Construction of a polynomial invariant annihilation attack of degree 7 for T-310

Nicolas T. Courtois, Aidan Patrick, Matteo Abbondati

University College London, Gower Street, London, UK

Abstract. Cryptographic attacks are typically constructed by black-box methods and combinations of simpler properties, for example in [Generalised] Linear Cryptanalysis. In this article we work with a more recent white-box algebraic-constructive methodology. Polynomial invariant attacks on a block cipher are constructed explicitly through the study of the space of Boolean polynomials which does not have a unique factorization and solving the so-called Fundamental Equation (FE). Some recent invariant attacks are quite symmetric and exhibit some sort of clear structure, or work only when the Boolean function is degenerate. As a proof of concept we construct an attack where a highly irregular product of 7 polynomials is an invariant for any number of rounds for T-310 under certain conditions on the long term key and for any key and any IV. A key feature of our attack is that it works for any Boolean function which satisfies a specific annihilation property. We evaluate very precisely the probability that our attack works when the Boolean function is chosen uniformly at random.

Key Words: Cold War, modern block ciphers, Feistel ciphers, T-310, multivariate polynomials, polynomial invariants, Boolean functions, annihilator space, ANF, polynomial rings, unique factorisation, weak keys, backdoors, Generalized Linear Cryptanalysis, algebraic cryptanalysis.

1 Block Ciphers and Round Invariant Attacks

Block ciphers are in widespread use since the 1970s. Their iterated structure is prone to numerous round invariant attacks, for example in Linear Cryptanalysis (LC). The next step is to look at non-linear polynomial invariants with Generalised Linear Cryptanalysis (GLC) first proposed by Harpes, Kramer, and Massey cf. [22] (Eurocrypt'95). A major open problem in cryptanalysis is the discovery of new invariant properties of non-trivial and complex type. Until recently, researchers have found extremely few such attacks, with some impossibility results [23,2,27,4]. Eventually recent works [26,8] show how to construct polynomial invariant attacks for block ciphers. For T-310 such an attack was first suggested in Section 7 of [14]. One specific example of such attack which was discussed was the key 771 in Section 8.3. in [8]. However the 771 solution of [8] and countless other solutions in [8] are not yet quite satisfactory. They only works when the Boolean function Z is modified and when it is very special. In almost all such results so far the Boolean function is extremely weak, the space of Boolean functions which work is small, and invariants are simple and of low degree. Can we do better? Do some complex attacks occur by accident or do they follow clear rules and how can we construct a sophisticated irregular attack step by step? In this article we demonstrate just one specific attack which has an interesting feature: it works for a large variety of Boolean functions including sometimes, or with a certain probability, when it is chosen at random.

1.1 Our Block Cipher

Our attack is constructed for T-310, an old Cold War Feistel cipher with 4 branches. This cipher offers enormous flexibility in the choice of the internal wiring. Most ciphers such as DES or AES also have this sort of flexibility in the choice of P-boxes, arbitrary invertible matrices inside the S-box, inside the mixing layers, however later these components are fixed. In T-310 this flexibility is "officially" supported: a large variety of possible choices of cipher wiring can be specified and used. Here if we find a weak setup, it will be entirely compatible with the original historical hardware. The exact cipher wiring specification in T-310 is called LZS or *Langzeitschlüssel* cf. and various keys studied by researchers and various known complete specifications are denoted by 2 digit or 3 digit numbers such as LZS 31 or LZS 903, cf. [20,25]. Our cipher uses Boolean functions on 6 variables which in our work will become a variable Z which has 64 bits, later called $Z00-Z63$. Initially we studied degenerated cases, and after many attempts we end with a scenario where this Boolean function is no longer chosen by the attacker, and a single attack works sometimes, for example, with probability 2^{-16}, for any Boolean function chosen at random. The specific type of weakness which we will exploit here is that a Boolean function in 6 variables almost always has numerous annihilators of degree 3, cf. Thm. 6.0.1. in [7], and furthermore sometimes it has further annihilators of a specific form. This abundant existence of annihilators can be attributed to the lack of unique factorization in the ring of Boolean polynomials B_N.

