A nonlinear invariant attack on T-310 with the original Boolean function

Nicolas T. Courtois

University College London, Gower Street, London, UK

Abstract. There are numerous results on non-linear invariant attacks on T-310. In all such attacks found so far, both the Boolean functions and the cipher wiring were contrived and chosen by the attacker. In this article we show how to construct an invariant attack with the original Boolean function which was used to encrypt government communications in the 1980s.

Keywords: Boolean functions · Feistel ciphers· T-310 · Generalized Linear Cryptanalysis · Polynomial invariants · Annihilators · k-normality · Algebraic Cryptanalysis

1 Introduction

In current work on non-linear cryptanalysis of block ciphers there were first attacks on really contrived block ciphers in [10, 2] and further ciphers which are very academic and not used in real-life applications. For example we have attacks on full SCREAM, iSCREAM, Midori64, [27] and many others. Can we break a real-life cipher? Another family of works approaches this question as follows: given the cipher wiring (in T-310 or in DES) under what conditions a certain polynomial invariant works? The key tool here is the so called Fundamental Equation (FE) in [6] and the best attacks of this type found so far are constructed by multiplying many well chosen polynomials together, cf. [19]. Some of these attacks have a very large success rate, so eventually if we make a slight modification to our real-life Boolean function, we eventually get an attack which works, cf. [17]. However until now, no researcher has yet succeeded to find a non-linear attack specifically designed to work with one single real-life Boolean function which he has not chosen. We simply need to adapt our whole non-linear attack to this specific Boolean function.

In the present article we show how to eventually achieve this. Here the cipher wiring will be special and specifically designed to make our attack work. This is particularly interesting because one cannot hope to change the Boolean function inside any real-life historical cipher machine in order to make an attack work. This is fixed and implemented in hardware. In contrast, it is possible and allowed to change the cipher wiring in T-310. The long term key in T-310 takes the form of a printed board, and was changed every few years [13]. In this paper we show that a super weak choice is possible.

This result is particularly significant for T-310, a government encryption system, the cost of which is thousands of times more complex and costly, than with modern commercial ciphers such as AES, cf. [15]. With our attack a large number of rounds does not help. We will construct an invariant property which propagates for any number of rounds.

This article is organised as follows. In Section 2 we describe our cipher. In Section 3 we study our Boolean function which is a central object of attention here. In Section 4.1 we study how our Boolean function can be annihilated by some polynomials. In Section 5 we describe our attack which comes with a mathematical proof. In Section 5.3. we give an example of a long-term key and explain possible variants of our result. We also explain how it can be re-interpreted in the light of a more general construction with many cycles cf. [19] and in Appendix A we discuss which features of our Boolean function would be relevant in further improved attacks.

2 T-310: a Complex Compressing Feistel Cipher with 4 Branches

We recall the definition of T-310 block cipher from [25]. T-310 operates on 36 bits blocks and the secret key on 240 bits.

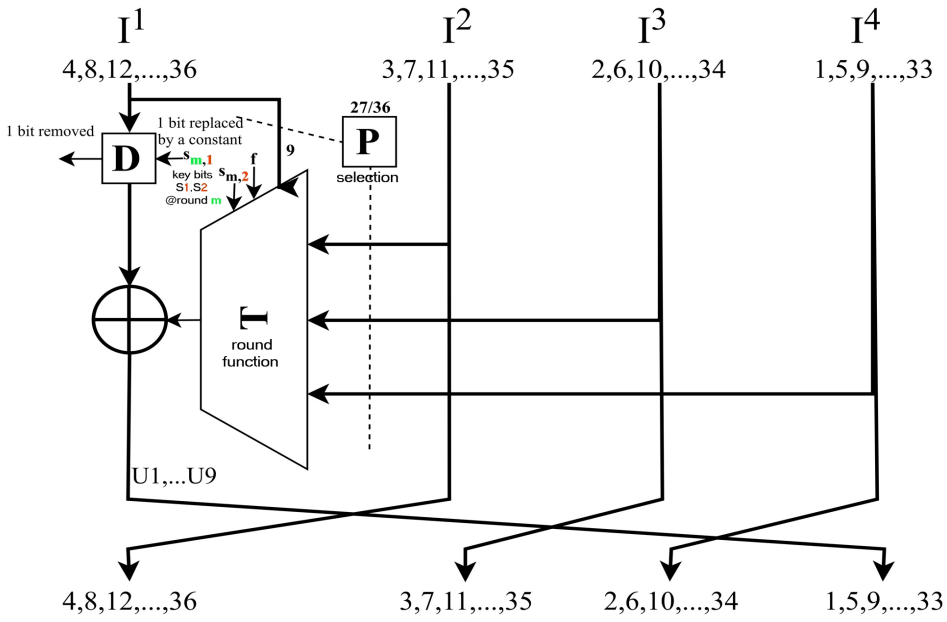

Fig. 1. High-level overview of one round of T-310

In each round only 2 key bits K, L are used and the same 2 bits are repeated after 120 rounds. The secret key is defined as $s_{1\ldots120,1\ldots2} \in \{0,1\}^{240}$ which is 240 bits. In addition each round has a round constant called F which derived from the public IV value. In all, for any $F, K, L \in GF(2)^3$ one round of this block cipher is a permutation[1]. on 36 bits. The wiring[2] of the cipher is defined by two functions: $D : \{1\ldots9\} \to \{0\ldots36\}$, $P : \{1\ldots27\} \to \{1\ldots36\}$. For example $D(5) = 36$ means that input bit 36 is connected to the wire called $D5$ in Fig. 4 which then becomes $U5 = y_{17}$ after XOR with bit $g4$. Then $P(1) = 25$ means that input 25 is connected as v1 or the 2nd input of Z_1 cf. Fig. 4.

