Algebraic Complexity Reduction and Weak Keys in GOST

Abstract. GOST 28147-89 is a well-known block cipher and the official encryption standard of the Russian Federation. It has large keys of 256 bits and at CHES 2010 it was shown that it has a lower implementation cost than any comparable cipher, which makes it a plausible alternative for AES-256 and triple DES. In 2010 GOST was submitted to ISO, to become an international standard.

Until 2010 it was believed that GOST should be secure given its quite large number of rounds. A turning point in the security of GOST is its "Reflection Property" from Indocrypt 2008, where it was shown to lead to a weak-key attack on GOST with time complexity of 2^{192} working however only for a proportion of 2^{-32} of weak keys. Then in 2011 it was suddenly discovered that GOST can be broken in great many different ways, and more than half of these new attacks also exploit this "Reflection Property". Great many of these recent attacks fall into a certain general framework of "Complexity Reduction", which is a sort of a "umbrella" paradigm introduced in 2010 and which allows to reduce the problem of attacking full 32-round GOST, to the problem of attacking the same cipher with less rounds, at the cost of well-chosen assumptions. The best currently known attacks in this category are two distinct methods with a complexity of 2^{216} GOST computations which appear in [11] An alternative second step for one of these attacks was recently proposed by Shamir *et al.* in [16] improving time complexity down to 2^{192}.

In this paper we show that in the family of attacks with "Complexity Reduction" there are many even faster attacks, this is for several interesting classes of weak keys. The main result of this paper is to show several new attacks on GOST with time complexity going down to 2^{120} for a still reasonable and quite large proportion of keys going down to 2^{-64}, see Fig. 2.

This result goes far beyond the topic of weak keys, and we show that this attack can in fact be transformed in a "regular" attack with **overall total** running time of only 2^{185} which becomes the fastest attack ever found on GOST in the scenario where GOST is used in encryption with keys chosen at random. In spite of its large data requirements it is arguably better than any previously discovered attack including the most recent 2^{192} result of [16].

Key Words: Block ciphers, Feistel schemes, key scheduling, self-similarity, fixed points, reflection attacks, advanced slide attacks, algebraic attacks.

1 The GOST Encryption Standard

The Russian encryption standard GOST 28147-89 is an important government standard [22]. Its large key size of 256 bits make GOST a plausible alternative for AES-256 and 3-key triple DES. Clearly GOST is a very serious military-grade cipher designed with most serious applications in mind. At least two sets of GOST S-boxes have been explicitly identified as being used by the most prominent Russian banks, cf. [34, 23].

GOST is a block cipher with a simple Feistel structure, 64-bit block size, 256-bit keys and 32 rounds. Each round contains a key addition modulo 2^{32}, a set of 8 bijective S-boxes on 4 bits, and a simple rotation by 11 positions. The most complete current reference implementation of GOST in the OpenSSL library contains eight standard sets of S-boxes [23]. The attacks we consider in this paper, as well as other recent attacks on GOST from [26, 12, 11, 14], work with a very similar complexity whatever are the S-boxes used in GOST, assuming however they are known.

1.1 GOST And ISO Standardisation

The cost of cryptography is still a huge problem for the industry, for example only in 2010 Intel implemented an encryption algorithm in some of its CPUs, and not yet in all of its CPUs. It is therefore very important to notice that in addition to the very long bit keys GOST has a much lower implementation cost than AES or any other comparable encryption algorithm. For example in hardware GOST 256 bits requires less than 800 GE, while AES-128 requires 3100 GE, see [29]. Thus it is not surprising that GOST became an Internet standard, it is part of many crypto libraries such as OpenSSL and Crypto++ [23, 36], and is increasingly popular also outside its country of origin [29]. Hard to think about a better algorithm for the industry with its ultra-low implementation cost and 20 years of cryptanalysis efforts behind it. In 2010 GOST was submitted to ISO 18033 to become a worldwide encryption standard. Only seven block ciphers have ever become an ISO standard. Then in 2011 single key attacks on GOST have been found, see [26, 12, 11, 14].

2 Cryptanalysis of GOST

Large keys and a large number of 32 rounds make that GOST seems a plausible encryption algorithm to be used for many decades to come. It is clear that the structure of GOST is in itself quite weak, for example compared to DES, and in particular the diffusion is not quite as good, it was however always stipulated that this should be compensated by a large number of 32 rounds cf. [21, 34, 33, 2] and also by the additional non-linearity and diffusion provided by modular additions [21, 30]. In [2], Biryukov and Wagner wrote: "A huge number of rounds (32) and a well studied Feistel construction combined with Shannon's substitution-permutation sequence provide a solid basis for GOST's security." Until 2011, no cryptographically significant attack on GOST was found, at least if we limit to attacks on GOST used in encryption with random keys, which was summarized

in 2010 in these words: "despite considerable cryptanalytic efforts spent in the past 20 years, GOST is still not broken", see [29].

