A Single-Key Attack on the Full GOST Block Cipher
Cryptography Final Report

Bhagirath - B19CSE021, Bharat - B19CSE022

Abstract— **The selected paper proposes a single key attack on GOST cipher. In this report we breifly summarise our discussion of interim report and then look at the limitations of the attack. Then we go through recent developments/improvements made to GOST cipher to evade the R-MITM attack and improve the version of R-MITM attack from the original paper. We then discuss other block ciphers on which the R-MITM is applicable and the one's that are immune. Finally we conclude by looking at other LWC algorithms which employ similar techniques to form an attack.**

Keywords— **GOST, Meet-in-the-middle attack, Reflection attack, Block cipher, Lightweight cryptography**

I. Introduction

The selected article introduces a single-key attack for the first time on GOST block cipher which works on all key classes. The proposed attack makes use of a new framework called the Reflection-Meet-in-the-Middle attack which leverages the meet-in-the-middle and reflection attack and recovers the key in 2^{225} computations with 2^{32} plaintexts.

GOST is a Feistel structure that is divided into 64-bit blocks. The round function is achieved by appending (modulo 2^{32}) a 32-bit round key to the right half of the block and then applying the function f shown in Figure 1. This function contains a Sbox layer consisting of eight separate 4 4 S-boxes, followed by a little-endian rotation of the 32-bit result by 11 bits to the left.

The entire GOST contains 32 rounds and an incredibly basic key schedule: the 256-bit key is broken into eight 32-bit words (K_1, K_2, ..., K_8). Each round of GOST employs one of these words as a round key in the following order: the first 24 rounds use the keys in their cyclic order (i.e. K_1 in rounds 1,9,17, K_2 in rounds 2,10,18, and so on).

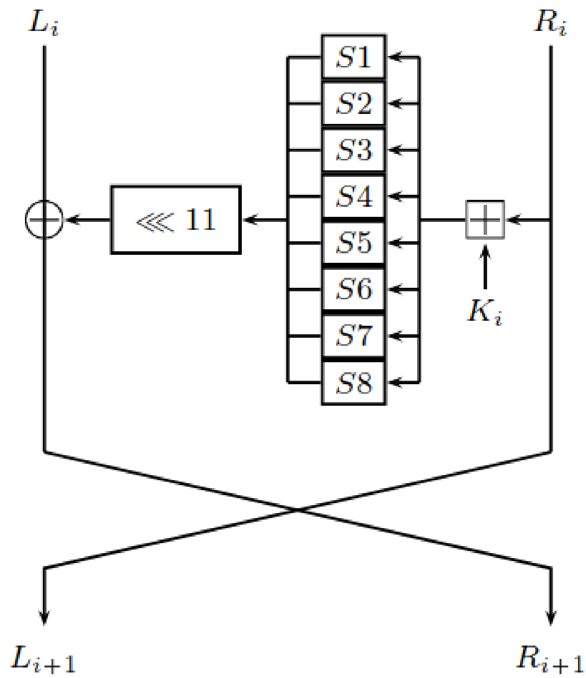

Fig. 1: A single round of GOST

A. DES vs GOST

A1 DES

DES stands for Data Encryption Standards, also known as Data Encryption Algorithm, a block cipher (works on a block of text) used to encrypt a block of 64-bit plain text with a 56-bit key to generate a block of 64-bit cipher-text. The DES algorithm is based on two cryptographic properties, substitution and transpositions, and is composed of 16 rounds, each of which conducts transpositions and substitutions. There are two variants available: double DES and triple DES. DES does not work on a bit-by-bit basis. As a result, it will not choose one bit and then process it. Instead, it computes or processes a whole 64-bit block of data.

A2 Comparison

The publicly accessible DES was purposefully designed with marginal settings (16 rounds, 56-bit keys), but the covert GOST utilized higher parameters (32 rounds, 256-bit keys), which appeared to give an extra margin of security. As a consequence, DES was theoretically cracked (using differential and linear approaches), but all single-key attacks disclosed before 2011 were only relevant to reduced-round versions of the cipher.