[1] This requires some complex technical conditions on the cipher wiring [14]

[2] This wiring is also called LZS or *Langzeitschlüssel* which means a long-term key.

In one round 9 new bits are created and $36-9=27$ bits are shifted by one position. The cipher uses 4 identical Boolean functions of 6 bits which are denoted by Z_1, Z_2, Z_3, Z_4 and sometimes also by 1-letter notations $Z(), Y(), X(), W()$ respectively. Below we give a set of closed formulas to compute the output bits y_{1-36} in each round from the input bits x_{1-36}.

$$y_{i+1} = x_i \text{ for any } i \neq 4k \qquad (\text{ with } 1 \leq i \leq 36) \tag{r0}$$
$$y_{33} = F + x_{D(9)} \tag{r1}$$
$$Z_1 \stackrel{def}{=} Z(L, x_{P(1)}, \ldots, x_{P(5)}) \tag{z1}$$
$$y_{29} = F + Z_1 + x_{D(8)} \tag{r2}$$
$$y_{25} = F + Z_1 + x_{P(6)} + x_{D(7)} \tag{r3}$$
$$Z_2 \stackrel{def}{=} Y(x_{P(7)}, \ldots, x_{P(12)}) \tag{z2}$$
$$y_{21} = F + Z_1 + x_{P(6)} + Z_2 + \quad x_{D(6)} \tag{r4}$$
$$y_{17} = F + Z_1 + x_{P(6)} + Z_2 + \quad x_{P(13)} + x_{D(5)} \tag{r5}$$
$$Z_3 \stackrel{def}{=} X(x_{P(14)}, \ldots, x_{P(19)}) \tag{z3}$$
$$y_{13} = F + Z_1 + x_{P(6)} + Z_2 + \quad x_{P(13)} + L + Z_3 + x_{D(4)} \tag{r6}$$
$$y_9 = F + Z_1 + x_{P(6)} + Z_2 + \quad x_{P(13)} + L + Z_3 + x_{P(20)} + x_{D(3)} \tag{r7}$$
$$Z_4 \stackrel{def}{=} W(x_{P(21)}, \ldots, x_{P(26)}) \tag{z4}$$
$$y_5 = F + Z_1 + x_{P(6)} + Z_2 + \quad x_{P(13)} + L + Z_3 + x_{P(20)} + Z_4 + x_{D(2)} \tag{r8}$$
$$y_1 = F + Z_1 + x_{P(6)} + Z_2 + \quad x_{P(13)} + L + Z_3 + x_{P(20)} + Z_4 + x_{P(27)} + x_{D(1)} \tag{r9}$$
$$x_0 \stackrel{def}{=} K \tag{s1}$$
$$F \in \{0,1\} \text{ is a round constant} \quad \text{depending on a (public) IV} \tag{$f1$}$$
$$K = s_{m \bmod 120,\ 1} \quad (\text{in encryption round } m = 0,1,2,\ldots) \tag{$k1$}$$
$$L = s_{m \bmod 120,\ 2} \quad (\text{in encryption round } m = 0,1,2,\ldots) \tag{$k2$}$$

Fig. 2. The specification of one round of T-310

Notation. In order for our polynomials to be short and compact we sometimes replace the 36 bits $x_1 - x_{36}$ by single letters, cf. Fig. 3. We avoid certain letters like F used elsewhere. We study polynomial invariants for one round with 36 variables, and variables x_i and y_i are treated "alike" and can be called by the SAME letter, for example $x_{36} = a$ and then $y_{36} = a$ also. If we want to avoid ambiguity, we will distinguish between the variable a at input denoted by a^i or just a, and the same variable at output denoted by a^o or a^ϕ where ϕ is a short notation for one round of encryption.

Numbers	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
Letters	V	U	T	S	R	Q	P	O	N	M	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i	h	g	f	e	d	c	b	a

Fig. 3. Variable naming conventions

3 The Original Boolean Function

Our Boolean function which we denote in short by a single letter Z, is used 4 times inside one encryption round. It was first specified in [21] dated 1973, and it is the same as for an earlier SKS cipher in page 39 of [26], and the same as in page 113 in [24] and page 256 in [25], except that the constant 1 was omitted (by mistake). We have:

$$\begin{aligned} Z(a,b,c,d,e,f) = 1 \oplus a \oplus e \oplus f \oplus ad \oplus bc \oplus be \oplus de \oplus ef \oplus \\ acd \oplus acf \oplus ade \oplus bcf \oplus bdf \oplus cef \oplus \\ abcd \oplus abce \oplus abef \oplus bcdf \oplus abcde \oplus acdef \end{aligned}$$