2.1 Linear and Differential Cryptanalysis of GOST

A basic assessment of the security of GOST against linear and differential cryptanalysis has been conducted in 2000 by Gabidulin *et al*, see [21]. The results are quite impressive: at the prescribed security of level of 2^{256}, 5 rounds are sufficient to protect GOST against linear cryptanalysis. Differential cryptanalysis of GOST seems comparatively easier and have attracted more attention. In [21] the authors also estimate that, 7 rounds should be sufficient to protect GOST against differential cryptanalysis. Moreover, two Japanese researchers [33], show that the straightforward classical differential attack with one single differential characteristic is unlikely to work **at all** for a large number of rounds. For full 32-round GOST such an attack with a single characteristic would work only for a negligible fraction of keys, see [33]. In the same paper [33], more advanced differential attacks on GOST are described. The best of these attacks allows to break about 13 rounds of GOST. Finally in 2011 better attacks of this type have been found, allowing finally to break full 32 round GOST slightly faster than by brute force [14].

2.2 Sliding and Reflection Attacks

According to Biryukov and Wagner, the structure of GOST, and in particular the reversed order of keys in the last 8 rounds, makes it secure against sliding attacks [1, 2]. However this exact inversion of keys allows other attacks in which fixed points are combined with a so called "Reflection" property [27]. This attack breaks GOST only for certain keys, which are weak keys. For these keys it is possible to break GOST with a complexity of 2^{192} and with 2^{32} chosen plaintexts [27].

2.3 Recent Developments in Reflection Attacks on GOST

A new Reflection-MITM which finally breaks GOST was presented at FSE 2011, see [26, 11]. This attack requires a lot of memory which makes it arguably worse even than some slower attacks with less memory. Many new attacks which also use reflections and even multiple reflections, which work for most GOST keys, and which allow to break full-round GOST with 256-bit keys, faster than brute force, have been recently found, see [11]. Most of these attacks require much less memory than in [26], and some are faster, see [11]. All these attacks (as well as in present paper) are modular and rely on basic facts which are subject to further improvement, which improvement in turn will further several results of these attacks. For example an alternative second step for one of these attacks was recently proposed by Shamir *et al.* in [16] improving the time complexity down to 2^{192}.

2.4 Attacks Without Reflections

One example of another novel attack which does **not** use any reflections, and in which no symmetric values whatsoever appear, and yet it allows to break GOST faster than brute force, with one single key, is given in [12]. Several other such attacks can be found in [11].

3 Some Vocabulary and Preliminary Remarks on GOST

In this paper we call **a P/C pair** a pair of known **P**laintext and **C**iphertext for full GOST, or for a reduced-round version of GOST.

GOST has 64-bit block size and the key size of 256-bit keys. Accordingly:

Fact 1. 4 P/C pairs are necessary to determine the GOST key. With 4 P/C pairs we expect to get on average about one key. We get the correct key together with a list of, sometimes a few, but on average less than 1 wrong keys.

With 5 P/C pairs we are able to discard all these wrong keys in practice: the probability that just one more key works for this extra P/C pair is 2^{-64}. This is unlikely to ever happen in a single key recovery attack.

Fact 2. A brute force attack on GOST takes 2^{255} GOST encryptions on average. *Justification:* We proceed as follows: we use one P/C pair and check all the possible keys. On average half way through the right key will be found. Only for an expected number of 2^{191} keys which are confirmed with the first pair, we do further comparisons. Most of these 2^{191} are **false positives**. This notion plays an important role in this paper. Here, and elsewhere, the key remark is that the total number of these false positives is small and the complexity of rejecting all the incorrect keys with additional P/C pairs is actually negligible. Indeed we have at most 2^{192} cases to be checked with another P/C pair. Then at most 2^{128} keys remain to be checked against the third P/C pair etc. Overall we need to do about $2^{255} + 2^{191} + 2^{127} + 2^{63} + 1$ GOST encryptions on average. This number is very close to 2^{255} GOST encryptions.

4 Algebraic Cryptanalysis and Low Data Complexity Attacks on Reduced-Round Block Ciphers

Algebraic attacks, can be defined as attacks in which the problem of key recovery is written as a problem of solving a large system of Boolean algebraic equations which follows the geometry and structure of a particular cryptographic circuit [35, 24, 3, 4, 6, 8, 19]. The problems are then solved by various type of algorithms and software such as Gröbner Bases [19, 18], ElimLin [10, 8] or various logical constraint satisfaction software [8, 31, 32]. For block ciphers these attacks typically work only for a limited number of rounds, for example to break 6 rounds of DES, but this is possible to do given only 1 known plaintext, see [8].

Application to GOST. GOST is a Feistel cipher with 32 rounds. In each round we have a round function $f_{k_i}(X)$ with a 32-bit sub-key k_i. Each round function contains a key addition modulo 2^{32}, a set of 8 bijective S-boxes on 4 bits, and a simple rotation by 11 positions. Our current best method for GOST is pretty much the same as the best known method for DES described in [8]. We recall one result from [11]:

Fact 3 (Key Recovery for 8 Rounds and 3 KP). Given 3 P/C pairs for 8 rounds of GOST we can produce 2^{64} candidates for the 256-bit key in time equivalent to 2^{120} GOST encryptions. The storage requirements are negligible and all the 2^{64} candidates can be produced in an uniform way, each of them is produced in time of 2^{56} GOST encryptions on average.