1.2 Boolean Polynomials, Annihilators and Absorbers

Let B_N be the ring of Boolean polynomials in N variables (polynomials in their ANF without powers or with $x^2 = x$ cancellations done when multiplying the polynomials). For the T-310 cipher we have $N = 6$. In this article we do not use the annihilator method of [9] but we rather work on absorption properties. A polynomial f absorbs g if $fg = f$. In theory both sorts of events are equivalent, absorption of g is the same as annihilation with $f(g + 1) = 0$. Annihilation, absorption and lack of unique factorization are key mathematical events which occur many times and are instrumental in making our attack work.

1.3 Round Invariants for Block Ciphers

Current research in application of polynomial invariants in symmetric cryptography has concentrated a lot on negative results, cf. [4,2,27] and has lacked substance or material to work with, in the form of real-life positive examples which work. Numerous results are about cipher components rather than full ciphers. For example for the AES-like S-box, it is possible to use the so called cross-ratio. However this type of invariant is still quite simple or operates on only one variable. In our research we study a substantially wider variety of multivariate

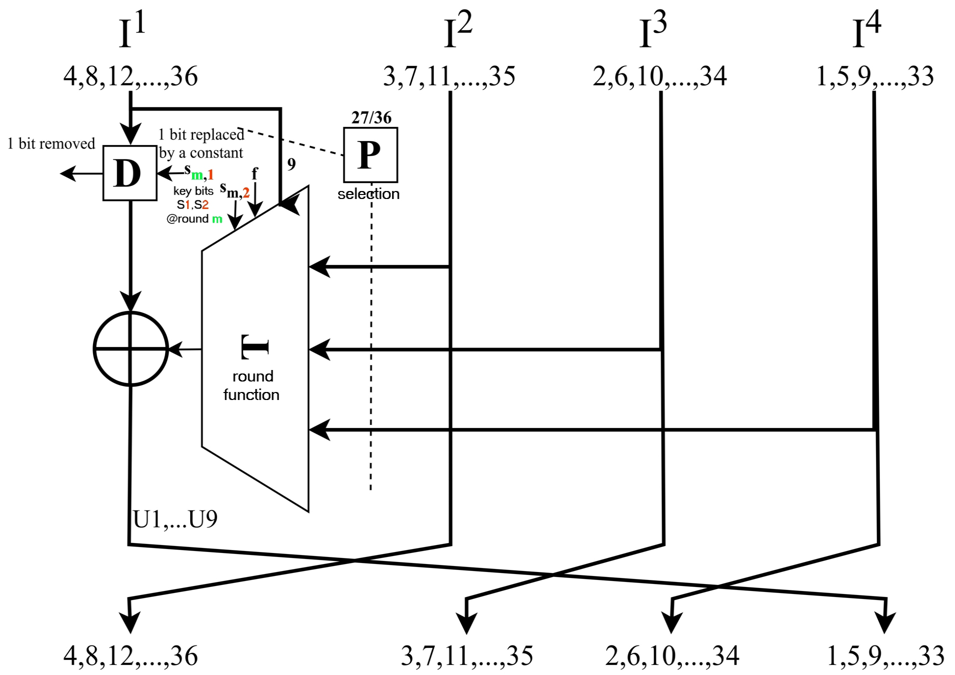

Fig. 1. T-310: a peculiar sort of Compressing Unbalanced Feistel scheme.

invariants with increasing size and complexity. In cryptographic invariants the main object of interest are **round invariants** for one round. Examples which work for more than one round also exist and a large variety of examples can be found in [8].