3.1 Design Criteria for our Boolean Function Z from 1970s

We refer to Sections 3.3 to 3.6 in [16] for a detailed discussion of the original design criteria which were used when this Boolean function was chosen in the 1970s. We recall these criteria briefly here, with some rewriting and re-interpretation, using a more contemporary vocabulary. In [21] dated 1973, we read that:

$$(73.1) \qquad |\{X = (X_1, X_2, \ldots, X_6) \in \{0,1\}^6 | Z(X) = 0\}| = 2^5$$

$$(73.2) \qquad |\{X \in \{0,1\}^6 | Z(X) = 0, HW(X) = r\}| \approx \tbinom{6}{r} \cdot \tfrac{1}{2}, \qquad r = 0..6$$

$$(73.3) \quad |\{X \in \{0,1\}^6 | Z(X_1,..,X_i,..,X_6) = Z(X_1,..,X_i \oplus 1,..,X_6)\}| \approx 2^5, \quad i = 1..6$$

$$(73.4) \qquad Z \text{ is not symmetric}$$

A later document [1] from 1976 specifies another set of properties, which frankly looks less like requirements, and much more as an evaluation of a Boolean function already chosen at the time, probably[3] between 1973 and 1975.

(76.1) All derivations of Z were computed as Zhegalkin polynomials, which is the same as Algebraic Normal Form (ANF), and also was computed as truth tables.

(76.2) Frequency of the function result being 1 with k fixed inputs was computed for $(k = 1,2,3)$.

(76.3) The *statistic structure* which means the full table of linear characteristics of this Boolean function Z was computed. Original documents do not describe any full attack such as Linear Cryptanalysis, cf. [16, 28]. Instead they say that this was studied in order to see if some (presumably cryptanalytic) "advantage" could be gained from an approximation of our function Z by other Boolean functions.

(76.4) Z is not symmetric, for example when arguments are permuted, and arguments are negated, or a combination of both.

[3] Some earlier documents about these questions, about cipher designs called ALPHA, SKS and OPERATION cited in [1] are not available to us, cf. also [15].

3.2 Relevance to Our Attack and Modern Theory of Boolean Functions

These old criteria can be studied again in relation to the attack we present in this article and to the contemporary theory of Boolean functions. The main idea is that the output of our Boolean function Z is expected to behave randomly and should not remain constant after different types of transformations. In point (76.4) it is required that Z should not be constant when we permute or negate some inputs. The point (76.2) is more precise about the balanced-ness of Z when k inputs are fixed. In this article, cf. Section 4.1 below, we study whet happens when some 3 affine functions of the 6 variables are fixed, for example when $(a+d)(b+c)(f+e)=1$, and the attacker would like Z to be constant over such (affine) spaces. We get a more refined notion of balanced-ness or rather lack of it, on smaller affine spaces. This is more general, and closely related to the question of 3-normal Boolean functions, proposed and studied by Dobbertin and Charpin, cf. [20, 4]. A good Boolean function should not be constant on spaces of this type which are not too small (in cardinality). Then if we further replace the word "constant" by "affine", we obtain the question of 3-weakly-normal Boolean functions first introduced by Charpin in [4].

3.3 What Is Wrong with the Original Boolean Function

As far as we can see there is nothing wrong with this Boolean function. We could of course mandate stricter requirements such as in DES, cf. [11], however in our opinion these requirements for DES, are too strict, and many of them are not even necessary for designing a secure cipher.

3.4 Is the Original Boolean Function Secure?

In our opinion most Boolean functions, are equivalent in terms of security which can be obtained, for any given block cipher. For example for some 90 % of Boolean functions chosen completely at random, we expect similar attacks to exist. Weak cases do certainly exist, but in our opinion no Boolean function or S-box is really particularly strong, see for example [8] for stream ciphers or [18] for a block cipher. Several papers show the importance od cipher wiring, cf. [3] and [9] for DES, and [18] for GOST, etc.

This suggests that an opportunistic approach to cryptanalysis will work: design many different attacks which work with a large probability for a random Boolean function, and eventually one of them will break T-310 in a real-life setting. The problem however is that this approach is expensive and obscure and tedious: it requires to generate vast databases of possible attacks, for example following the general framework of [19], and we have not implemented it yet, and it might require some substantial computing power (and storage). Moreover we would obtain plenty of attacks but little understanding, of why some attacks exist and how to construct more attacks.

Therefore it is useful to try to work from the prescribed Boolean function and see what makes that a certain attack might work. We also hope that this approach is more illuminating. This is why we also wish to construct an attack by formal polynomial algebra, and by paper and pencil. This rather than working with some (more obscure) sets of constraints or with partitioning of some spaces or with complex formal algebra (or logic) software tools.

4 On the Annihilation Complexity of Z

One of the main vulnerabilities of Boolean functions, which lead to attacks on block and stream ciphers alike, cf. [12, 6] are annihilation vulnerabilities. Here also we have the 90 % question as above. Numerous attacks based on annihilation specifically, on block and stream ciphers alike were shown to work for a large proportion of Boolean functions or S-boxes, cf. [17] or for all Boolean functions [12, 8]. We need therefore to see what kind of annihilation events exist with our Boolean function, and in general.