5 Algebraic Cryptanalysis with Complexity Reduction

Following [12, 11] the work of cryptanalyst who wants to cryptanalyse GOST can be split into two independent tasks. First task is how to achieve a software algebraic attack on a reduced-round version as discussed in the previous section. The second question is if and **how** the complexity major variants of full GOST with 32 rounds can ever be **reduced** to a problem of breaking a cipher with much less rounds. Only then we can ever hope to be able to apply results such as Fact 3. In fact **only in the recent 5 years** it became possible to design and implement an appropriate last step (cf. Fact 3) for many such attacks.

5.1 Reductions and Black-Box Reductions

The main idea is as follows [12, 11]: In order to reduce the attack complexity, we exploit the self-similarity of the cipher (due for example to a weak key schedule) and add some well-chosen assumptions which produce interesting and sometimes quite non-trivial consequences due to the high-level structural properties of the cipher, which makes cryptanalysis problems smaller, simpler and easier to solve. In this process we need to minimise the costs (in terms of probability that our assumptions hold) and to maximise the benefits (in terms of the number and the complexity of interesting relations which hold under these assumptions).

This process is called **Algebraic Complexity Reduction**, see [12, 11]. In most cases what we get is to compute (guess or determine) many internal values inside one or several decryptions, and literally break the cipher apart into smaller pieces. In particular we have **Black-Box Algebraic Complexity Reductions** where we obtain real black-box reductions, to for example the same cipher with strictly less rounds (and less data) again at the cost of some well-chosen assumptions. This creates new important **optimisation problems** in symmetric cryptanalysis: which deals with the fundamental question of how we can reduce the complexity of a cipher in cryptanalysis to a simpler problem, with a limited quantity of data, and with greatly reduced complexity, and this in the best possible (optimal) way while many interesting and non-trivial solutions will exist. One example of a Black-Box Algebraic Complexity Reduction from 2^{64} KP for 32 rounds of GOST, to 4 KP for 8 rounds of GOST, can be found in [12]. Reductions exploit self-similarity of different blocks and their inverses, fixed points in certain components, and reflections.

Reductions can be compared in terms of the number of pairs obtained, the resulting reduced number of rounds, success probability, and in terms of plaintext complexity, see [11, 12]. In this paper we focus on weak key attacks only and our goal is to achieve time complexities better than any attack of [11, 12], and as a matter of fact, better than any other known attack on GOST, for the family of weak keys of comparable size.

6 High-level Structure of GOST

GOST is a Feistel cipher with 32 rounds. In each round we have a round function $f_k(X)$ with a 32-bit key which uses a 32-bit segment of the original 256-bit key which is divided into eight 32-bit sub-keys $k = (k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7)$.

One 32-bit sub-key is used in each round, and their exact order is as follows:

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

Table 1. Key schedule in GOST

We write GOST as the following functional decomposition (to be read from right to left):

$$Enc_k = \mathcal{D} \circ \mathcal{S} \circ \mathcal{E} \circ \mathcal{E} \circ \mathcal{E} \tag{1}$$

Where $\mathcal{E}$ is exactly the first 8 rounds which exploits the whole 256-bit key, $\mathcal{S}$ is a swap function which exchanges the left and right hand sides and does not depend on the key, and $\mathcal{D}$ is the corresponding decryption function with $\mathcal{E} \circ \mathcal{D} = \mathcal{D} \circ \mathcal{E} = Id$.

This decomposition is the same as used at Indocrypt 2008 [27] and it is at the heart of several recent single-key attacks on full 32-round GOST [26, 11].

6.1 The Internal Reflection Property of GOST

We start with the following observation which is also used in weak attacks on GOST from [27] and to cryptanalyse the GOST hash function Crypto 2008 [25]. Both attacks also exploit fixed points, which only propagate for 32 rounds for certain keys and can only work for some special (weak) keys. In this paper (and also in [11, 26]) this property is exploited in a much more fundamental way: without any fixed points and for arbitrary keys.

Fact 4 (Internal Reflection Property). Consider the last 16 rounds of GOST $\mathcal{D} \circ \mathcal{S} \circ \mathcal{E}$ for one fixed GOST key. This function has an exceptionally large number of fixed points: applied to X gives the same value X with probability 2^{-32} over the choice of X, instead of 2^{-64} for a random permutation.

Justification: This comes from the fact that the state of the cipher after the first 8 rounds $\mathcal{E}$ is symmetric with probability 2^{-32}, and $\mathcal{D} \circ \mathcal{E} = Id$.

7 Weak Key Attacks on GOST

Weak keys offer a considerable degree of extra freedom to the attacker. One basic attack of this type has been published [27]. This attack breaks GOST for weak keys which occur with probability 2^{-32}. For these keys the attack allows to break GOST with a time complexity of 2^{192} and given 2^{32} chosen plaintexts.

Let d denote the density of keys for which a given attack works, defined as the probability that the attack will work for a key chosen uniformly at random. In attacks with algebraic complexity reduction of [11, 12], most attacks work for more than half of all keys, with typically $d \geq 0.63$ or better.

