $2^{15-2.0} \approx 2^{13.0}$ pairs which occur by accident.

Again, because we are dealing with a sum of many almost totally independent events, as in [11, 12] and due to the Central Limit Theorem [56], the standard deviation is expected to be exactly the square root of $2^{13.0}$.

7 Guess Then Determine Attacks on GOST

In this section we explain how to compute output bits for a certain round of GOST with incomplete knowledge of all the key bits on which this bit depends in this and previous rounds. We have the following basic fact:

Fact 7.0.2 *The input on 4 bits of any particular S-box in GOST can be computed as:* $a = x + k + c \bmod 16$ *where k are the 4 key bits at this S-box, c is a single carry bit with* $c = 1 \Leftrightarrow x' + k' + c' \geq 16$ *where* x' *and* k' *are the data and the key at the previous S-box, and* c' *is the previous carry bit. This is illustrated in Fig. 14 below.*

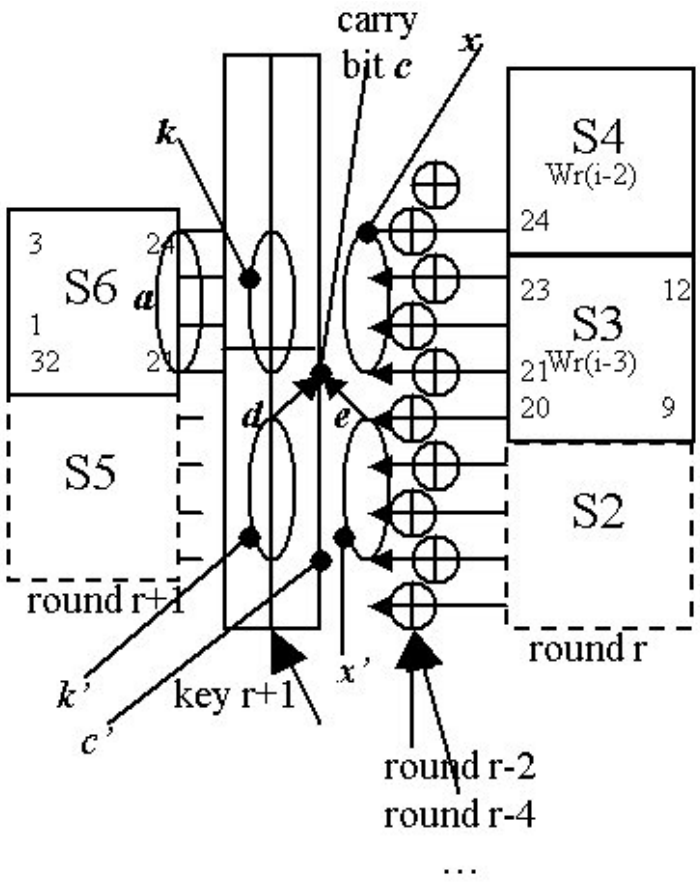

Fig. 14. Computation of the Input Of One S-box With A Carry Bit

Remark: A simplified version of this fact is also shown and used in [21].

In our attack we exploit the weakness of carry propagation in the addition modulo 2^{32}. It is possible to see that carry bits such as c can be guessed with a surprisingly high accuracy. We observe that:

1. We define $Wr(i)$ by the equation $Wr(i)-1=(i-1) \bmod 8$. This corresponds to the number of S-box within 1..8 with wrap-around.
2. The input of each S-box Si in round $r+1$ is
$$a = x+k+c \bmod 16$$
and depends on (i) the 4 key bits k at the entry of this Si and (ii) x obtained from the outputs of two S-boxes in round r with numbers $Wr(i-2)$ and $Wr(i-3)$ XORred with the appropriate bits after round $r-2$ (this part does not change in round $r-1$), and (iii) one carry bit c.
3. The carry bit c is such that
$$c = 1 \Leftrightarrow x'+k'+c' \geq 16$$
where x' and k' are the data and the key at the previous S-box, and c' is the previous carry bit.
4. The previous carry bit influences the result with low probability which will be quantified below.

From here we easily obtain that:

Fact 7.0.3 *Let $i>1$ (with S1 there is no carry entering as in Fig 14.*

We assume that the attacker knows the whole 64-bit output of round $r-2$, the input of one S-box Si at round $r+1$, and the key k at the same Si in round $r+1$, and the state of $Wr(i-2)$ and $Wr(i-3)$ in round r.

Let k' be the unknown key at S-box i-1 in round $r+1$ (cf. Fig 14).

Then we have the following results:

1. *Let d,e be respectively the most significant bits of k' and x'.*
The bit d is obtained from 1 lower bit from $Wr(i-3)$ XORed with the appropriate state from round $r-2$, which is bit 20 on our example at Fig. 14,
If $d=e=1$, we have $c=1$ with probability 1 and we can compute a.
If $d=e=0$, we have $c=0$ with probability 1 and we can compute a.
If $d+e=1$, we have $c=0$ or $c=1$ which are more or less equally likely. Here we get exactly two possibilities for a.
On average we obtain $2\times 1/4\times 1+1/2\times 2=1.5=2^{0.6}$ possibilities for a.
These possibilities for a are computed using only 5 bits of the key and the state of only 2 S-boxes in the previous round.
2. *If the attacker knows the whole 4-bit x' he can compute $k'+x'$ with an interval of incertitude of 8 instead of 16 previously. Thus only with probability 1/4 there will be two answers. Thus on average we obtain $1/4\times 2+3/4\times 1=1.25=2^{0.3}$ possibilities for a.*
3. *The same happens if the attacker knows the whole 4-bit k' but not x'.*
4. *If the attacker knows k' AND the whole state of $Wr(i-4)$ in round r, he can compute c correctly with probability of roughly about $1-2^{-4}$.*

Justification: The first result is straightforward and we will derive the second result. Each of these probabilities can be established by a discrete simulation checking all the possible cases.

The top bit b of x' is known due to $Wr(i-3)$, therefore the expected value of x' is about $x' \approx 8*b+4$, and the whole k' is known. It is easy to see that the expected approximation error (computed as average over 8x8 cases) is $|x'-8*b+4|=1.31$.