4.1 On Algebraic Immunity and Annihilation Complexity of Z

The space of annihilators for Z has dimension 32, cf. Thm. C.2. in Appendix C of [17]. Inside there exist 10 linearly independent annihilators of degree 3 which are listed in [13]. Accordingly the so called Algebraic Immunity is 3. One of these 10 is particularly simple and is $ac+bc+ace+bce$, which can be factored as:

$$(a+b)c(1+e)Z = 0$$

Is this unusual? Yes and no, we know that (cf. Thm 6.1. in [17] based on earlier Thm 6.0.1. in [12]) for any Boolean function either Z or $Z+1$ has at least one annihilator of degree 3. A stronger result is that either Z or $Z+1$ will have an annihilator of degree being a product of 3 affine factors, which is in fact a direct consequence of a normality property reported in [4], based on earlier work of [22]. Therefore we are definitely not surprised that equations such as $(a+b)c(1+e)Z = 0$ exist.

4.2 Additional Annihilation Properties of Z

What is surprising, and we are not aware of any theoretical estimation which would confirm that, is that multiple solutions exist for **both** Z and $Z+1$ when Z is a balanced function. This is indirectly what makes the present work possible: if one annihilation property would not work for specific technical reasons, another might eventually work. The current construction in this article is such that it requires three disjoint sets of 2 variables. However we expect that there exist many other non-linear invariant attacks [19]. There exist vast quantities of annihilation properties for this (and any other) Boolean function. For example we have observed that:

$$(Z+e)(c+e)(d+e)(a+b) = 0$$

and

$$(Z+b)adf = 0$$

We have however paid particular attention to properties which would look like the property required in [17] which was $Z(a+b)(c+d)(e+f) = 0$ modulo a re-ordering a variables. We found for example that:

$$(Z)(f+d+1)(a+c)(b+d) = 0.$$

Finally we tried to relax our requirement of annihilating Z or $Z+1$ itself, as there are too few ways to annihilate Z. For this reason we allow the addition of arbitrary affine functions to Z, which corresponds to our more general attack in framework in [19], which works with a more general notion of weak normality for Boolean functions in [4]. We then found that we also have:

$$(Z+f+c)(a+d)(b+c)(f+e)=0$$

This equation (and only this one) is exploited in the present paper. We believe that a similar attack could be designed for many other ways to annihilate Z. Many other ways to annihilate Z exist and we expect that some are more suitable for being exploited inside an attack. We should also note that it is not exactly the so called Algebraic Immunity, but rather the more appropriate notions of normality and weak normality of [4], which are relevant here. Some important additional ways to annihilate Z specifically chosen for further applications in cryptanalytic attacks in mind are studied in Appendix A and in [13].

5 Constructing A Non-Linear Invariant Attack On T-310

In this paper we present one particularly strong attack on T-310 which is inspired by a similar attack described in [17]. However the objectives in the present article are very different than previously and we will need a number of additional technical constraints on the cipher wiring. We define the following 8 basic polynomials:

$$\begin{cases} A \stackrel{def}{=} (q+e) & \text{which is bits } 20,32 \\ B \stackrel{def}{=} (r+f) & \text{which is bits } 19,31 \\ C \stackrel{def}{=} (s+g) & \text{which is bits } 18,30 \\ D \stackrel{def}{=} (t+h) & \text{which is bits } 17,29 \\ E \stackrel{def}{=} (S+u) & \text{which is bits } 4,16 \\ F \stackrel{def}{=} (T+v) & \text{which is bits } 3,15 \\ G \stackrel{def}{=} (U+w) & \text{which is bits } 2,14 \\ H \stackrel{def}{=} (V+x) & \text{which is bits } 1,13 \end{cases}$$

and we observe that we have a pseudo-cycle, also shown in Fig. 5:

$$H \to G \to F \to E \to ?\ D \to C \to B \to A \to ?\ H$$

where $H \to G$ is a trivial transition, and we also write $G = H^{\phi}$, which is due to the internal wiring: these bits are just shifted inside this cipher, cf. rule (r0) in Fig. 2. Two transitions however, are problematic and marked with question marks. These are not quite correct and they are in fact simply rather **impossible** to achieve. They would be true if certain complex Boolean functions namely $W+c+f$ and $Y+c+f$ were equal to zero for every input. This is not the case, however some multiples of these functions will be zero. Here $W+c+f$ denotes a Boolean function shifted by the addition of 3rd and 6th variable to the output.

5.1 An Invariant Attack with the Original Boolean Function

We now present our main result. For simplicity it fixes the order of bits for the two Boolean functions and used colour coding and one example mapping of inputs for better readability. More generally inputs our two Boolean functions W and Y could be permuted if this is done in a consistent way for both. An example of a full cipher wiring which works will be given in page 11.

Theorem 5.2 (Invariant Attack using the Original Boolean Function).