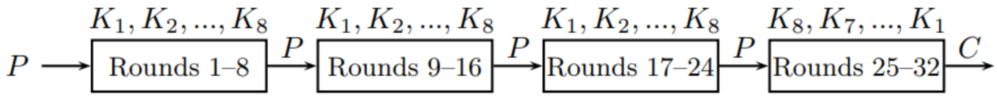

Fig. 2: Reflection property of GOST

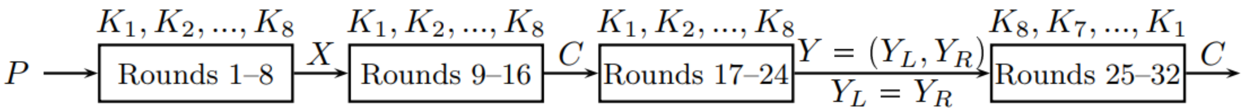

Fig. 3: Reflection property of GOST

II. Isobe's attack

The attack used by the original author[1] i.e. the reflection meet-in-the-middle attack, we have called it the Isobe's attack based on the author's name.It took the use of a remarkable reflection feature discovered by Kara. When the left and right halves of the state are equal after 24 rounds (which happens with probability 2_{-32}), the final 16 rounds form the identity mapping, reducing the effective number of rounds from 32 to 16. Isobe created a new key-extraction procedure for the last 16 rounds of GOST that took 2^{192} time and 2^{64} memory, and he used it 2^{32} times for various plain-text/cipher-text combinations to obtain the entire 256-bit key in a total of 2^{32} data, 2_{64} memory, and 2^{224} time. This is substantially quicker than an exhaustive search, but neither the time complexity nor the memory complexity is realistic.

A. Limitations of attack

Isobe's attack relies heavily on the assumption that S-boxes are invertible. Because the GOST standard does not specify the S-boxes and there is no requirement to make them invertible in a Feistel structure, Isobe's attack may not be relevant to some valid versions of this standard. A similar issue arises with most of Courtois' attacks, as their complexity is only assessed for one specific option of S-boxes detailed in which is utilized in the Russian banking system, and it is probable that the difficulties would alter other choices of S-boxes.

Only those who use bijective S-boxes are vulnerable to the attack. Isobe's attack is limited to a single input-output pair gained during the first 16 rounds of GOST (by guessing intermediate values obtained after 4 and 12 rounds) and so can be paired with the reflection property, but not the two input-output pairs produced by the fixed point property. The various intermediate encryption values obtained for all the potential values of big sets of keys are stored in a vast quantity of memory in MITM attacks.

III. Preliminaries

A. Fixed point property

This property says that the probability for a random plaintext to be a fixed point for any key is 2^{-64} and and hence we need to know around 2^{64} plain texts to get a single fixed point. And if we have c * 2^{64} plaintexts where c is a fraction then the probability that a fixed point is present in the plain text can be given as c. This implies that the success probability and time,data complexity get reduced by a factor of c. This revelation gives us reason to try the fixed point attack on GOST even when not much is known from GOST codebook.

B. Reflection property

Assume that after 24 rounds of GOST, the encryption of a plaintext P yields a 64-bit number Y, with the 32-bit right and left halves of Y equal (i.e. $Y_R = Y_L$). As a result, at the end of round 24, exchanging the two halves of Y has no effect on the intermediate encryption value. The round keys K_1–K_8 are applied in reverse order in rounds 25–32, while Y is subjected to the identical processes as in rounds 17–24, but in reverse order. As a result, after 32 rounds, the encryption of P yields the ciphertext C.

We get two 8-round input-output pairs (P, X) by predicting the state of P's encryption after eight rounds, which is denoted by the 64-bit number X. (X, C). The likelihood that a random plaintext will provide such asymmetric value Y after 24 rounds is 2^{-32} for an arbitrary key, implying that we will have to test around 2^{32} known plaintexts (in addition to guessing X) to get the two pairs. It's worth noting that the reflection property really offers us another "half pair" (C,Y), in which the 64-bit word C is produced from C by swapping its right and left 32-bit halves, and the 32-bit right and left halves of Y are equal.

C. Meet in the middle attack

Meet-in-the-middle (MITM) attacks are effective against block ciphers in which some intermediate encryption variables (bits or combinations of bits) are dependent only on a subset of key bits from the encryption side and another subset of key bits from the decryption side: the attacker guesses the relevant key bits from the encryption and decryption sides independently and only tries keys whose values suggested by the computed intermediate variables match. While the entire 32-round GOST can withstand such attacks, the 8-round GOST uses round keys that are completely independent. Thus, after four encryption rounds, the whole 64-bit value is determined solely by round keys K_1–K_4 on the encryption side and round keys K_5–K_8 on the decryption side.