We would like to have $\mathcal{P}$(Inputs) = $\mathcal{P}$(Output ANF) where $\mathcal{P}$ will be a Boolean function. We are looking for polynomials $\mathcal{P}$ the value of which does not change after we apply a transformation called "a round" we call ϕ. This round function ϕ is typically a bijection and is like one round of encryption. In addition, typically it is NOT one fixed permutation but it has a parameter, a secret key and potentially additional parameters. The more parameters, the harder it becomes to find invariants. For example the T-310 cipher can be viewed as each round is applying one of the 8 possible permutations $\phi_0 : \{0,1\}^{36} \to \{0,1\}^{36}$ up to $\phi_7 : \{0,1\}^{36} \to \{0,1\}^{36}$ and the choice of which ϕ is actually used depends on 2 bits of the secret key and 1 bit of the IV (which is public and known to the attacker), all using the original notations of [24]. Technically speaking, just finding such invariants is easy and they exist in vast numbers, yet many are in some sense trivial or degenerated, cf. [8]. A key problem is to find **simultaneous invariants** that hold in all the eight cases, i.e. for all of $\phi_0 \dots \phi_7$ simultaneously. This was a big problem in early research on this topic cf. [8] but it is not a problem in this article (we construct a solution which works nevertheless and there are many more variables which happily are also eliminated).

1.4 The Role of [Unique] Polynomial Factorisation

One of the crucial insights in our attack is that the space of Boolean functions does not have a unique factorisation. This topic has an interesting history and was pioneered by a Russian mathematician Zhegalkin as early as 1927, cf. [28]. In 1936 an American mathematician Marshall Stone has reflected on the fact that this is a very useful and simple method to "arithmetize" the Boolean algebra, which is of great interest and led Stone to substantially rewrite his paper, cf. [28]. Today researchers prefer to use the terms such as ring Boolean polynomials B_n for n variables, or Algebraic Normal Form (ANF), or work in certain quotient polynomial rings, which are all essentially the same.

The main question which makes the impossible possible in this article, is the lack of unique factorisation in B_n. This has very important consequences in cryptanalysis which is further emphasised if we look at the exact result and construction in this article. We have our main invariant polynomial $\mathcal{P}$, which is product of 7 simpler polynomials. With unique factorisation it would be extremely difficult to have a polynomial like the one we use to be invariant for a block cipher. There will be only some "trivial" ways to do it, for example that each polynomial is itself a linear invariant for 1 round, or a solution with a cyclic permutation. A cyclic solution would imply that each polynomial is a linear invariant after 7 rounds (not 1). We only get trivial attacks, or a cipher pathologically weak w.r.t. classical Linear Cryptanalysis.

With a lack of unique factorisation we can do a lot better. There will also be new non-trivial solutions which do NOT decompose into simpler attacks and can only be shown to hold as a whole by polynomial algebra. Moreover inside our proof that our attack actually works there will be also numerous additional events where the lack of unique factorisation is needed or plays an important role. For example, when we factor some polynomial or when we discover that annihilators of a specific form exist with a relatively large probability or when we show that different types of annihilators are related, cf. later Lemma 4.3. More interestingly, it happens also when we establish later that $Y\mu = \mu$ and $W\mu = \mu$ with the same μ, while the sets of variables used in polynomials Y and W are entirely disjoint. Finally, it is also visible when in order to prove these two absorptions we will factor μ in two radically different ways with disjoint variables, cf. page 4. Some of such "lack of unique factorisation events" are surprising and counter-intuitive and they are the heart of what makes that T-310 can be broken.

2 Non-Linear Cryptanalysis through Formal Coding

The concept of cryptanalysis with non-linear polynomials a.k.a. Generalized Linear Cryptanalysis (GLC) was introduced at Eurocrypt'95, cf. [22]. A key question is the existence of round-invariant I/O sums: when a value of a certain polynomial is preserved after 1 round. Many researchers have in the past failed to find any such properties, Bi-Linear and Multi-Linear attacks were introduced [11,12] for Feistel ciphers branches specifically. In this article and unlike in [26] we focus on invariants which work for 100 % of the keys and we focus on stronger invariants which hold with probability 1.