Given the eight polynomials $A-H$ defined as above and reproduced also in Fig. 5, AND for each cipher wiring for T-310 s.t.

$$\begin{cases} \{D(1),D(4),P(20),P(27)\} = \{5\cdot 4, 8\cdot 4, P(23), P(26)\} \\ \{D(5),D(8),P(6),P(13)\} = \{1\cdot 4, 4\cdot 4, P(9), P(12)\} \end{cases}$$

AND if the Boolean function (used twice as W and as Y for different sets of inputs) is such that we have:

$$(Z+f+c)(a+d)(b+c)(f+e) = 0$$

AND for any mapping of any 3 out of 6 polynomials B,C,D,F,G,H into 6 inputs of W defined by integers $P(21),\ldots,P(26)$ connected[4] in a way which preserves[5] the partitioning in three sets or pairs in $(a+d)(b+c)(f+e)$, for example:

$$17,30,18,29,31,19$$

AND for any mapping of the remaining 3 pairs of inputs to[6] the 6 inputs of Y defined by integers $P(7),\ldots,P(12)$ of $Y()$, which also are connected preserving the three sets of pairs, for example

$$13,2,14,1,3,15$$

THEN for any short term key of 240 bits, and for any initial state on 36 bits, we have the non-linear invariant

$$\mathscr{P} = ABCDEFGH$$

holding with probability 1.0 for any number of rounds and for any F,K,L.

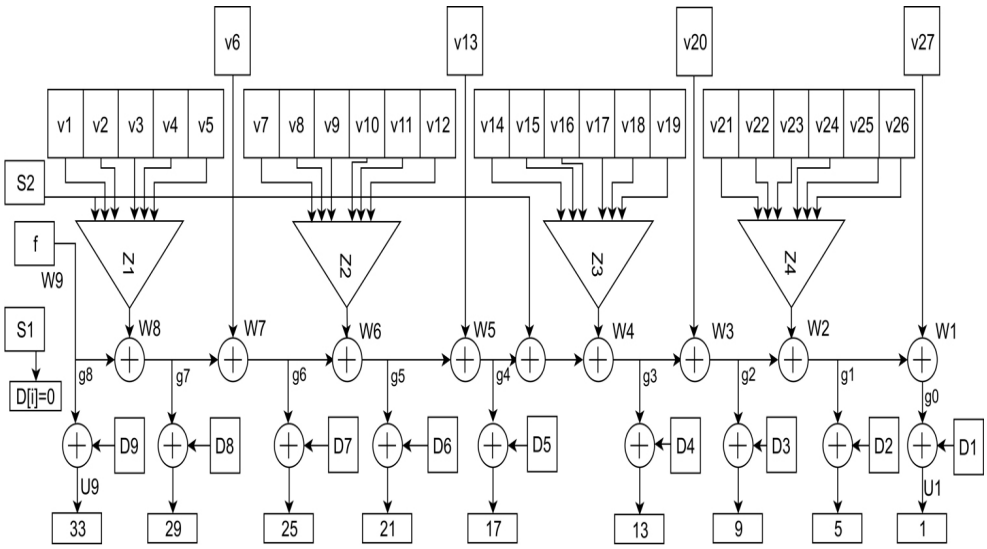

Fig. 4. The internal structure of one round of T-310 block cipher

[4] For example we connect $2\cdot 3$ inputs of D,C,B to the 6 inputs of W.

[5] For example if one input A is b the other must be c.

[6] For example we connect $2\cdot 3$ inputs of H,G,F to the 6 inputs of Y.

Proof:

We start by observing that, following the path from output 13 to 1 in Fig. 4, or adding together the equations (r6) and (r9) in Fig. 2 we get:

$$H^o = y_{13} + y_1 = x_{D(4)} + x_{P(20)} + W(.) + x_{P(27)} + x_{D(1)} = (W() + x_{P(23)} + x_{P(26)}) + (x_{32} + x_{20}) =$$
$$= (W + f + c)(.) + A^i$$

which is true knowing that $\{D(1), D(4), P(20), P(27)\} = \{5 \cdot 4, 8 \cdot 4, P(23), P(26)\}$.