Let's pretend we have two 8-round input-output pairs and a few more 32-round plaintext/ciphertext pairings. After four rounds of GOST:

1. Encrypt both inputs and acquire two 64-bit intermediate encryption values for each of the 2_{128} possible K_1–K_4 (i.e., 2^{128} intermediate values of 128 bits each).
2. Keep a list of the intermediate values sorted by these 128 bits and the associated K_1–K_4 value.
3. Decrypt both outputs, obtain two 64-bit intermediate values, and search the sorted list for these two values for each of the 2128 possible K_5–K_8 values.
4. Get the relevant value of K_1–K_4 from the sorted list for each match, then concatenate the value of K_1–K_4 with the value of K_5–K_8 from the previous step to get a full 256-bit key.

IV. Improving the Attack

As seen from the above section i.e. section 4 and also 2.A, the attack[1] is not the best version of itself and it can be significantly improved. We thought of a way to improve the attack by modifying its 3-subset meet-in-the-middle attack and reducing the memory complexity of it. Using the fixed point property and the property of reflection attack, we can reduce this from an attack of full 32 round GOST to an attack of 8 round GOST with two known input-output pairs. We are using the GOST's high degree of self-similarity. We can create an algorithm where during iteration over the possible plaintext-ciphertext pairs of 32 rounds, we get a set of values of two input-output pairs for $G_{k1,...k8}$. Apply an 8-round attack for every pair, this gives a corresponding 256-bit GOST key. We then verify key suggestions as we do in the Meet-in-the-middle attacks. This method is comparatively faster than brute force or exhaustive search.

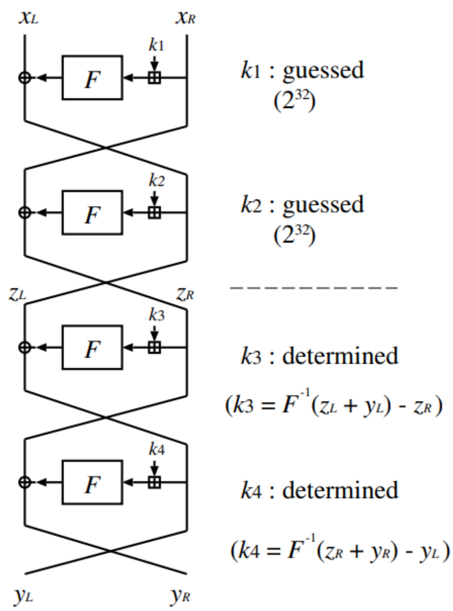

Fig. 4: Equivalent keys of 4 rounds of GOST

The 8-round attack is similar to the original MITM attack proposed by the original author[1]. If we have two 8-round I/O pairs which are (I, O) and (I*, O*). Let us take a word Y, it is what we got after encryption of I four times. So it is of 64 bits.

1. Apply the 4-round attack on (I,Y) to get 2^{64} possible values for K_1-K_4.
2. Encrypt I* partially giving Y* using the above values that we got and store Y* as $(Y_L{}^*, Y_R{}^*)$ with K_1 - K_4 as list L (The values of K_1-K_4 will be unique for every set of Y* we get)
3. Apply the 4-round attack on (Y,O) to get 2^{64} possible values for K_5-K_8.
4. Decrypt O* partially giving Y* using the above values that we got and store Y* as $(Y_L{}^*, Y_R{}^*)$.
5. Search the values we got in step 4 in L.
6. If we find the value in the list, test the full 256 bit keys.

On average, the time complexity will come out to be 2^{64}

Table 1: Comparison of various GOST attacks

Reference	Type of attack	Rounds	Data	Time Complexity	S-boxes
[2]	Slide	24	2^{63} ACP	2^{63}	Russian Banks
[2]	Slide	30	2^{63} ACP	2^{253}	Russian Banks
[3]	Reflection	30	2^{32} KP	2^{224}	Russian Banks
[2]	Slide (2^{128} weak keys)	32	2^{63} ACP	2^{63}	Russian Banks
[3]	Reflection (2^{224} Weak keys)	32	2^{32} CP	2^{192}	Russian Banks
[4]	Differential	21	2^{56} CP	Not given	Russian Banks
[5]	Differential	32	2^{35} CP	2^{244}	Russian Banks
[6]	Boomerang	32	$2^{7.5}$ CP	2^{248}	Russian Banks
[1]	Reflection MITM	32	2^{32} KP	2^{225}	Bijective
[7]	Unnamed	8	2^{64} CP	2^{248}	Russian Banks
[8]	Differential	32	2^{64} CP	2^{226}	Russian Banks
[9]	Reflection	32	2^{32} CP	2^{224}	any
[9]	Fixed point	32	2^{64} CP	2^{216}	Russian Banks
[10]	Fixed point MITM	32	2^{64} CP	2^{192}	any