We decide that $c = 1 \Leftrightarrow 8 * b + 4 + k' + c' \geq 16$. This will be accurate unless $x' + k' + c' < 16$ AND $8 * b + 4 + k' + c' \geq 16$ or the vice versa with the difference between these two numbers being on average 1.31. Each of these 2 cases occurs with probability of very roughly about $1.3/16 \approx 2^{-4}$. Overall we expect that with probability $1 - 2^{-3}$ our computation of c is correct.

In the similar way we derived the other results.

Important Remark: The intention of this theorem is **not** that is some cases the computations done in our attack will be incorrect and therefore we might miss some cases and the attack would fail. We can handle it in a very different way. Each time we will determine if $c = 1$, by checking $x' + k' + c' \geq 16$ with more or less exact approximations of x' and k' and c', we know exactly the margin of error and know exactly when there will be two possibilities for c. In all these cases we are simply going to include in our enumeration two cases, one with $c = 0$ and one with $c = 1$, which is going to change the 4 outputs of the affected S-box.

8 An Improved Differential Attack on GOST

For the ease of reading we split our attack in 5 stages. All the stages should be seen as a part of the same Depth First Search procedure with several levels.

8.1 Basic Tree Search Methodology

At each level we guess key (or internal data) bits, reject some cases, then guess more key bits to go to the next level, the reject again in order to reduce the number of times we visit a node at the third level, etc. This is illustrated in Fig. 15 below.

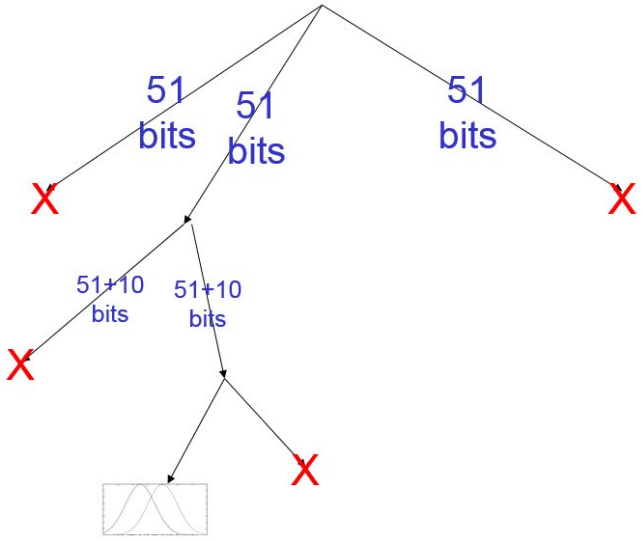

Fig. 15. The principle of progressive filtering, tree exploration, depth-first

The complexity of these attacks is computed level per level: first we compute total time spent at first level after guessing say some 51 bits, this means that there may be up to 2^{51} branches to explore [as an example]. The red cross in Fig. 15 means that we reject this branch by checking some condition. There is usually some cost associated with this checking which needs to be counted, and actually quite frequently the majority of time will be spent rejecting various branches. Even though most tree branches need to be explored only with low probability we have to add the costs of exploring them all.

As we advance in the attack tree the time complexity spent at one level may both increase or decrease. On one side the probability to arrive at this level for a particular set of choices decreases with many early aborts: however in the mean time we need to guess additional bits which increases the number of branches to explore. This is illustrated in Fig: 16 below.

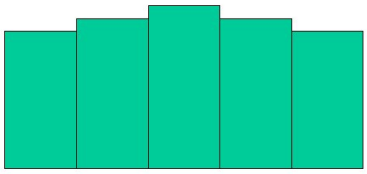

Fig. 16. Example how the complexity of tree exploration splits per each level

The total time complexity will be the sum of these figures, which in case of substantial differences is usually essentially equal to the maximum of these numbers. More advanced and highly optimized attacks tend however to be more balanced and in general we need to sum the cost of all levels carefully.

Now we apply this methodology to GOST. We describe an attack which is essentially a one single depth-first search procedure with several levels which we for better readability we organized in 5 major stages and which is later summarized in Fig. 20 page 45 and a basic description can also be found in [27].

8.2 The Actual Tree Search - First Two Levels

In our attack we are going to work simultaneously on data, which will be pairs of encryptions out of 2^{64} pairs possible, and on keys. We have found that [partly] guessing the data and [partly] determining the keys can be more economical then just guessing the keys as suggested by Fig. 15 above.

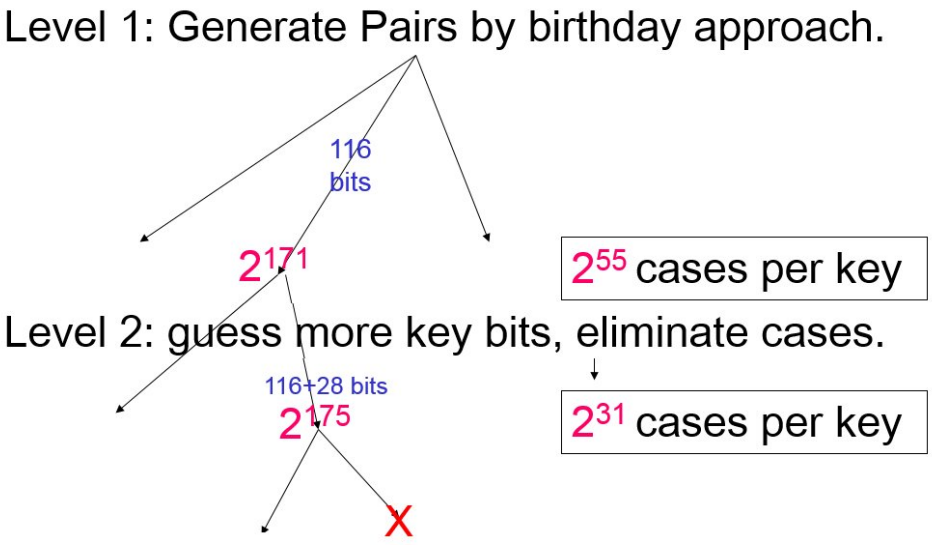

Fig. 17. Revised progressive filtering: working on data pairs AND key bits.