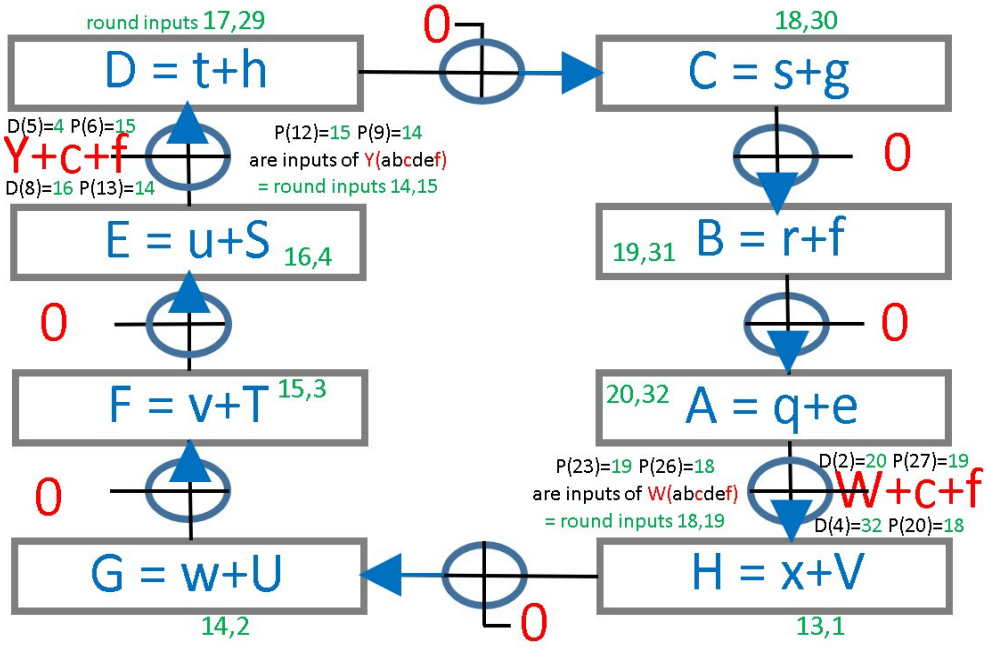

Fig. 5. Our attack can be studied in terms of a cycle of size 8 with on 8 basic polynomials A to H which are expected to be transformed into each other in a closed loop in every 8 consecutive applications of our cipher. The invariant polynomial is actually the product of all these $\mathscr{P} = ABCDEFGH$. Some transitions are trivial. Other are only true if the assumptions of our attack are satisfied which makes that two polynomials namely $(Y + c + f)$ and $(W + c + f)$ are simultaneously annihilated inside our attack.

Then following the path from output 29 to 17 in Fig. 4, or adding together the equations (r2) and (r5) in Fig. 2 we get:

$$D^o = y_{29} + y_{17} = x_{D(8)} + x_{P(6)} + Y(.) + x_{P(13)} + x_{D(5)} = (Y(.) + x_{P(9)} + x_{P(12)}) + (x_{16} + x_4) =$$
$$= (Y + f + c)(.) + E^i$$

which in turn works knowing that $\{D(5), D(8), P(6), P(13)\} = \{1 \cdot 4, 4 \cdot 4, P(9), P(12)\}$.

We also have 6 trivial transitions such as $A^o = B^i$ due to shifting of bits by one position, cf. (r0) in Fig. 2.

We now put this all together.

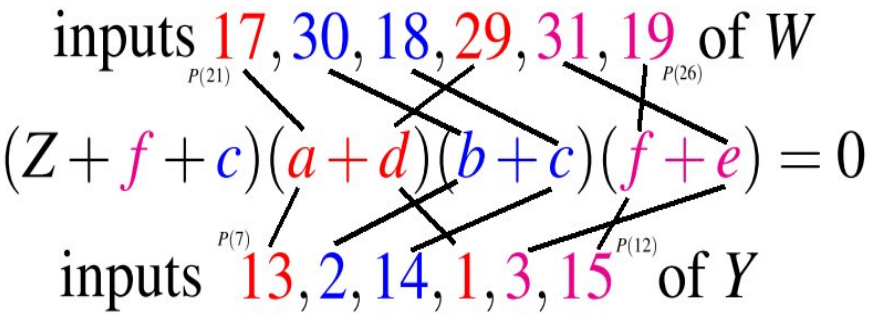

Fig. 6. Example of how we map inputs of W and Y to 3 sets with 2 variables out of 6 in a way consistent with our annihilation property

At the input side $\mathscr{P}$ is equal to $\mathscr{P}^i = ABCDEFGH$ and at the output of our cipher we have:

$$\mathscr{P}^o = A^oB^oC^oD^oE^oF^oG^oH^o = B^iC^iD^i((Y+f+c)()+E^i)F^iG^iH^i((W+f+c)()+A^i) =$$

at this moment only input variables are left and we can drop the exponents such as A^i and we get a pure expression which depends only on the input variables:

$$\mathscr{P}^o = BCD(Y(.)+E)FGH(W(.)+A) =$$

Now we observe that the inputs of $W()$ are $17, 30, 18, 29, 31, 19$, and our assumption was $(Z+f+c)(a+d)(b+c)(f+e) = 0$ which means that

$$(W+f+c)(B)(C)(D) = 0.$$

and this equation allows us to erase $(W+f+c)(BCD)$ because any multiple of it is equal to 0 and we get:

$$\mathscr{P}^o = BCD((Y+f+c)()+E)FGHA =$$

Likewise the inputs of $Y()$ are $13, 2, 14, 1, 3, 15$, and our assumption was $(Z+f+c)(a+d)(b+c)(f+e) = 0$ which means that

$$(Y+f+c)(F)(G)(H) = 0.$$

and this equation allows us to erase any multiple of $(Y+f+c)(FGH)$ leading to:

$$\mathscr{P}^o = BCDEFGHA = \mathscr{P}^i$$

which ends our proof.

5.3 Application Notes and Observations

One long term key which works is LZS 515 given below. It has a bijective round function which is not strictly required (except that T-310 would then be under threat of powerful attacks, [14]). It was generated with some trial and error using CryptoMiniSat to find collisions automatically and thus exclude keys which are not bijective.

```
515: P=9,5,7,34,33,15,13,2,14,1,3,15,14,11,26,16,6,5,
23,19,17,30,18,29,31,19,18 D=20,36,12,32,4,8,24,16,28
```

Notes. Remark: in our example we mapped inputs of W to B,C,D which are consecutive states on one half of our cycle in Fig. 5. This is NOT necessary, and we can use any 3 out of 6 polynomials B,C,D,F,G,H for W and the other 3 for Y. These 6 polynomials are such that they are mapped to another polynomial from the 8 basic polynomials A,B,C,D,E,F,G,H by a "trivial" transition which does not impose any conditions or one marked with 0 in Fig. 5. Such polynomials are sometimes called "transformable polynomials" using the terminology of [19] which name means that they belong to our set of 8 polynomials after transformation by the round function ϕ, for example we write $G = H^{\phi}$ which shows that H is a "transformable" polynomial. An interesting research question is to design a better attack which reuses these polynomials, cf. Appendix A.