In this paper we will have d between 2^{-32} and 2^{-64}. Some weak-keys which are frequent enough to occur in the real life. For example, given that the population of our planet is about 2^{33}, and one person can use during their life many cryptographic keys, an attack with $d = 2^{-32}$ should be considered as nearly-realistic: it is plausible to assume that at some moment in the future 2^{32} different GOST keys will be used worldwide, making one of these keys vulnerable to one of our attacks or the the attack from [27].

Given the fact that even without weak keys, one can find a plethora of different attacks on GOST faster than brute force, see [11, 12], the reader can imagine that there exists also many interesting weak keys attacks on GOST. We are just going to exhibit some examples which we found interesting, either because they have a large d, or low complexity. We start by recalling the method of [27], this is our reference Family 0 of weak keys:

7.1 Weak Key Family 0

Fact 5 (Weak Keys Family 0, $d = 2^{-32}$, Reduction to 1 KP for 8R). We define the Weak Keys Family 0 by keys such that $\mathcal{E}$ has a fixed point A which is symmetric, i.e. $\overline{A} = A$. This occurs with density $d = 2^{-32}$.

For every key in Weak Keys Family 0, given 2^{32} chosen plaintexts for GOST, we can compute A and obtain 1 P/C pair for 8 rounds of GOST correct with very high probability of about 2^{-1}.

Justification: If A is a symmetric value such that $\mathcal{E}(A) = A$ then $Enc_k(A) = A$. However there are also, on average, about one values for which $Enc_k(A) = A$, as every permutation of 64 bits has about one fixed point which occurs by accident, not due to the internal structure. Thus we obtain 1 P/C pair for 8 rounds of GOST $\mathcal{E}(A) = A$, which is correct only with high probability of about 1/2.

7.2 Key Recovery With Family 0

Now in [27], this method of Fact 5 is used to recover keys with time complexity of 2^{192} and negligible memory. This is very hard to improve because the attack uses only 1 KP for 32 rounds, and there are 2^{192} keys for which this pair is correct, and all these keys must be checked against additional P/C pairs for the full 32-rounds. In the next section we will introduce another family of weak keys, with the same density d where we will be able at last to improve the time complexity of the attack.

7.3 Weak Key Family 1

Now we are going to exhibit another family of weak keys, with the same density but with a possibility to obtain more P/C pairs and improve the time complexity of the attack. We will also require 2^{64} KP instead of 2^{32} CP.

Fact 6 (Weak Keys Family 1, $d = 2^{-32}$, Reduction to 3 KP for 8R). We define the Weak Keys Family 1 by keys such that if A is a fixed point of Enc_k for the full 32 rounds, A is not symmetric (cf. Family 0) and $C = \mathcal{E}^2(A)$ is symmetric. This occurs with density of about $d = 2^{-32}$.

For every key in Weak Keys Family 1, given 2^{64} KP for GOST, we can compute A, B and obtain 1 P/C pairs for 8 rounds of GOST correct with probability of about 2^{-2}. Furthermore we can compute A, B, C and get 3 P/C pairs for 8 rounds correct with probability of about 2^{-34}.

Justification: On average we have one fixed point for the whole cipher, and the probability that is $C = \mathcal{E}^2(A)$ is symmetric is then $d = 2^{-32}$ taken over all possible GOST keys. For these keys we proceed as follows:

1. First we observe that since A is a fixed point and very few other fixed points exist for the permutation Enc_k, we can obtain A in time 2^{64} and our guess will be correct with very high probability of about 2^{-1}.
2. Then we observe that if B is defined as $B = \mathcal{E}(A)$ then we have:

   $$B = \mathcal{E}(A) = \mathcal{E}(Enc_k(A)) = \mathcal{E}(\mathcal{D}(\mathcal{S}(\mathcal{E}^3(A)))) = \mathcal{S}(\mathcal{E}^3(A)) = \mathcal{S}(\mathcal{E}^2(B))$$

   therefore we have

   $$\mathcal{E}^2(B) = \overline{B}.$$

   Then, as many times before, if $C = \mathcal{E}^2(A)$ is symmetric then in addition we have an internal reflection and we get that

   $$Enc_k(B) = \mathcal{E}^2(B) = \overline{B}.$$

   Thus not only we were able to guess obtained A to be correct with very high probability of about 2^{-1}, we can guess also B by searching the 2^{64} KP for points such that $Enc_k(B) = \overline{B}$ which points will be equal to our $B = \mathcal{E}(A)$ with very high probability of about 2^{-1}, as about one other fixed point on average will exist for the permutation $X \mapsto \overline{Enc_k(X)}$.
3. We get 1 pair for 8 rounds: $B = \mathcal{E}(A)$ correct with probability about 2^{-2}.
4. Now we also guess the symmetric value C, the guess will be correct with probability of about 2^{-32}.
5. Overall we guess A, B, C with probability of about 2^{-34}.
6. We get 3 pairs for 8 rounds: $B = \mathcal{E}(A)$, $\mathcal{E}(B) = C$ and $\mathcal{E}(C) = \overline{B}$.

Now we need to examine the consequences of this reduction with 2 P/C pairs.