A good measure of progress of such an attack is the number of data pairs (2 encryptions) per key which are yet allowed at any given stage/level in the attack. Initially we need to consider that all 2^{127} pairs are possible for every key. Then we determine a set of sub-keys and at at level 1 we will already have 2^{55} possible pairs per key considered.

Then the number of pairs per key will decrease steadily in further steps of the attack in spite of guessing additional key bits, see the column "per key" in Fig 20 page 45.

8.3 Attack Stage 1 - First 4 and Last 4 Rounds

We proceed as follows:

1. We are given 2^{64} KP which are assumed to be stored in a database.
2. We have the Alpha property cf. Fig. 13 which holds for $2^{13} + 2^{11.9}$ distinct pairs i, j of encryptions for the full 32=6+20+6 rounds, cf. Fact 6.2.1.
3. First we are going to reduce the total number of pairs from 2^{127} to a lower number, by a birthday-like approach which avoids the enumeration of all possible pairs.
4. Given an assumption on a certain number of key bits, we define as **inactive bit** a bit where P_i and P_j collide (the difference is 0) at a certain bit location inside the cipher, **if** our assumption about the key is correct.
5. Our attack will have many steps in which we are going progressively guess some key bits, then reduce the space of pairs considered due to our differentials, which reduce the number of pairs under attack and make it feasible to guess additional key bits at a later stage.
6. We want to write constraints which describe the following events which occur in the first 3 then 4 and the last 3,4 rounds in our property Alpha, cf.Fig. 18.

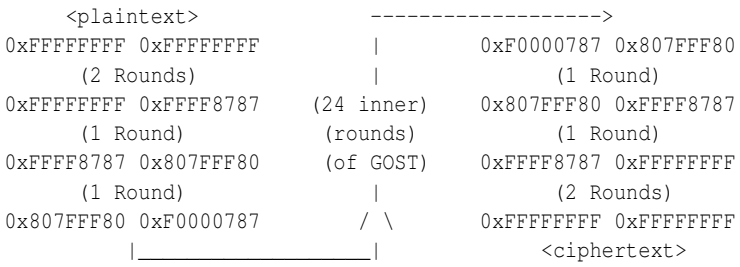

Fig. 18. First 4 and last 4 rounds in the Alpha Property of Fig. 13

   - The output after the addition of the output of S7 and S1 after round 2 gives 8 inactive bits at 0 which are 3-6,11-14. This is implied by the set $0xFFFF8787$ which our Alpha property imposes after round 2.
   - The output after the addition of the output of S4,S5,S6,S7 after round 3 gives 15 inactive difference bits at 0 which are 24-31,1-7 (excluding bit 32). This is implied by the set $0x807FFF80$ which occurs after round 3.
7. We consider and try to guess the following key bits: all key bits at rounds 1,2,3 and 20 key bits for S-boxes S12345 in round 4.
8. We observe that for any guess of these $96 + 20 = 116$ key bits, we get 8+15+13=36 cancelations after rounds 1-4 as explained above, and 36 more cancelations on exactly the same S-boxes with the same keys after round 29 going backwards.
   This can be seen as a collision on 36+36=72 bits, computed as a function of type $f_k(P_i)$ where k represents 116 bits of the key and i is one of the 2^{64} KP cases.

9. For each 116 possible guesses for our selected key bits we compute 2^{64} possible strings on 72 bits for each P_i. Only a proportion of one out of 2^8 values on 72 bits are taken. For any given case i the probability that there is another j for which the 72 bits collide is 2^{-8}.
10. Then we can enumerate in time of maybe $4 \cdot 2^{64}$ CPU clocks some $2^{64-8} = 2^{56}$ possible i or j with $2^{56}/2 = 2^{55}$ distinct pairs i, j which collide on these 72 bits.
    Another way of looking at this is as follows: there are (we do NOT ever enumerate all of them) about 2^{127} pairs P_i, P_j and there are about 2^{127} differences $f_k(P_i) - f_k(P_j)$ on 72 bits. Some $2^{127-72} = 2^{55}$ of these differences will have all the 72 bits at 0.
11. These 2^{55} pairs per key assumption can be enumerated efficiently. A simplified method is as follows: We make a hash table where at address being a hash of $f_k(P_i)$ on 72 bits, and we store i as well. Each time the value is already taken we output a collision. We will output a list of 2^{55} pairs P_i, P_j.
    Memory required is roughly about 2^{70} bytes.
12. The total time spent in these steps of the attack should not exceed 2^{116+64} times the cost of computing roughly speaking 1 round of GOST.
    It is not needed to do as much work as computing 2^{116} times 4 first rounds of GOST and 4 last rounds of GOST. Basically the cost of computing the first 3+ and the last 3+ rounds of GOST can be neglected. More precisely it will be amortized in 2^{20} sub-cases of the 2^{96} cases, in which we just need to evaluate 4 S-boxes in round 3 and 4 S-boxes in round 30, which is roughly feasible to do in most an equivalent of 1 round of GOST.
    Therefore we estimate that we need only about $2^{116+64} \cdot 8$ CPU clocks, which could be seen as an equivalent of roughly about 2^{174} GOST encryptions.

To summarize, we can thus in total overall time equivalent to about 2^{174} GOST encryptions and with memory of about 2^{70} bytes, enumerate $2^{171} = 2^{116+55}$ cases of type k_{116}, i, j. We get on average 2^{55} possible pairs i, j for each key assumption on 116 bits.

In Fig. 20 we summarize all the current and further steps of our attack.