Open Problems. One referee of this paper have asked what would be the percentage of long term keys which would satisfy the official KT1/KT2 criteria approved for government communications [25]. The answer is most likely 0 % with the current attack. In this paper we worked with paper and pencil and the attack is very simple. The next step is to search for further more complicated attacks for example following the methodology of [19] (and again subject to questions of polynomial reuse, cf. Appendix A). It appears that the space of possible attacks is extremely large and has not yet been studied in full. Therefore in our opinion an improved attack with a more complex invariant configuration which works with fully compliant KT1 or KT2 keys is likely to exist.

6 Conclusion

In this article we showed that it is possible to design a nonlinear invariant attack on T-310 on-demand, with the exact original Boolean function which was used to encrypt government communications in the 1980s. This was not achieved before, as far as we can see. We worked by formal algebra from a well-chosen annihilation property.

Our cipher wiring is very special, however such modifications are officially allowed, in the sense of being 100 % compatible with the original T-310 encryption hardware. Therefore we obtain a quasi-realistic attack scenario. This explains why it is important to be able to test cryptographic algorithms for defects when they are used in practice.

There are numerous ways in which a non-linear invariant attack could be exploited in cryptanalysis in order to actually decrypt some communications, cf. for example Section 6 in [5]. This is a complex technical and combinatorial optimization question which we consider to be outside the scope of this article.

References

1. Arbeitsgebiet 113: *Sachstandbericht zur Arbeit am Chiffrieralgorithmus des Gerätes T 310/50*, MfS-020-XI/674/76, 51 pages, Berlin, 31 December 1976, also known as MfS-Abt-XI-532
2. Arnaud Bannier, Nicolas Bodin, and Eric Filiol: *Partition-Based Trapdoor Ciphers*, `https://ia.cr/2016/493`.
3. Lawrence Brown, Jennifer Seberry, *On the design of permutation P in DES type cryptosystems*. In Eurocrypt 89, LNCS 434, pp. 696-705. Springer, 1990.
4. Pascale Charpin: *Normal Boolean functions*, Journal of Complexity, vol. 20, Issues 2–3, pp 245–265, 2004.
5. Nicolas T. Courtois, Marios Georgiou: *Variable elimination strategies and construction of nonlinear polynomial invariant attacks on T-310*, In Cryptologia, vol. 44, Iss. 1, pp. 20-38. At `https://doi.org/10.1080/01611194.2019.1650845`
6. Nicolas T. Courtois: *On the Existence of Non-Linear Invariants and Algebraic Polynomial Constructive Approach to Backdoors in Block Ciphers*, `https://ia.cr/2018/807`, last revised 27 Mar 2019.
7. Nicolas T. Courtois: *Structural Nonlinear Invariant Attacks on T-310: Attacking Arbitrary Boolean Functions*, `https://ia.cr/2018/1242`, revised 12 Sep 2019.
8. Nicolas Courtois: *Algebraic Attacks on Combiners with Memory and Several Outputs*, ICISC 2004, LNCS 3506, pp. 3–20, Springer 2005. Extended version available on `https://ia.cr/2003/125/`.
9. Nicolas Courtois: *Feistel Schemes and Bi-Linear Cryptanalysis*, in Crypto 2004, LNCS 3152, pp. 23–40, Springer, 2004.
10. Nicolas Courtois: *The Inverse S-box, Non-linear Polynomial Relations and Cryptanalysis of Block Ciphers*, in AES 4 Conference, Bonn May 10-12 2004, LNCS 3373, pp. 170-188, Springer, 2005.
    `https://www.researchgate.net/publication/221005723/\_The\_Inverse\_S-Box\_Non-linear\_Polynomial\_Relations\_and\_Cryptanalysis\_of\_Block\_Ciphers`
11. Nicolas Courtois, Guilhem Castagnos and Louis Goubin: *What do DES S-boxes Say to Each Other ?* Available on `https://ia.cr/2003/184/`.
12. Nicolas Courtois and Willi Meier: *Algebraic Attacks on Stream Ciphers with Linear Feedback*, Eurocrypt 2003, Warsaw, Poland, LNCS 2656, pp. 345–359, Springer. Extended version: `www.nicolascourtois.com/toyolili.pdf`.
13. Nicolas T. Courtois, Klaus Schmeh, Jörg Drobick, Jacques Patarin, Maria-Bristena Oprisanu, Matteo Scarlata, Om Bhallamudi: *Cryptographic Security Analysis of T-310*, Monography study on the T-310 block cipher, 132 pages, received 20 May 2017, last revised 29 June 2018, `https://ia.cr/2017/440.pdf`
14. Nicolas T. Courtois, Maria-Bristena Oprisanu: *Ciphertext-only attacks and weak long-term keys in T-310*, in Cryptologia, vol 42, iss. 4, pp. 316–336, May 2018. `http://www.tandfonline.com/doi/full/10.1080/01611194.2017.1362065`.
15. Nicolas Courtois, Jörg Drobick and Klaus Schmeh: *Feistel ciphers in East Germany in the communist era*, In Cryptologia, vol. 42, Iss. 6, 2018, pp. 427-444.
16. Nicolas Courtois, Maria-Bristena Oprisanu and Klaus Schmeh: *Linear cryptanalysis and block cipher design in East Germany in the 1970s*, in Cryptologia, 05 Dec 2018, `https://www.tandfonline.com/doi/abs/10.1080/01611194.2018.1483981`
17. Nicolas T. Courtois: *Structural Nonlinear Invariant Attacks on T-310: Attacking Arbitrary Boolean Functions*, `https://ia.cr/2018/1242`, revised 12 Sep 2019.