Legend	
CP	Chosen plaintext
ACP	Adaptive chosen plaintext
KP	Known plaintext

for steps 1-4. The time complexity for steps 5-6 is 2^{64} as we try 2^{64} full keys (size of list L). The combined time complexity comes out to be $2^{64+64} = 2^{128}$ which is similar to Isobe's attack [1]. But, the memory complexity is only 2^{64} over here which was 2^{128} previously.

V. Other attacks on GOST

Several attacks on GOST have been published over the last 30 years. Seki and Kaneko [4] proposed a differential attack on 13-round GOST. As shown in [5], an attack can be improved up to 21 rounds in the related-key configuration. On the entire GOST, Fleischmann demonstrated a related-key boomerang attack that works for any S-box [6]. Biham demonstrated sliding attacks on the reduced GOST [2], just like other forms of attacks. Their approach relies on self-similarity among round functions in the encryption process and is unaffected by the S-box values employed. This technique can be used to attack the 24-round GOST even if the attacker does not know the values of S-boxes. This attack can be upgraded up to 30 rounds if the values are known. Furthermore, this technique can attack the entire GOST for a class of 2^{128} weak keys. Kara then offered a 30-round GOST [3] reflection attack. This attack also employs round function self-similarity and is applicable to all bijective S-boxes. The slide attack presented by Biham [2] differs in that it makes use of commonalities in both encryption and decryption operations. The reflection attack takes advantage of these similarities to create fixed points for specific round functions. Furthermore, the complete GOST can be attacked using the reflection technique for a class of 2^{224} weak keys.

There have been weak-key attacks and related key attacks on the full GOST cipher and some of them have been mentioned above but not all. The reason for this is that they cannot be applied to many keys. For example, the rate of weak keys in a weak key attack is 2^{-32} . Given there are 2^{256} keys, there are 2^{256-32} i.e., 2^{224} weak keys. Keys that cant is attacked would be 2^{256} -$2^{224} \simeq 2_{256}$. A related key attack is also not of importance in a real sense since it assumes access to the relation between encryption-decryption and the attacker. This is why single-key attacks are of the utmost value for security.

VI. Applicability of RMITM attack on various block ciphers

The attack works well on the GOST cipher but the GOST cipher also falls in the category of one of the ultra-lightweight block ciphers, which is one of the reasons for its weak response to the attack. The reason for this is that the authors of [11] wanted to create a lightweight version of the Soviet GOST cipher, they created a 651GE implementation. The S-box of this version decreases the GE metric [12]. However, the same cant be said about AES. The attack does not work on AES. One of the pre-requisites for the RMITM attack is that large parts of the cipher need to be independent of particular key bits. The reflection skip does not take place without this. This does not hold for AES which is why the number of rounds that can be broken with this technique is small. In a paper by Bogdanov et al. [13] , he mentions how this can be countered by using a concept called bicliques.

The results of the attack on ciphers having a simple key schedule like KTANTAN, IDEA, XTEA, and LED [14]have been better when compared to block ciphers consisting of a complex key schedule like Rijndael or Blowfish, no formidable results have been received so far[15]. The reason for this lies in the fundamentals of the attack. It requires at least two sets of keys of neutral bits. They are part of the secret key. It is easy to find them for a simple key schedule because the subkeys are derived from secret key bits. This is not as easy with a complex key schedule which is why it fails.

The KTANTAN block cipher is breakable using this attack with known plaintexts while putting fewer constraints on the selection of key bits [16]. The attack on XTEA, LED, Piccolo has also been studied by the author of the original paper. He concluded in the paper that this attack mainly exploits the low key-dependency , block cipher has large parts which are independent from key bits and a block cipher which has direct key expansion is more vulnerable to the attack. The three attacks mentioned above have simple key expanding functions. The ciphers were not fully attacked but they were partially broken with this attack, like XTEA 29 rounds out of 64, LED-128 16 rounds out of 48 and Piccolo-128 21 rounds out of 31.

VII. Application of techniques used in R-MITM in LWC algorithms

Many variations of the techniques used in the R-MITM attack i.e 3-subset MITM and reflection property have been used in other LWC to create attacks better suited to target block ciphers. Here we'll make a note of such attacks and the techniques that were used to create them.