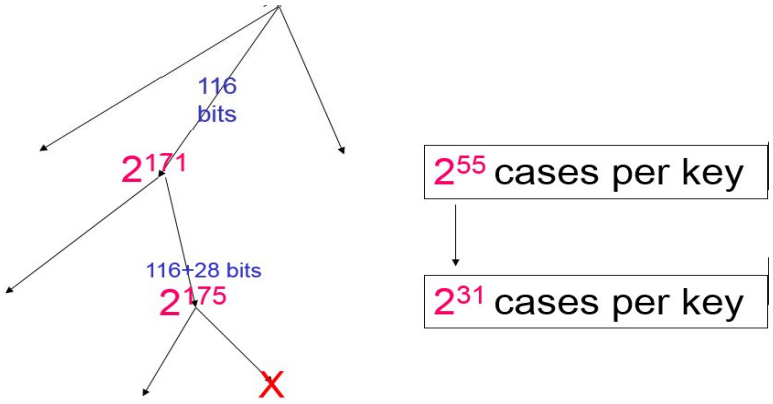

Fig. 19. Summary of selected initial stages in our attack.

guess key at S-boxes	correct	difference	new bits to cancel after outputs of	after round	new inactive bits (60 in total)		enumerate cases	per key	key bits assum.	time GOST encrypt.
								2^{127}		
all bits in R12	2^{-64}	FFFF8787	S7,S1	2,31	8	4-7, 12-15	birthday			
S3*4567R3	2^{-20}	807FFF80	S456^{3}7	3,30	15	24-31,1-7	attack			
S812R3 S12345R4	2^{-32}	F0000787	S2345^{1}	4,29	13	16-28	2^{55+116}	2^{55}	116	2^{174}
							2^{171}			
S6R4 S78R5	2^{-12}	80780000	S8	5,28	4	8-11	$2^{171+12-4-4}$	2^{47}	128	2^{179}
S7R4 S1R5	2^{-8}	80780000	S1	5,28	4	12-15	$2^{175+8-4-4}$	2^{39}	136	2^{174}
S8R4 S2R5	2^{-8}	80780000	S2	5,28	4	16-19	$2^{175+8-4-4}$	2^{31}	144	2^{174}
							2^{175}			
S3R5 S4^{1}5R6	2^{-9}	00000700	S5^{3}	6,27	3	29,30,31	$2^{175+9+1.2-3-3}$	$2^{26.2}$	153	2^{176}
S4R5 S6R6	2^{-8}	00000700	S6	6,27	4	32,1-3	$2^{179.2+8-4-4}$	$2^{18.2}$	161	$2^{178.2}$
$2^{18.2}+2^{11.5}$	is	chosen	at	$2^{2.4}$	standard		deviations	$*2^{-24}$	to survive	
except for the	right	161 bits	we	have	to remain		$2^{179.2-24}$	or 2^{-6}	per key only	
S56R5 S781R6 S23R7	2^{-28}	00000700 80000000	S8^{1} S3	6,27 7,26	1+ 4	8 20-23	$2^{155.2+28-5-5}$	$2^{8.2*}$	189	2^{175}
$2^{8.2}+2^{10}$	is	chosen	at	$2^{5.9}$	standard		deviations	-	certitude	
									total	2^{179}

Fig. 20. Summary of all major steps in our main attack on GOST

8.4 Attack Stage 2 - Working on Rounds 5 and 28

Now we are going to work on additional key assumptions with the objective to decrease the number of pairs per key from 2^{55} to a much lower number so that we will be able later at Stage 4 apply the distinguisher given by Fact 6.2.1.

1. Now we look at the difference $0x80780000$ obtained after round 5, where outputs of S-boxes S8, S1 and S2 are 'newly' inactive which is a cancelation on 12 bits after round 5, cf. earlier Fig. 13 page 36 or Fig. 21 page 48.
   First we will work on S8, then on S1, then on S2.
2. First we guess additional 4+4 key bits. The situation is the same as in Fig. 14 with boxes S78 at round 5 depending mostly on boxes S456 in round 4.
   We guess 4 bits at S-box S6 in round 4, needed only to compute the bit 31 entering S8 at round 5, and the 4 key bits at S8 in round 5, and an approximation on the 4-key bits at S7 in round 5, which together with outputs of S4 and 1 bit from S5 in round 4, can be used to compute the carry entering S-box S8 at round 5 with probability of about $1-2^{-4}$ (cf. Fact 7.0.3).
3. More over and quite importantly we do **not** allow any errors in our computations. In rare cases where there is an ambiguity about the carry, because for example we have 15 and the carry added from S6 in round 5 could matter, we simply check both cases. This leads to a negligible increase in the total number of cases checked from about 2^{171+12} to about $(1+2^{-4})2^{171+12}$, see Fact 7.0.3.3. For simplicity we ignore these additional numbers which are negligible compared to other numbers in this attack.
   Later during the attack, when the key at $S6$ and early S-boxes becomes known, these additional cases will be eliminated instantly. In fact we can also leave these additional cases, everything we do later in our attack can tolerate a small proportion of additional incorrect cases.
4. With these 12 new key bits, we can enumerate 2^{171+12} cases k_{116+12}, i, j. In each cases with probability 2^{-4} the 4 bits XORed to the output of S-box S8 become inactive at round 4, and with probability 2^{-4} they also become inactive at round 29.
5. Accordingly in time of about 2^{177} computations of 2/32 full GOST, which is about 2^{179} GOST computations, (assuming one takes 2^9 CPU clocks). We reject most cases except $2^{171+12-4-4} = 2^{175}$ cases k_{128}, i, j.
6. This, is 2^{47} cases per key.
7. Now we guess 8 more key bits. These are 4 bits at S-box S7 in round 4 which output 3 is needed to compute the input of S1 in round 5 (there is no carry entering S1). We also guess 4 key bits at S1 in round 5.
8. Now we have an enumeration of 2^{175+8} cases k_{136}, i, j, where we now have 136 key bits. In this list with probability 2^{-4} the 4 bits XORed to the output of S-box S1 become inactive at round 4, and with probability 2^{-4} they also become inactive at round 29.
9. Accordingly in time of about 2^{175+8} computations of 2/32 full GOST, which is about 2^{174} GOST computations, we have an enumeration $2^{175+8-4-4} = 2^{175}$ cases k_{136}, i, j.
10. Now we guess 8 more key bits. These are 4 bits at S-box S8 in round 4 which outputs are 8-11 and which are needed to compute the input of S-box S2 in round

5 (the carry entering S2 is already known for S1 in round 5 above). We also guess 4 key bits at S2 in round 5.

11. Thus we consider the enumeration of 2^{175+8} cases k_{144}, i, j, where we now have 144 key bits. In this list with probability 2^{-4} the 4 bits XORed to the output of S-box S2 become inactive at round 4, and with probability 2^{-4} they also become inactive at round 29.