We call $\mathcal{P}$ a polynomial invariant if the value of $\mathcal{P}$ is preserved after one round of encryption, i.e. if $\mathcal{P}(\text{Inputs}) = \mathcal{P}(\text{Output ANF})$. In this article we work with one specific block cipher with 36-bit blocks. The main point is that any block cipher round translates into relatively simple Boolean polynomials, if we look at just one round. We follow the methodology of [8] in order to specify the exact mathematical constraint, known as the Fundamental Equation or FE, cf. Section 3, so that we could have a polynomial invariant attack on our cipher. Such an attack will propagate for any number of rounds (if independent of key and other bits). In addition it makes sense following [8] to consider that the Boolean function is an unknown. We denote this function by a special variable Z. We then see that our attack works if and only if Z is a solution to a certain algebraic equation [with additional variables]. The main interest of making Z a variable is to see that even if Z is extremely strong, some advanced "product" attacks might sometimes work nevertheless, as will be shown below.

2.1 Notation and Methodology

In this article the sign + denotes addition modulo 2, and frequently we omit the sign * in products. For the sake of compact notation we frequently use short or single letter variable names. For example let $x_1, \ldots, x_{36}$ be inputs of a block cipher each being $\in \{0, 1\}$. We will avoid this notation and name them with small letters $a - z$ and letters $M - V$ when we run out of lowercase letters. We follow the backwards numbering convention of [8] with $a = x_{36}$ till $z = x_{11}$ and then we use specific capital letters $M = x_{10}$ till $V = x_1$, see Fig. 4 page 9. This avoids some "special" capital letters following notations used since the 1970s [16,25,24]. We consider that each round of encryption is identical except that they can differ only in some "public" bits called F (which are known to the attacker) and some "secret" bits called $S1$ or K and $S2 = L$. Even though these bits ARE different in different rounds we will omit to specify in which round we take them because our work is about constructing **one round** invariants (extending to any number of rounds). This framework covers most block ciphers ever made except that some ciphers would have more "secret" or "public" bits in one round. The capital letter Z is a placeholder for substitution of the following kind

$$Z(e_1, e_2, e_3, e_4, e_5, e_6)$$

where $e_1 \ldots e_6$ will be some 6 of the other variables. In practice, the e_i will represent a specific subset of variables of type *a*-*z*, or others, such as L. At the end Z must be replaced by a formula like:

$$Z \leftarrow Z00 + Z01 * L + Z02 * c + Z03 * Lc + \ldots + Z62 * cklfh + Z63 * Lcklfh$$

where Zij are coefficients of the Algebraic Normal Form (ANF).

2.2 Constructive Approach Given the Cipher Wiring

Our attack methodology starts from a given block cipher specified by its ANFs for one round. In practice we will work with T-310. The block size is 36 bits and the key has 240 bits. The hardware encryption cost with T-310 is hundreds of times bigger than AES or 3DES, cf. [16]. Does it make this cipher very secure? Not quite, if we can construct algebraic invariants which work for any number of rounds.

2.3 ANF Coding of One Full Round

We number the cipher state bits from 1 to 36 where bits $1, 5, 9 \ldots 33$ are those freshly created in one round, cf. Fig 1. Let $x_1, \ldots, x_{36}$ be the inputs and let $y_1, \ldots, y_{36}$ be the outputs. One round of our cipher can be described as 36 Boolean polynomials out of which only 9 are non-trivial:

$$
\begin{aligned}
y_{33} &= F + x_{D(9)} \\
&Z1 \overset{def}{=} Z(S2, x_{P(1)}, \ldots, x_{P(5)}) \\
y_{29} &= F + Z1 + x_{D(8)} \\
y_{25} &= F + Z1 + x_{P(6)} + x_{D(7)} \\
&Z2 \overset{def}{=} Z(x_{P(7)}, \ldots, x_{P(12)}) \\
y_{21} &= F + Z1 + x_{P(6)} + Z2 + \quad x_{D(6)} \\
y_{17} &= F + Z1 + x_{P(6)} + Z2 + \quad x_{P(13)} + x_{D(5)} \\
&Z3 \overset{def}{=} Z(x_{P(14)}, \ldots, x_{P(19)}) \\
y_{13} &= F + Z1 + x_{P(6)} + Z2 + \quad x_{P(13)} + S2 + Z3 + x_{D(4)} \\
y_{9} &= F + Z1 + x_{P(6)} + Z2 + \quad x_{P(13)} + S2 + Z3 + x_{P(20)} + x_{D(3)} \\
&Z4 \overset{def}{=} Z(x_{P(21)}, \ldots, x_{P(26)}) \\
y_{5} &= F + Z1 + x_{P(6)} + Z2 + \quad x_{P(13)} + S2 + Z3 + x_{P(20)} + Z4 + x_{D(2)} \\
y_{1} &= F + Z1 + x_{P(6)} + Z2 + \quad x_{P(13)} + S2 + Z3 + x_{P(20)} + Z4 + x_{P(27)} + x_{D(1)} \\
&x_0 \overset{def}{=} S1 \\
y_{i+1} &= x_i \text{ for all other } i \neq 4k \qquad (\text{ with } 1 \leq i \leq 36)
\end{aligned}
$$

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

Two things remain unspecified: the P and D boxes or the internal wiring. In T-310 this specification is called an LZS or *Langzeitschlüssel* which means a long-term key setup. We simply need to specify two functions $D : \{1 \ldots 9\} \rightarrow \{0 \ldots 36\}$, $P : \{1 \ldots 27\} \rightarrow \{1 \ldots 36\}$. For example $D(5) = 36$ means that input bit 36 is connected to the wire which becomes $U5 = y_{17}$ after XOR of Fig. 1. Then $P(1) = 25$ means that input 25 is connected as v1 or the 2nd input of $Z1$. We also apply a special convention where the bit S1 is used instead of one of the $D(i)$ by specifying that $D(i) = 0$.

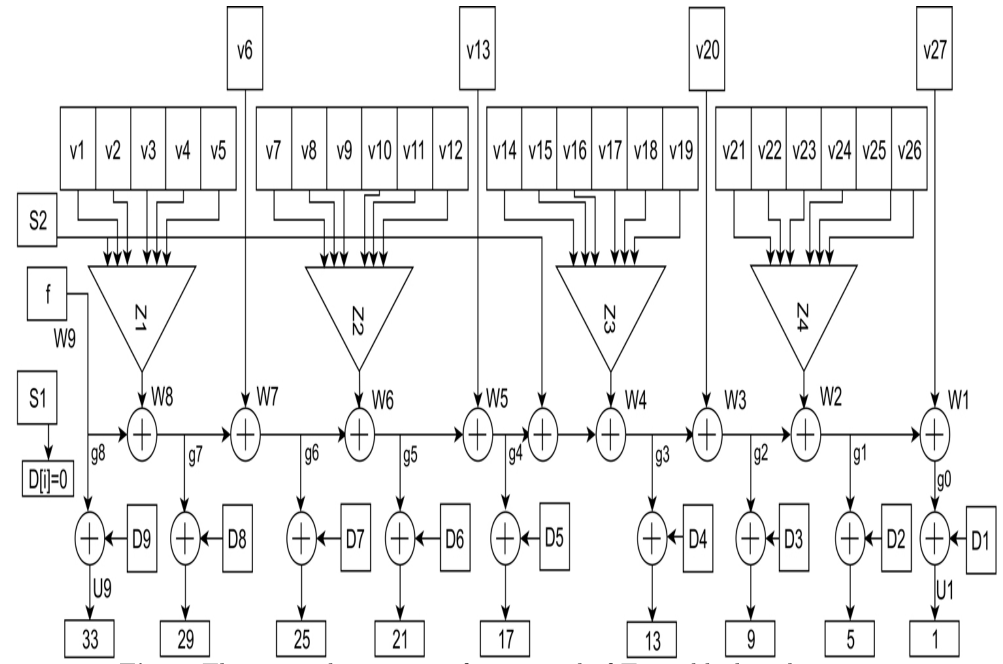

Fig. 3. The internal structure of one round of T-310 block cipher.