GIFT is a fast and more secure version of the PRESENT cipher which is a ultra light weight block cipher. The 3 subset MITM technique is used to create an attack against GIFT-64 with a complexity of 2^{112} as discussed in [17]. Katan[18] , Twine[19], Keeloq are some more LWC's which are exploited by variations of MITM attacks. Another use application of reflection attack proposed by O Kara is used in reflection of cryptanaylsysis of PRINCE like ciphers[20].

After the single key attack was proposed changes were made to GOST to make it immune to reflection attack and thus 2-GOST was created. Still the proposed 2-GOST is affected by reflection attack but it in case of weak key.

VIII. Conclusion

We discussed various applications of R-MITM attack and ciphers where R-MITM is effective and ineffective/ We also saw LWC algorithms which used techniques of R-MITM attack. We then tried to improve the method by using properties of the 4 rounds of GOST to reduce the memory from 2^{128} to 2^{64}. The trick was to get the 4 round keys K_5-K_8 and check for them in the list created before using K_1-K_4. This is something that we call in cryptography as guess and determine attack.

We talked about the limitations of the attack, the recent developments in the domain surrounding it and a table (Table 1) to do a comparative analysis. We further looked at how the attack is more powerful on block ciphers having simple key schedule than the ones with complex key schedule.

References

1. Isobe . A Single-Key Attack on the Full GOST Block Cipher *Springer.* 2011;6733:290–305.
2. Biham E., Dunkelman O., , Keller N.. Improved Slide Attacks *Biryukov.* 2017:153-156.
3. Kara O.. Reflection Cryptanalysis of Some Ciphers *Springer.* 2008:294–307.
4. Seki H., Kaneko T.. Differential Cryptanalysis of Reduced Rounds of GOST *Springer.* 2000:315–323.
5. Ko Y., Hong S., Lee W., Lee S., Kang J-S.. Related Key Differential Attacks on 27 Rounds of XTEA and Full-Round GOST *Springer.* 2008:299–316.
6. Fleischmann E., Gorski M., H¨uehne J., Lucks. S.. Key Recovery Attack on full GOST *Springer.* 2009;6429:299–316.
7. Courtois . Security Evaluation of GOST in View of International Standardisation *Cryptology ePrint Archive.* 2011.
8. Courtois , Misztal . Differential Cryptanalysis of GOST *Cryptology ePrint Archive.* 2011.
9. Courtois . Algebraic Complexity Reduction and Cryptanalysis of GOST *Cryptology ePrint Archive.* 2011.
10. Dinur Itai, Dunkelman Orr, Shamir Adi. Improved Attacks on Full GOST *19th international conference on Fast Software Encryption.* 2012.

11. Poschmann A., Ling S., , Wang H.. 256 Bit Standardized Crypto for 650 GE GOST Revisited *Cryptographic Hardware and Embedded Systems.* 2010.
12. Hatzivasilis George, Fysarakis Konstantinos, Papaefstathiou Ioannis, Manifavas Charalampos. A Review of Lightweight Block Ciphers *Journal of Cryptographic Engineering.* 2018.
13. Bogdanov Andrey, Khovratovich Dmitry, Rechberger Christian. Biclique Cryptanalysis of the Full AES *ASIACYRPT - advances in cryptology.* 2011.
14. Isobe Takanori, Shibutani Kyoji. Security Analysis of the Lightweight Block Ciphers XTEA, LED and Piccolo *Springer.* 2012;7372:71-86.
15. Isobe Takanori, Shibutani Kyoji. All Subkeys Recovery Attack on Block Ciphers: Extending Meet-in-the-Middle Approach *Springer.* 2012:202–221.
16. Bogdanov Andrey, Rechberger Christian. A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN *Springer.* 2011;6544:229–240.
17. Sasaki , Yu . Integer Linear Programming for Three-Subset Meet-in-the-Middle *Springer.* 2018:227-243.
18. Cannière Christophe De, Dunkelman Orr, Knežević Miroslav. KATAN and KTANTAN — A Family of Small and Efficient Hardware-Oriented Block Ciphers *Springer.* 2009:272–288.
19. Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, Kobayashi , Eita . Twine A lightweight versatile block cipher 2011;2011.
20. Soleimany Hadi, Blondeau Céline, Yu Xiaoli, et al. Reflection cryptanalysis of PRINCE-like ciphers *Journal of Cryptology.* 2015;28:718–744.