Accordingly in time of about 2^{175+8} computations of 2/32 full GOST, which is about 2^{174} GOST computations, we enumerate about $2^{175+8-4-4} = 2^{175}$ cases k_{144}, i, j. We are left with 2^{31} pairs i, j on average for each key assumption on 144 bits which will be the cases which we will check in later steps of our attack.

For the right key assumption we will also obtain the $2^{11.9}$ cases which have the property Alpha for the correct GOST key

8.5 Attack Stage 3

We will continue the process of guessing additional key bits and decreasing the number of cases per key assumption.

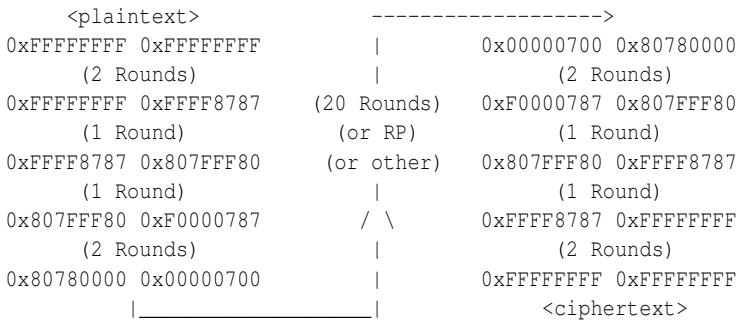

Fig. 21. First 6 and last 6 rounds in the Alpha Property of Fig. 13

1. At this stage, in each case, we know all key bits in rounds 1,4 and key bits S-boxes S1278 in round 5, for a total of 144 key bits.
2. Now in round 6 we have the difference 0xF0000787 which becomes 0x00000700 , cf. Fig. 21. The S-box outputs which are going to become inactive are: 3 outputs of S5 with numbers 29,30,31, the whole of S6 with numbers 32,1-3, and one lower bit of S8 with number 8.
3. We will first work on S5, then on S6, and later on S8.
4. First we guess 9 key bits: for S3 at round 5, and for S5 at round 6 and just one most significant bit for S4 at round 6. We have 3 inactive bits 29-31. Following Fact 7.0.3. this allows to determine exactly the carry bit c with probability $1/2$, and the attacker knows in which case it is (when $d = e$, cf. Fact 7.0.3.1.), and otherwise we have two cases to include (when $d \neq e$, cf. Fact 7.0.3.1.).
   Overall on average we have $(1+2)/2 \approx 2^{0.6}$ more cases to check and we compute the output of S5 at round 6 about $2^{175+9+0.6} = 2^{177}$ times.
5. In addition we also need to compute the output of S5 at round 27 in each of these cases. In the same way sometimes this generates 1 or 2 cases to check, and overall we get another factor of $2^{0.6}$.
6. Accordingly in time of about $2^{175+9+0.6+0.6}$ computations of 2/32 full GOST, which is about $2^{176.2}$ GOST computations, we obtain a list of $2^{175+9+1.2-3-3} = 2^{179.2}$ cases k_{153}, i, j. This is $2^{26.2}$ cases per key.
7. Then we guess 8 more key bits: for S4 at round 5, and for S6 at round 6. We have 4 inactive bits 32,1-3.
8. Accordingly in time of about $2^{179.2+8}$ computations of 2/32 full GOST, which is about $2^{178.2}$ GOST computations, we obtain a list of
   $2^{179.2+8-4-4} = 2^{179.2}$ cases k_{161}, i, j.
9. This is only $2^{18.2}$ cases per key on 161 bits which is within reach of our distinguisher attacks.
10. The total time spent in all the above steps is is about $2^{178.5}$ GOST computations, and probably only half of this number on average is needed.

8.6 Attack Stage 4

Now we are going to be able to see if 161 key bits are right or wrong.

We recall Fact 6.2.1. For the full 32-round GOST and on average over the GOST keys, there exists two disjoint sets with $2^{13}+2^{11.9}$ distinct pairs of plaintexts $P_i \neq P_j$ which have the Alpha property.

We have $2^{18.2}$ cases per key, which for the right key on 161 bits contains these correct $2^{13}+2^{11.9}$ cases. All these cases come from the fact that we have independently in the first 6 and the last 6 rounds, checked if certain set of twice 55 differences are at 0, which gives 2^{17} pairs surviving. We have also produced an overhead of some $2^{1.2}$ additional cases which result from incertitude due to further unknown key bits which gives $2^{18.2}$ pairs total.

As before, It is clear that these $2^{18.2}$ pairs obtained in the specific case of the right 161-bit key, occur at random due to the random intersection between cases which may occur at the beginning of GOST, and at the end of GOST, without correlation between these events.

It is easy to see that $2^{18.2}$ such pairs on average, with an expected standard deviation of about $2^{9.1}$, are still going to occur if we explicitly exclude about $2^{64+14-1} \ll 2^{127}$ cases where a difference of type $[80700700, 80700700]$ occurs after 6+7 rounds AND at 6+7 rounds from the end, which as explained for Fact 5.0.1 occurs with very low probability of about $2^{-32.4}$ and in fact less, because in our case it is not yet certain that the difference is as expected after round 7.

However because the $2^{11.9}$ cases do ALL have differences of type $[80700700, 80700700]$ after 6+7 rounds AND at 6+7 rounds from the end, the two sets are disjoint. To summarize we obtain the following result:

Fact 8.6.1 *After Stage 3 of our attack, if the 161 bits are wrong, most of the time (this will be quantified below) we get about* $2^{18.2}$ *cases per key.*

We assume that the attacker will decide that the key on 161 bits is correct if he sees at least $2^{18.2}+2^{11.5}$ *cases for this key. Otherwise he will reject it.*

The correct 161-bits key will be accepted with probability of 95%.

Incorrect 161 bits will be accepted with probability of about 2^{-39}.

Justification: A correct 161 bits should give about $2^{18.2}+2^{11.9}$ cases with standard deviation of $2^{9.1}$ and will be rejected only if we are below $2^{18.2}+2^{11.5}$ cases which is on one side of and outside of $(2^{11.9-11.5})/2^{9.1} = 2$ standard deviations. By applying the Gauss error function [56] we see that a correct key will be accepted with probability of about 95%.