2.4 The Substitutions

Overall one round can be described as 36 Boolean polynomials of degree 6; out of which only 9 are non-trivial. One round of encryption is viewed as a sequence of substitutions where an output variable is replaced by a polynomial algebraic expression in the input variables. Here is a (shortened) example following the cipher specification step-by-step for the long-term key 551 used in [8]:

$$\begin{aligned}
a &\leftarrow b \\
b &\leftarrow c \\
c &\leftarrow d \\
d &\leftarrow F + i \\
&[\ldots] \\
&[\ldots] \\
V &\leftarrow F + Z1 + O + Z2 + q + L + Z3 + i + Z4 + k + K
\end{aligned}$$

In order to have shorter expressions to manipulate we frequently replace $Z1 - Z4$ by shorter abbreviations Z, Y, X, W respectively. We also replace S2 by a single letter L (used at 2 places). The other key bits $S1 = K$ will only be used if some $D(i) = 0$ due to the special convention defined above: S1 is then used and XORed at the output instead of one of the $x_{D(i)}$ bits from the previous round.

3 The Fundamental Equation

In order to break our cipher we need to find a polynomial expression $\mathcal{P}$ say

$$\mathcal{P}(a,b,c,d,e,f,g,h,\ldots) = abcdijkl + efg + efh + egh + fgh$$

using any number between 1 and 36 variables such that if we substitute in $\mathcal{P}$ all the variables by the substitutions defined we would get exactly the same polynomial expression $\mathcal{P}$, i.e. $\mathcal{P}(\text{Inputs}) = \mathcal{P}(\text{Output ANF})$ are equal as multivariate polynomials. We obtain:

Definition 3.1 (Compact Uni/Quadri-variate FE). Our "Fundamental Equation (FE)" to solve is simply a substitution like:

$$\mathcal{P}(\text{Inputs}) = \mathcal{P}(\text{Output ANF})$$

or more precisely

$$\mathcal{P}(a,b,c,d,e,f,g,h,\ldots) = \mathcal{P}(b,c,d,F+i,f,g,h,F+Z1+e,\ldots)$$

where again $Z1 - Z4$ are replaced by Z,Y,X,W. In the next step, Z will be replaced by an Algebraic Normal Form (ANF) with 64 binary variables which are the coefficients of the ANF of Z, and there will be several equations, and four **instances** Z,Y,X,W of the same Boolean function:

Definition 3.2 (A Multivariate FE). At this step we will rewrite FE as follows. We will replace Z1 by:

$$Z \leftarrow Z00 + Z01*L + Z02*j + Z03*Lj + \ldots + Z62*jhfpd + Z63*Ljhfpd$$

Likewise we will also replace $Z2$:

$$Y \leftarrow Z00 + Z01*k + Z02*l + Z03*kl + \ldots + Z62*loent + Z63*kloent$$

and likewise for $X = Z3$ and $W = Z4$ and the coefficients $Z00\ldots Z63$ will be the same inside $Z1-Z4$, however the subsets of 6 variables chosen out of 36 will be different in $Z1-Z4$. Moreover, some coefficients of $\mathcal{P}$ may also be variable.

In all cases, all we need to do is to solve the equation above for Z, i.e. determine 64 binary variables $Z00\ldots Z63$. This formal algebraic approach, if it has a solution, still called Z for simplicity, or $(\mathcal{P},Z)$ will **guarantee** that our invariant $\mathcal{P}$ holds for 1 round. This is, and in this article we are quite lucky, IF this equation does not depend on three bits F,K,L. Solving this equation in general, is the general discovery process of [8] which we do not use here. We rather work with basic paper and pencil maths and build our attack from scratch in stages.