Fact 7 (Key Recovery for Weak Keys Family 1, $d = 2^{-32}$).
One can recover the keys for the Weak Keys Family 1 with 2^{64} KP, running time of 2^{152} GOST encryptions and with negligible memory.

Justification: We use Fact 3 with the 3 P/C pairs obtained, and in total time equivalent to 2^{120} GOST encryptions we obtain 2^{64} candidates for the GOST key k. This needs to be multiplied by 2^{34} attempts to guess A, B, C.

In both cases the number of false positives is 2^{128} because we mount this attack with two pairs obtained from $Enc_k()$: one such that $Enc_k(A) = A$ and another such that $Enc_k(B) = \overline{B}$. The time to reject all the false positive keys with additional P/C pairs for the full 32-round GOST can be neglected in comparison to our 2^{154} GOST encryptions, which is the overall total time for this attack.

7.4 Weak Key Family 2

In this section we exhibit another family of weak keys, with the same density but with a possibility to obtain really much more P/C pairs and yet reduce in half of the cases we will present, the data complexity back to 2^{32} chosen ciphertexts. This attack is closely related to several attack described in [11]. They are both based on Reduction 3 from [11] where by requiring that B is also symmetric, (which happens only for weak keys but the probability of these keys is quite large of 2^{-32}) we will be able to simultaneously improve the probability of our guess being true, from 2^{-128} to 2^{-64} to obtain 4 P/C pairs for 8 rounds, and reduce the data complexity back to 2^{32} chosen ciphertexts.

Fact 8 (Weak Keys Family 2, $d = 2^{-32}$, Getting 3,4 and 5 KP for 8R). We define the Weak Keys Family 2 by keys such there exists A such that all the three values $\mathcal{E}(A)$, $\mathcal{E}^2(A)$ and $\mathcal{E}^3(A)$ are symmetric. This occurs with density $d = 2^{-32}$. For every key in Family 2, we have the following reductions:
-with 2^{32} CC we obtain 3 P/C pairs for 8 rounds of GOST correct with $P = 2^{-64}$,
-with 2^{64} KP we obtain 4 P/C pairs for 8 rounds of GOST correct with $P = 2^{-64}$,
-with 2^{32} CC we obtain 4 P/C pairs for 8 rounds of GOST correct with $P = 2^{-96}$,
-with 2^{64} KP we obtain 5 P/C pairs for 8 rounds of GOST correct with $P = 2^{-96}$.

Justification: See Fig. 1.

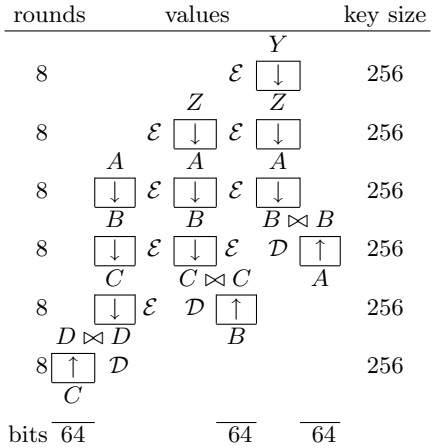

Fig. 1. A weak-key black-box reduction which gives up to 5 pairs for 8 rounds

We have three encryptions with internal reflection $C = Enc_k(A)$, also $B = Enc_k(Z)$, and $A = Enc_k(Y)$ where due to the internal reflection we have $C = \mathcal{E}^2(A)$, $B = \mathcal{E}^2(Z)$, and $A = \mathcal{E}^2(Y)$.

There are two interesting attack scenarios. In all cases we start by guessing C and B which are symmetric and therefore, to decrypt these and obtain respectively A and Z we need only 2^{32} CC. However if we also want to decrypt A to obtain Y, we need 2^{64} KP.

We proceed as follows:

1. We guess B, C and our guess is correct with probability 2^{-64}.
2. We determine A, Z by decrypting B and C which are both symmetric.
3. We get 3 pairs $\mathcal{E}(Z) = A$, $\mathcal{E}(A) = B$, $\mathcal{E}(B) = C$ and our guess is correct with probability 2^{-64}.
4. Furthermore, if we also decrypt A we get also one additional pair $\mathcal{E}(Y) = Z$, at the price of 2^{64} KP because A is not symmetric.
5. Going one step backwards, we don't decrypt A but also guess D which is symmetric, We get 4 pairs $\mathcal{E}(Z) = A$, $\mathcal{E}(A) = B$, $\mathcal{E}(B) = C$, $\mathcal{E}(C) = D$ and our guess is correct with probability 2^{-96}.
6. Now if we combine guessing D and decrypting A, we get 5 pairs given 2^{64} KP and our guess is correct with probability 2^{-96}.

Fact 9 (Key Recovery for Weak Keys Family 2, $d = 2^{-32}$).
One can recover the keys for the Weak Keys Family 2 with 2^{32} CC, running time of 2^{184} GOST encryptions and with negligible memory.

Justification: This is obtained by combination of the first reduction of Fact 8 and of Fact 3 which allows to enumerate a set of solutions and the time is 2^{64+120} GOST encryptions. The total number of full 256-bits keys which are false positives which need to be checked against additional P/C pairs for the full 32 rounds of the cipher is comparatively smaller, about 2^{128}, which is unlikely to influence the overall complexity of the attack which will be 2^{184} GOST encryptions.