If the 161 bits are wrong, we are outside of and on one side of, $2^{11.5-9.1} = 2^{2.4}$ standard deviations. Here the Gauss error function [56] gives a probability only about 2^{-24}.

8.7 Attack Stage 5

We need to do some additional guessing and filtering. Up till now, with total time of up to about $2^{178.5}$ GOST computations, we are able to enumerate $2^{179.2-24} = 2^{155.2}$ cases k_{161}, i, j. Our 161 bits of the key are all the bits for the first 4 rounds, and 24 bits at round 5 for S781234, and 9 bits at round 6 for S5,S6 and one bit at S4.

1. We guess the remaining 8 bits to complete round 5 with boxes S56. Then we guess the key at boxes S7181 at round 6 and at S213 in round 7. This is a total of 28 bits. For simplicity we guess all these bits (a more refined approach is NOT needed because the total time spent in this step is small).
2. The output after S8 in round 6 needs to cancel on 1 bit which is number 8, and the output of S3 in round 7 needs to cancel on 4 bits which are 20-23. This is implied by the sets $0x00000700$ and $0x80000000$ in the Alpha property obtained after round 6 and 7.
3. Accordingly in time of about $2^{155.2+28}$ computations of 2/32 full GOST, which is about 2^{175} GOST computations, we reject most cases except some $2^{155.2+28-5-5} = 2^{173.2}$ cases k_{189}, i, j.
   This seems to be about 2^{-16} per key on average, which comes from the fact that only some 161-bit sub-keys are present in the keys on 189 bits. However if we look only at 2^{165} keys on 189 bits which are actually present, we have $2^{8.2}$ cases per key.
4. We assume that the attacker will reject all cases where the count is less than $2^{8.2} + 2^{10}$.
5. Then it is easy to see that if the key is correct, it will be accepted with probability very close to 1.
6. If the key is wrong, we observe that $2^{8.2} + 2^{10}$ is outside $2^{5.9}$ standard deviations. Here the Gauss error function [56] gives a figure much smaller than 2^{-256}.

Summary: Thus given 2^{64} KP and in an average time of about 2^{179} GOST computations, we are able to determine with certitude 189 bits of GOST key. The remaining 66 bits can then be found by brute force. The attack was designed to work for 95% of GOST keys.

Applicability: Current attack was optimized for just one set of GOST S-boxes. The space of possible variants ot this attack is very large. It is not true to be believe that this attack would not work for a certain well-designed set of S-boxes [53, 52]. On the contrary, similar results exist for any set of S-boxes cf. [22, 13, 14, 49], cf. Sections 4.4 and 4.12 etc. We conjecture that for any set of bijective S-boxes in GOST (the worst case) there is a differential attack substantially faster than brute force and essentially the same as the one presented in this paper.

9 Conclusion

GOST 28147-89 is a well-known block cipher and the official encryption standard of the Russian Federation. On page 334 of the well known 1996 edition of book "Applied Cryptography" [55] by Bruce Schneier we read that "Against differential and linear cryptanalysis, GOST is probably stronger than DES". In 2000 Russian researchers claimed that as few as 7 rounds out of 32 are sufficient to protect GOST against DC, cf. [34, 33]. In the same year Japanese researchers [54] show that more powerful differential attacks exist, exploiting sets of differentials [54]which allow to break about 13 rounds of GOST out of 32.

Many new attacks on GOST have been proposed since 2011 [9, 26, 37, 10–12, 15, 31, 21, 19, 20, 49, 22, 23]. In 2011 Courtois and Misztal have found new differential sets for GOST [10] most of which can also be seen as "truncated" differential attacks [46]. If one exploits the key scheduling one can break full GOST faster than brute force [11] which attack was further refined in [12] to achieve about 2^{224} and further work and refinements can be found in [49, 22, 23]. In this paper we present a first fully optimized improved advanced differential attack on full 32-round GOST. Given 2^{64} KP we can recover the full 256-bit key for GOST within only about 2^{179} GOST computations on average for a success probability of 95%. The memory is about 2^{70} bytes. **This is the fastest single-key attack on GOST found so far.** The best previous single-key attack on GOST was 2^{192} of [31] which could be improved to 2^{191} in [26]. Our attack is several thousands of times faster. Our 2^{179} is an approximation and inexact result which can probably yet be improved slightly. At this moment our attack was optimized only for one set of S-boxes, but there is little doubt that it works for other S-boxes, cf. [13].

What's New. In a very recent paper about advanced differential cryptanalysis, in Section 1.1. page 3 of [1] we read: *Truncated differentials, [...] in some cases allow to push differential attacks one or two rounds further.*
Our research on GOST shows that we can **gain not two but closer to 20 rounds** compared to previously known differential attacks [33, 34, 54].

Multiple-Key Attacks. In practice ciphers are NOT used with single keys. If GOST is used with many different keys, there are better advanced differential-black box reduction attacks on GOST, with only 2^{32} of data per key instead of 2^{64}, which can recover some but not all GOST keys, at a total cost as low as 2^{101} GOST computations total to find one key, see [20, 26].