18. 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.
19. Nicolas T. Courtois, Matteo Abbondati, Hamy Ratoanina, and Marek Grajek: *Systematic Construction of Nonlinear Product Attacks on Block Ciphers*, In ICISC 2019, LNCS 11975, pp 20-51, Springer, 2020.
20. Hans Dobbertin: *Construction of bent functions and balanced Boolean functions with high nonlinearity*, in: FSE'94, LNCS 1008, Springer, Berlin, pp. 61–74, 1994.
21. Document MfS-Abt-XI-183, which is a documentation of SKS V/1 and contains a selection of pages extracted from a larger document known as MfS-020-Nr. 747/73, 1973.
22. S. Dubuc: *Etude des propriétés de dégénérescence et de normalité des fonctions booléennes et construction de fonctions q-aires parfaitement non-linéaires*, Ph.D. Thesis, Université de Caen, 2001.
23. C. Harpes, G. Kramer, and J. Massey: *A Generalization of Linear Cryptanalysis and the Applicability of Matsui's Piling-up Lemma*, Eurocrypt'95, LNCS 921, Springer, pp. 24–38.
24. Referat 11: *Kryptologische Analyse des Chiffriergerätes T-310/50. Central Cipher Organ, Ministry of State Security of the GDR, document referenced as 'ZCO 402/80', a.k.a. MfS-Abt-XI-594, 123 pages, Berlin, 1980.*
25. *Klaus Schmeh: The East German Encryption Machine T-310 and the Algorithm It Used*, In Cryptologia, vol. 30, iss. 3, pp. 251–257, 2006.
26. VEB Steremat "Hermann Schlimme", *Gerätesystem SKS V/1, Gerät DE1Z*eichnungs-Nr. 310017, Band 2, also known as MfS-Abt-XI-415, and a.k.a. B 86/1-31/77, Berlin, 1976.
27. Yosuke Todo, Gregor Leander, and Yu Sasaki: *Nonlinear invariant attack: Practical attack on full SCREAM, iSCREAM and Midori64*, In Journal of Cryptology, pp. 1–40, April 2018.
28. ZCO: *Charakterisierung der Booleschen Funktion Z*, handwritten document, MfS-020-XI/493/76, 24 pages, 1976.

A Future Attacks: Shared Factors and Fewer Variables

This article is the first successful attempt, to construct a realistic non-linear attack "on demand", given one fixed specific real-life Boolean function. We needed to adapt our attack to the Boolean function. It is obvious that a lot more can be done in this direction. Current research on invariant attacks on T-310 privileges annihilation equations of low degree which corresponds to the notion of k-normality in Boolean functions [4]. There is another major approach to this problem. To look at ways to annihilate Z with less than 6 variables (but maybe more factors). We have already see that with $(Z+b)adf = 0$. We list some more such solutions below which use only 4 active variables. With less variables it is easer to find an invariant attack, some variables are already eliminated (!).

In addition, if we want to apply the general attack framework of [19], it is clear that we need to **re-use** affine factors. This in order to optimize the total degree of the product attack, and also avoid product of polynomials $\mathscr{P}$ to become zero itself, compromising the hopes for a valid invariant attack. For this reason, it is particularly interesting to study cases when certain affine factors in 6 variables are repeated within a larger number of annihilation conditions. This leads to new very specific optimization problems. For example given k linear factors, maximize the number of affine shifts of Z which can be annihilated simultaneously. An example is worth a thousands words, therefore we give here an example of such a configuration with a shared factor of d.

$$(Z+1)*(1+a+b)*(d)*(1+a+f)=0$$

$$(Z+b)*(d)*(1+b+e)*(f)=0$$

Them main idea is that in one single attack where the invariant will impose that (among others) $d=1$ at each round, the polynomials $(Z+1)$ and $(Z+b)$ would be annihilated simultaneously for 2 different reasons. This example is incomplete, in the sense that it does not list numerous other related polynomials with some shared factors and with the same original Boolean function. The number of possibilities is simply very large. The reuse of annihilating factors is expected to increase the probability that some invariant property $\mathscr{P}$ would work on T-310. One method to construct such attacks could be the general cyclic product attack construction of [19].