7.5 Weak Key Family 3

In this section we explore if better attacks exist, and in particular attacks with complexity less than 2^{128}, at the price of further decreasing the density of weak keys to values which are no longer even remotely realistic.

Fact 10 (Weak Keys Family 3, $d = 2^{-64}$, Getting 4 KP for 8R). We define the Weak Keys Family 3 by keys such there exists A such that $\mathcal{E}(A) = \overline{A}$, $\mathcal{E}^2(\overline{A}) = A$. This occurs with density $d = 2^{-64}$. For every key in Family 3, we have the following: with 2^{64} KP we obtain 4 P/C pairs for 8 rounds of GOST, correct with probability of roughly about $P = 2^{-1}$.

Justification: We proceed as follows:

1. First we observe that A is a fixed point for $Enc_k(\cdot)$. Indeed

$$Enc_k(A) = \mathcal{D}(\mathcal{S}(\mathcal{E}^3(A))) = \mathcal{D}(\mathcal{S}(\mathcal{E}^2(\overline{A}))) = \mathcal{D}(\mathcal{S}(A)) = \mathcal{D}(\overline{A}) = A.$$

   Therefore given 2^{64} KP we can identify A. Due to other possible fixed points, our guess will be correct with probability roughly about $P = 2^{-1}$.
2. Moreover if we define $B = \mathcal{E}(\overline{A})$ we have $A = \mathcal{E}(B)$ and

$$Enc_k(\overline{A}) = \mathcal{D}(\mathcal{S}(\mathcal{E}^3(\overline{A}))) = \mathcal{D}(\mathcal{S}(\mathcal{E}(A))) = \mathcal{D}(\mathcal{S}(\overline{A})) = \mathcal{D}(A) = B.$$

   Therefore we can determine B from A.

3. Moreover, if we encrypt B we obtain another interesting value C defined as:

$$Enc_k(B) = \mathcal{D}(\mathcal{S}(\mathcal{E}^3(B))) = \mathcal{D}(\mathcal{S}(\mathcal{E}^2(A))) = \mathcal{D}(\mathcal{S}(\mathcal{E}(\overline{A}))) = \mathcal{D}(\mathcal{S}(B)) = \mathcal{D}(\overline{B}) = C.$$

   with the property that $\mathcal{E}(C) = \overline{B}$.
4. Overall our triple A, B, C will be correct with probability about $P = 2^{-1}$. We get 4 P/C pairs for 8 round which are $\mathcal{E}(A) = \overline{A}$, $\mathcal{E}(\overline{A}) = B$, $\mathcal{E}(B) = A$ $\mathcal{E}(C) = \overline{B}$ and these are correct with probability 2^{-1}.

Fact 11 (Key Recovery for Weak Keys Family 3, $d = 2^{-64}$).
One can recover the keys for the Weak Keys Family 3 with 2^{64} KP, running time of 2^{121} GOST encryptions and with negligible memory.

Justification: This is obtained by combination of the current reduction of Fact 10. This reduction manages to exploit the information obtained from as many as three different encryptions for $Enc_k()$ for $A, \overline{A}$ and B. It is therefore possible to see that in this attack, the total number of false positives which need to be checked against additional P/C pairs for the full 32 rounds is only 2^{64}.

7.6 Summary of Our Weak Key Attacks

In Table 2 we can compare our attacks with weak keys compared to selected other attacks with regular keys

It is important to note that with regular keys, we had no attack below 2^{192}, see [11]. One of the barriers to achieve really fast attacks are false positives. With attacks such all our black box reduction attacks, only if we manage to simultaneously exploit similarities inside two encryptions for 32 rounds and convert them to a number of simpler P/C pairs for 8 rounds we will have less than 2^{192} false positives. And this condition is necessary, but not sufficient. Finally for $d = 2^{-32}$ we have been able to obtain two attacks which are below 2^{192}, the lowest being 2^{154} with Family 1, however the Family 2 is also interesting because it has much lower data requirements. Our estimations of false positives given here are conservative in the case of weak keys.

Finally and furthermore, an interesting question now is whether attacks below 2^{128} can exist for some 256-bit GOST keys. We see that this can be achieved with Family 3 and for smaller number of only 2^{192} GOST keys, still quite large.