References

1. Martin Albrecht and Gregor Leander: *An All-In-One Approach to Differential Cryptanalysis for Small Block Ciphers*, preprint available at eprint.iacr.org/2012/401/.
2. Lyudmila K. Babenko, Evgeniya Ishchukova: *Differential analysis of GOST encryption algorithm*, In SIN 2010, pp. 149-157, ACM, 2010.
3. Alex Biryukov, David Wagner: *Advanced Slide Attacks*, In Eurocrypt 2000, LNCS 1807, pp. 589-606, Springer 2000.
4. Eli Biham, Adi Shamir, *Differential Cryptanalysis of DES-like Cryptosystems*, Journal of Cryptology, vol. 4, pp. 3-72, IACR, 1991.
5. Eli Biham, Adi Shamir, *Differential cryptanalysis of the full 16-round DES*, In Crypto'92, pp. 487-496, LNCS 740, Springer-Verlag, 1992.
6. Matt Blaze, *"Re: Reverse engineering and the Clipper chip"*, Newsgroup post at sci.crypt, 15 August 1996, https://groups.google.com/group/sci.crypt/msg/5cd14a329372cc5a
7. Nicolas Courtois: *The Best Differential Characteristics and Subtleties of the Biham-Shamir Attacks on DES*, On eprint.iacr.org/2005/202.
8. Ladislaus Bortkiewicz: *The Law of Small Numbers;* (Das Gesetz der kleinen Zahlen), book, in German, 1898.
9. Nicolas Courtois: *Security Evaluation of GOST 28147-89 In View Of International Standardisation*, in Cryptologia, Volume 36, Issue 1, pp. 2-13, 2012. http://www.tandfonline.com/toc/ucry20/36/1 An earlier version which was officially submitted to ISO in May 2011 can be found at http://eprint.iacr.org/2011/211/.
10. Nicolas Courtois, Michał Misztal: *Aggregated Differentials and Cryptanalysis of PP-1 and GOST*, In CECC 2011, 11th Central European Conference on Cryptology. In Periodica Mathematica Hungarica Vol. 65 (2), 2012, pp. 1126, Springer.
11. Nicolas Courtois, Michał Misztal: *First Differential Attack On Full 32-Round GOST*, in ICICS'11, pp. 216-227, Springer LNCS 7043, 2011.
12. Nicolas Courtois, Michał Misztal: *Differential Cryptanalysis of GOST*, In Cryptology ePrint Archive, Report 2011/312. 14 June 2011, http://eprint.iacr.org/2011/312.
13. Nicolas T. Courtois, Theodosis Mourouzis, Michał Misztal, Jean-Jacques Quisquater, Guangyan Song: *Can GOST Be Made Secure Against Differential Cryptanalysis? In Cryptologia, vol. 39, Iss. 2, 2015, pp. 145-156.*
14. *Nicolas Courtois, Theodosis Mourouzis, Advanced Truncated Differential Attacks Against GOST Block Cipher and Its Variants*, In Computation, Cryptography, and Network Security, Springer, pp. 351-380, 2015.
15. Nicolas T. Courtois, Theodosis Mourouzis: *Advanced Differential Cryptanalysis and GOST Cipher*, accepted for a 30 minute oral presentation at the 3rd IMA Conference on Mathematics in Defence At Tom Elliott Conference Centre, QinetiQ, Malvern, UK on Thursday 24 October 2013. 6-pages paper in CD-ROM and web proceedings planned.
16. Nicolas T. Courtois, Theodosis Mourouzis: *Advanced Differential Cryptanalysis of SIMON 64/128* , preprint, 2015.
17. Nicolas Courtois: *Cryptanalysis of Two GOST Variants With 128-bit Keys*, In Cryptologia vol. 38(4), pp. 348-361, 2014. At http://www.tandfonline.com/doi/full/10.1080/01611194.2014.915706.
18. Nicolas Courtois, Jerzy A. Gawinecki, Guangyan Song: *Contradiction Immunity and Guess-Then-Determine Attacks On GOST*, In Tatra Mountains Mathematic Publications, Vol. 53 no. 3 (2012), pp. 65-79.
19. Nicolas T. Courtois: *Cryptanalysis of GOST in the Multiple Key Scenario*, In post-proceedings of CECC 2013, Tatra Mountains Mathematical Publications. Vol. 57, no. 4 (2013), p. 45-63. At http://www.sav.sk/journals/uploads/0124133006Courto.pdf

20. Nicolas Courtois: *On Multiple Symmetric Fixed Points in GOST*, In Cryptologia, Volume 39, Issue 4, 2015, pp. 322-334, http://www.tandfonline.com/doi/full/10.1080/01611194.2014.988362.
21. Nicolas T. Courtois: *Low-Complexity Key Recovery Attacks on GOST Block Cipher*, In Cryptologia, Volume 37, Issue 1, pp. 1-10, 2013.
22. Nicolas T. Courtois, Theodosis Mourouzis: *Enhanced Truncated Differential Cryptanalysis of GOST*, in SECRYPT 2013, Reykjavik, July 2013, http://www.nicolascourtois.com/papers/sec13.pdf
23. Nicolas T. Courtois, Theodosis Mourouzis: *Propagation of Truncated Differentials in GOST*, in proc. of SECURWARE 2013, http://www.thinkmind.org/download.php?articleid=securware_2013_7_20_30119
24. Nicolas Courtois, Theodosis Mourouzis, Anna Grocholewska-Czurylo and Jean-Jacques Quisquater: *On Optimal Size in Truncated Differential Attacks*, In post-proceeding of CECC 2014 conference, Studia Scientiarum Mathematicarum Hungarica.
25. Nicolas Courtois, Gregory V. Bard: *Algebraic Cryptanalysis of the Data Encryption Standard*, In Cryptography and Coding, 11-th IMA Conference, pp. 152-169, LNCS 4887, Springer, 2007. Preprint available at eprint.iacr.org/2006/402/. Also presented at ECRYPT workshop Tools for Cryptanalysis, Krakow, 24-25 September 2007.
26. Nicolas Courtois: *Algebraic Complexity Reduction and Cryptanalysis of GOST*, Monograph study on GOST cipher, 2010-2014, 224 pages, available at http://eprint.iacr.org/2011/626.
27. Nicolas Courtois: *An Improved Differential Attack on Full GOST*, in "The New Codebreakers a Festschrift for David Kahn", LNCS 9100, Springer, March 2016.
28. Don Coppersmith, *The Data Encryption Standard (DES) and its strength against attacks*, IBM Journal of Research and Development, Vol. 38, n. 3, pp. 243-250, May 1994.
29. Don Coppersmith, *The development of DES*, *Invited Talk, Crypto'2000, August 2000.*
30. *Differential Cryptanalysis, wikipedia entry 05 May 2013, http://en.wikipedia.org/wiki/Differential_cryptanalysis.*
31. *Itai Dinur, Orr Dunkelman and Adi Shamir: Improved Attacks on Full GOST*, FSE 2012, LNCS 7549, pp. 9-28, 2012, early version available at http://eprint.iacr.org/2011/558/.
32. Ali Doganaksoy, Bariş Ege, Onur Koçak and Fatih Sulak: *Cryptographic Randomness Testing of Block Ciphers and Hash Functions*, In http://eprint.iacr.org/2010/564.
33. Vitaly V. Shorin, Vadim V. Jelezniakov and Ernst M. Gabidulin: *Linear and Differential Cryptanalysis of Russian GOST*, Preprint submitted to Elsevier Preprint, 4 April 2001
34. Vitaly V. Shorin, Vadim V. Jelezniakov and Ernst M. Gabidulin: *Security of algorithm GOST 28147-89*, (in Russian), In Abstracts of XLIII MIPT Science Conference, December 8-9, 2000.
35. I. A. Zabotin, G. P. Glazkov, V. B. Isaeva: *Cryptographic Protection for Information Processing Systems*, Government Standard of the USSR, GOST 28147-89, Government Committee of the USSR for Standards, 1989. An English translation can be found at ftp.funet.fi/pub/crypt/cryptography/papers/gost/russian-des-preface.ps.gz
36. A Russian reference implementation of GOST implementing Russian algorithms as an extension of TLS v1.0. is available as a part of OpenSSL library. The file gost89.c contains eight different sets of S-boxes and is found in OpenSSL 0.9.8 and later: http://www.openssl.org/source/
37. Takanori Isobe: *A Single-Key Attack on the Full GOST Block Cipher*, In FSE 2011, pp. 290-305, Springer LNCS 6733, 2011.
38. Jialin Huang and Xuejia Lai: *What is the Effective Key Length for a Block Cipher: an Attack on Every Block Cipher*, eprint.iacr.org/2012/677.