Reduction cf.	Red. 1 in [11]		Family 0	Family 1	Red. 3 in [11]	Family 2	Family 3
Key Size	256						
Keys Density d	0.63		2^{-32}		0.63	2^{-32}	2^{-64}
From (data 32 R)	2^{32} KP		2^{32} CP	2^{64} KP		2^{32} CC	2^{64} KP
Obtained (for 8R)	**2** KP		**1** KP	**3** KP	**3** KP	**3** KP	**4** KP
Valid w. prob.	2^{-96}		2^{-1}	2^{-34}	2^{-96}	2^{-64}	2^{-1}
Storage bytes	2^{132}	-	-	2^{67}		-	2^{67}
♯ False Positives	2^{192}			2^{128}			2^{64}
Attack Time 32 R	2^{224}	2^{248}	2^{192}	$\mathbf{2^{154}}$	2^{216}	$\mathbf{2^{184}}$	$\mathbf{2^{121}}$
Time / d	2^{225}	2^{249}	2^{223}	2^{186}	2^{217}	2^{216}	$\mathbf{2^{185}}$

Table 2. Our attacks with weak keys compared to selected regular attacks from [11]

Remark: In the last line we propose to compare all these attacks by comparing the value T/d where T is the time complexity. This can be seen as an expected complexity of an attack on a "sufficiently diverse" population of keys, to recover one of these keys. For example, in the last column we see that GOST has at least 2^{192} keys with security of only 2^{120}. This is quite good, because it means that if GOST is used with at least 2^{64} different keys, one can recover one of these keys in total time of not more than $T/d = 2^{185}$ (applying the same attack also in cases it does not yield a valid key, because the key is not weak, and aborting after 2^{120} GOST encryptions). This is substantially less than 2^{216} of [11] which is the best attack which works for most keys.

8 Conclusion

The Russian encryption standard GOST is implemented in OpenSSL and other crypto libraries [23, 36], used by Russian banks, and increasingly also on the Internet. It appears that GOST has a substantially lower gate count than any comparable cipher, cf. [29]. In 2010 GOST was submitted to ISO to become an international standard.

The strange idea of Algebraic Cryptanalysis (a.k.a. attacks "Formal Coding") has been around for more than 60 years [35, 24]. Yet only in the last 10 years several efficient software tools for solving various NP-hard problems involved have been developed, while numerous specific vulnerabilities leading to efficient attacks of this type have been found. A number of stream ciphers are indeed broken [6, 4, 5]. However only some weak block ciphers such as KeeLoq and GOST can be broken by such attacks [9, 11, 12] and this is mostly due to high-level structural properties due to a weak key schedule. Many of these attacks follow the general idea of "Algebraic Complexity Reduction" where with well-chosen assumptions one greatly reduces the complexity of the cipher as a circuit and makes algebraic cryptanalysis suddenly feasible. The best attack of this type on GOST has a complexity of 2^{216}, see [11].

In this paper we show that this methodology allows some even faster attacks for several interesting classes of weak keys. Our new results are summarized on Fig. 2. We demonstrated the existence of attacks on GOST below 2^{192} for a large proportion of 2^{-32} of all keys, and furthermore we showed that there are attacks on GOST below 2^{128} for a still reasonable and quite large proportion of keys, going down to 2^{-64}. Overall we demonstrated that the security of GOST degrades quite quickly in the multiple-key scenario, because some keys will be weaker and can be recovered faster than expected.

Importantly, we achieve way more than a weak key attack. Our best attack in the last column of Fig. 2 can in fact be transformed into a "regular" attack on a diverse population of at least 2^{64} different keys, with 2^{64} KP per key. Then one can recover one of these keys in total time of not more than $T/d = 2^{185}$ which is less than 2^{216} of [11] and the 2^{192} result of [16] which are the best currently known attacks which work for most keys. If we assume diverse keys and are interested in breaking only some weaker keys, we have established a new absolute record in cryptanalysis of GOST: an attack which requires an overall total time of 2^{185} GOST encryptions. Though it is not exactly a single-key attack, all these attacks are very much comparable in terms of running time data requirements and practical significance. Arguably we have contributed the fastest algorithmic shortcut attack on GOST found to this day in the scenario where GOST is used in encryption with keys generated at random. It remains an academic attack which disqualifies GOST as a credible international ISO standardization candidate but does not threaten its practical applications in the near future.

References

1. A. Biryukov, D.Wagner: *Slide Attacks,* In proceedings of FSE'99, LNCS 1636, pp. 245-259, Springer, 1999.
2. Alex Biryukov, David Wagner: *Advanced Slide Attacks,* In Eurocrypt 2000, LNCS 1807, pp. 589-606, Springer 2000.
3. Nicolas Courtois and Josef Pieprzyk: *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations,* Asiacrypt 2002, LNCS 2501, pp.267-287, Springer.
4. Nicolas Courtois and Willi Meier: *Algebraic Attacks on Stream Ciphers with Linear Feedback,* Eurocrypt 2003, LNCS 2656, pp. 345-359, Springer. An extended version is available at `http://www.minrank.org/toyolili.pdf`
5. Nicolas Courtois: *Fast Algebraic Attacks on Stream Ciphers with Linear Feedback,* Crypto 2003, LNCS 2729, pp: 177-194, Springer.
6. Nicolas Courtois: *General Principles of Algebraic Attacks and New Design Criteria for Components of Symmetric Ciphers,* in AES 4, LNCS 3373, pp. 67-83, Springer, 2005.
7. Gregory V. Bard, Nicolas T. Courtois and Chris Jefferson: *Efficient Methods for Conversion and Solution of Sparse Systems of Low-Degree Multivariate Polynomials over GF(2) via SAT-Solvers,* `http://eprint.iacr.org/2007/024/`.
8. 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/`.
9. Nicolas Courtois, Gregory V. Bard, David Wagner: *Algebraic and Slide Attacks on KeeLoq,* In FSE 2008, pp. 97-115, LNCS 5086, Springer, 2008.
10. Nicolas Courtois and Blandine Debraize: *Algebraic Description and Simultaneous Linear Approximations of Addition in Snow 2.0.,* In ICICS 2008, 10th International Conference on Information and Communications Security, 20 - 22 October, 2008, Birmingham, UK. In LNCS 5308, pp. 328-344, Springer, 2008.
11. Nicolas Courtois: *Algebraic Complexity Reduction and Cryptanalysis of GOST,* Preprint available at `http://www.nicolascourtois.com/papers/gostac11.pdf`.
12. Nicolas Courtois: *Security Evaluation of GOST 28147-89 In View Of International Standardisation,* will appear in Cryptologia in early 2012. An earlier version which was officially submitted to ISO in May 2011 and can be found at `http://eprint.iacr.org/2011/211/`.
13. Nicolas Courtois, Michał Misztal: *First Differential Attack On Full 32-Round GOST,* in ICICS'11, pp. 216-227, Springer LNCS 7043, 2011.
14. Nicolas Courtois, Michał Misztal: *Differential Cryptanalysis of GOST,* In Cryptology ePrint Archive, Report 2011/312. 14 June 2011, `http://eprint.iacr.org/2011/312`.
15. Nicolas Courtois, Michał Misztal: *Aggregated Differentials and Cryptanalysis of PP-1 and GOST,* In 11th Central European Conference on Cryptology, post-proceedings in preparation.
16. Itai Dinur, Orr Dunkelman and Adi Shamir: *Improved Attacks on Full GOST, 11 October 2011,* At `http://eprint.iacr.org/2011/558/`.
17. Charles Bouilleguet, Patrick Derbez, Orr Dunkelman, Nathan Keller, Pierre-Alain Fouque: *Low Data Complexity Attacks on AES*, Cryptology ePrint Archive, Report 2010/633. `http://eprint.iacr.org/2010/633/`.
18. Jean-Charles Faugère: *A new efficient algorithm for computing Gröbner bases without reduction to zero (F5),* Workshop on Applications of Commutative Algebra, Catania, Italy, 3-6 April 2002, ACM Press.

19. Gwenolé Ars, Jean-Charles Faugère: *An Algebraic Cryptanalysis of Nonlinear Filter Generators using Gröbner Bases* INRIA research report, available at https://hal.ccsd.cnrs.fr/.
20. E. K. Grossman, B. Tuckerman: *Analysis of a Weakened Feistel-like Cipher,* 1978 International Conference on Communications, pp.46.3.1-46.3.5, Alger Press Limited, 1978.
21. 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
22. 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. In Russian, translated to English in `http://www.autochthonous.org/crypto/gosthash.tar.gz`
23. 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/`
24. J. Hulsbosch: *Analyse van de zwakheden van het DES-algoritme door middel van formele codering,* Master thesis, K. U. Leuven, Belgium, 1982.
25. Florian Mendel, Norbert Pramstaller, Christian Rechberger, Marcin Kontak and Janusz Szmidt: *Cryptanalysis of the GOST Hash Function,* In Crypto 2008, LNCS 5157, pp. 162 - 178, Springer, 2008.
26. Takanori Isobe: *A Single-Key Attack on the Full GOST Block Cipher,* In FSE 2011, pp. 290-305, Springer LNCS 6733, 2011.
27. Orhun Kara: *Reflection Cryptanalysis of Some Ciphers,* In Indocrypt 2008, LNCS 5365, pp. 294-307, 2008.
28. John Kelsey, Bruce Schneier, David Wagner: *Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and triple-DES,* In Crypto'96, pp. 237-251, LNCS 1109, Springer, 1996.
29. 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.
30. C. Charnes, L. O'Connor, J. Pieprzyk, R. Savafi-Naini, Y. Zheng: *Comments on Soviet encryption algorithm,* In Advances in Cryptology - Eurocrypt'94 Proceedings, LNCS 950, A. De Santis, ed., pp. 433-438, Springer, 1995.
31. Igor Semaev: *Sparse Algebraic Equations over Finite Fields,* SIAM J. Comput. 39(2): 388-409 (2009).
32. Haavard Raddum and Igor Semaev: New Technique for Solving Sparse Equation Systems, ECRYPT STVL website, January 16th 2006, available also at `eprint.iacr.org/2006/475/`
33. Haruki Seki and Toshinobu Kaneko: *Differential Cryptanalysis of Reduced Rounds of GOST. In SAC 2000, Selected Areas in Cryptography, Douglas R. Stinson and Stafford E. Tavares, editors, LNCS 2012, pp. 315323, Springer, 2000.*
34. *Bruce Schneier: Section 14.1 GOST,* in *Applied Cryptography,* Second Edition, John Wiley and Sons, 1996. ISBN 0-471-11709-9.
35. Claude Elwood Shannon: *Communication theory of secrecy systems,* Bell System Technical Journal 28 (1949), see in particular page 704.
36. Wei Dai: Crypto++, a public domain library containing a reference C++ implementation of GOST and test vectors, `http://www.cryptopp.com`