39. Orhun Kara and Ferhat Karakoç: *Fixed Points of Special Type and Cryptanalysis of Full GOST*. In CANS 2012, LNCS 7712, pp 86-97, 2012.
40. L.V. Kovalchuk: *Generalized Markov ciphers: evaluation of practical security against differential cryptanalysis*, in: Proc. 5th All-Russian Sci. Conf. "Mathematics and Safety of Information Technologies" (MaBIT-06), 25-27 Oct. 2006, MGU, Moscow, pp. 595-599, 2006 [in Russian].
41. A.N. Alekseychuk, L.V. Kovalchuk, and S.V. Pal'chenko: *Cryptographic parameters of s-boxes that characterize the security of GOST-like block ciphers against linear and differential cryptanalysis*, Zakhist Inform., No. 2, pp. 12-23, 2007 [in Ukrainian].
42. L. V. Kovalchuk: *Upper-bound estimation of the average probabilities of integer-valued differentials in the composition of key adder, substitution block, and shift operator*, In Cybernetics And Systems Analysis Vol. 46, Number 6 (2010), pp. 936-944, Springer.
43. L. V. Kovalchuk and O. A. Sirenko: *Analysis of mixing properties of the operations of modular addition and bitwise addition defined on one carrier*, In Cybernetics And Systems Analysis Vol. 47, Number 5 (2011), pp. 741-753, Springer.
44. A. N. Alekseychuk and L. V. Kovalchuk: *Towards a Theory of Security Evaluation for GOST-like Ciphers against Differential and Linear Cryptanalysis*, Preprint 9 Sep 2011, `http://eprint.iacr.org/2011/489`.
45. Lars R. Knudsen: Block Ciphers The Basics, Spring 2011, `https://www.cosic.esat.kuleuven.be/ecrypt/courses/albena11/slides/LRK-basics.pdf`.
46. Lars R. Knudsen: *Truncated and Higher Order Differentials*, In FSE 1994, pp. 196-211, LNCS 1008, Springer.
47. Gregor Leander, Axel Poschmann: *On the Classification of 4 Bit S-Boxes, In Proceedings of WAIFI'07, 1st international workshop on Arithmetic of Finite Fields.*
48. *Nick and Alex Moldovyan: Innovative Cryptography*, textbook, 2nd edition, Charles River Media, Boston, 2007.
49. Theodosis Mourouzis: *Optimizations in Algebraic and Differential Cryptanalysis*, PhD thesis, under superivsion of Dr. Nicolas T. Courtois, University College London, January 2015, `http://discovery.ucl.ac.uk/1462141/2/PhD_Thesis_Theodosis_Mourouzis.pdf`
50. Kaisa Nyberg: *Differentially Uniform Mappings for Cryptography*, In Eurocrypt 1993, LNCS 765, pp. 55-64, Springer 1994.
51. Axel Poschmann, San Ling, and Huaxiong Wang: *256 Bit Standardized Crypto for 650 GE GOST Revisited*, In CHES 2010, LNCS 6225, pp. 219-233, 2010.
52. Vladimir Rudskoy and Andrey Chmora: *Working draft for ISO/IEC 1st WD of Amd1/18033-3: Russian Block Cipher GOST, ISO/IEC JTC 1/SC 27 N9423, 2011-01-14, MD5=feb236fe6d3a79a02ad666edfe7039aa*
53. *Vladimir Rudskoy, Andrey Dmukh: Algebraic and Differential Cryptanalysis of GOST: Fact or Fiction*, In CTCrypt 2012, Workshop on Current Trends in Cryptology, affiliated with 7th International Computer Science Symposium in Russia (CSR-2012), 2 July 2012, Nizhny Novgorod, Russia. Full papers will be submitted and published in a special issue of Russian peer-review journal Mathematical Aspects of Cryptography. An extended abstract is available at: `https://www.tc26.ru/invite/spisokdoc/CTCrypt_rudskoy.pdf` slides are available at: `https://www.tc26.ru/documentary%20materials/CTCrypt%202012/slides/CTCrypt_rudskoy_slides_final.pdf`
54. Haruki Seki and Toshinobu Kaneko: *Differential Cryptanalysis of Reduced Rounds of GOST. In SAC 2000, LNCS 2012, pp. 315-323, Springer, 2000.*
55. *Bruce Schneier: Section 14.1 GOST*, in *Applied Cryptography*, Second Edition, John Wiley and Sons, 1996. ISBN 0-471-11709-9.
56. Standard Deviation – wikipedia article, 13 June 2011, available at `http://en.wikipedia.org/wiki/Standard_deviation`.