4 A New Invariant Attack of Degree 7

We would like to show that a strong invariant property of type $\mathcal{P}(\text{Inputs}) = \mathcal{P}(\text{Output ANF})$ can be "engineered" and will happen under specific conditions of our choice, for any number of rounds any key and any IV. We define the following set of 8 (linear) polynomials with 36 variables (which are the same as in [9]):

$$\begin{cases} A \overset{def}{=} (i+m) & \text{which is bits } 24, 28 \\ B \overset{def}{=} (j+n) & \text{which is bits } 23, 27 \\ C \overset{def}{=} (k+o) & \text{which is bits } 22, 26 \\ D \overset{def}{=} (l+p) & \text{which is bits } 21, 25 \\ E \overset{def}{=} (y+O) & \text{which is bits } 8, 12 \\ F \overset{def}{=} (z+P) & \text{which is bits } 7, 11 \\ G \overset{def}{=} (M+Q) & \text{which is bits } 6, 10 \\ H \overset{def}{=} (N+R) & \text{which is bits } 5, 9 \end{cases}$$

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. 4. Variable naming conventions

Theorem 4.1 (A Degree 7 Invariant Attack). Let
$$\mathcal{P} = (1+A+H)(B+H)(1+C+H)(D+H)(E+H)(1+F+H)(G+H)$$

then $\mathcal{P}$ is a non-zero polynomial of degree 7. We also assume that

$$\begin{cases} \{D(2), D(3)\} = \{6 \cdot 4, 7 \cdot 4\} \\ \{D(6), D(7)\} = \{2 \cdot 4, 3 \cdot 4\} \end{cases}$$

and that inputs of Y are in order bits $27, 6, 10, 23, 21, 25$ and inputs of W are in order bits $26, 9, 5, 22, 7, 11$. If the Boolean function used inside the cipher has, after adding 1, an annihilator as follows:

$$(Z+1) * (a+d+b+c+1)(a+d+e+f+1) = 0 \tag{1}$$

Then $\mathcal{P}$ is a round invariant for any key, any IV, and any number of rounds.

Remark 1: We call this a "product attack" because $\mathcal{P}$ is a product of several polynomials. Our attack was initially inspired by [9] and designed for LZS 265 of [9]. However this attack is substantially less trivial and does not have the strong symmetries compared to the product attack in [9]. It uses very few actual properties of LZS 265, and any key which has the 4+6+6 properties listed above will also be broken. This however will exclude any real-life keys [20].
Remark 2: It may seem that Boolean functions which satisfy these properties are extremely rare. In reality annihilators of degree 3 are near-systematic cf.

Thm. 6.0.1. in [7], and annihilators of degree 2 of the sort we are looking for here do also happen sometimes, for example when the Boolean function is chosen at random, see Section 5. One example of a Boolean function which works here is

$$Z = b+ac+bc+abc+bd+abd+bcd+abcd+e+ce+ace+bde+af+bf+abf+bcf+df+cdf+ abcdf + ef + bef + cef + acef + bcef + bcdef + abcdef + 1$$

Additionally, there exist numerous permutations of inputs of Y and W which will also work if letters $a - f$ are also permuted accordingly. For simplicity we study only the basic attack without these extra variants.

Proof of Thm. 4.1. We work on round invariants, and we recall that we call the variable x_{36} at the input and y_{36} at the output by the same letter a, cf. Fig. 4 above. Then if A is some polynomial, say $a + N$ or $9, 36$ we will call the input-side version of it $A^i = x_9 + x_{36}$. The output side version will be called $A^o = y_9 + y_{36}$. At a later moment we will eliminate all output variables and use only input variables, at this moment it will no longer be necessary to distinguish input and output sides: it will be inputs only. We recall our assumption:

$$\begin{cases} \{D(2), D(3)\} = \{6 \cdot 4, 7 \cdot 4\} \\ \{D(6), D(7)\} = \{2 \cdot 4, 3 \cdot 4\} \end{cases}$$

and following [9] or simply following step by step a walk from output 9 to output 5 in Fig. 3 above we see that:

$$H^o = y_9 + y_5 = x_{D(3)} + W(.) + x_{D(2)} = W(.) + A^i \qquad (2a)$$

$$D^o = y_{25} + y_{21} = x_{D(7)} + Y(.) + x_{D(6)} = Y(.) + E^i \qquad (2b)$$

Here we see how the polynomials D and H on the output side can be rewritten as expressions using only input-side variables (where all bits FKL are already eliminated). polynomial $\mathcal{P}$ to itself on the output side Then we remark that some transitions are trivial for example $H^i = G^o$ and many other: $H \to G \to F \to E$ and $D \to C \to B \to A$. The output-side polynomial,

$$(1+A^o+H^o)(B^o+H^o)(1+C^o+H^o)(D^o+H^o)(E^o+H^o)(1+F^o+H^o)(G^o+H^o)$$

Is then equal to:

$$(B^i + W(.) + A^i + 1)\ (C^i + W(.) + A^i + 1)\ (Y(.) + E^i + W(.) + A^i)$$
$$(D^i + W(.) + A^i)\ (E^i + W(.) + A^i + 1)(H^i + W(.) + A^i)$$

at this moment we have only inputs left and we can use shorter notations and the remainder of the proof will be a formal game of equality of Boolean polynomials with 36 input variables and 3 F, K, L variables.

$$\begin{aligned} &(B + W + A + 1)(C + W + A + 1)(Y + E + W + A)(D + W + A) \\ &(E + W + A + 1)(H + W + A) \end{aligned} \qquad (3)$$

Finally we add the last expression to the input polynomial $(1 + A + H)(B + H)(1 + C + H)(D + H)(E + H)(1 + F + H)(G + H))$ and obtain that the our invariant holds if and only if our sum of the two polynomials is zero. In other terms our FE sum, which we would like to be zero, is exactly equal to:

```
ABCDFGY+ABCDFG+ABCDFHY+ABCDFH+ABCDGHY+ABCDGH+ABCDGY+ABCDG+ABCFGHY+ABCFGH+
ABCFHY+ABCFH+ABDFGHY+ABDFGH+ABDFGY+ABDFG+ABDGHY+ABDGH+ABDGY+ABDG+ACDFGHY+
ACDFGH+ACDFHY+ACDFH+ACFGHY+ACFGH+ACFHY+ACFH+BCDEFGW+BCDEFG+BCDEFHW+BCDEFH+
BCDEGHW+BCDEGH+BCDEGW+BCDEG+BCDFGHW+BCDFGHY+BCDFGWY+BCDFGW+BCDFHWY+BCDFHY+
BCDGHWY+BCDGHW+BCDGWY+BCDGW+BCEFGHW+BCEFGH+BCEFHW+BCEFH+BCFGHWY+BCFGHY+
BCFHWY+BCFHY+BDEFGHW+BDEFGH+BDEFGW+BDEFG+BDEGHW+BDEGH+BDEGW+BDEG+BDFGHWY+
BDFGHW+BDFGWY+BDFGW+BDGHWY+BDGHW+BDGWY+BDGW+CDEFGHW+CDEFGH+CDEFHW+CDEFH+
CDFGHWY+CDFGHY+CDFHWY+CDFHY+CEFGHW+CEFGH+CEFHW+CEFH+CFGHWY+CFGHY+CFHWY+CFHY
```

which we have been able[1] to factor as:

$$\begin{aligned}&(B+C)(G+H)(B+H)(B+F)(C+D)\\ &[(A+H)Y+(E+H+1)W+\\ &\quad(D+H)YW+(D+H)(A+E)]\end{aligned}\tag{4}$$

Let the product of first 5 multiplicative terms be denoted by a special letter:

$$\mu=(B+C)(G+H)(B+H)(B+F)(C+D)\tag{5}$$

all the we need to show now is that μ annihilates the product of all other factors and our work is finished. An easy observation is that our equation reduces to 0 when $Y, W=1$. In Sage we type:

```
sage: R.<A,B,C,D,E,F,G,H> = BooleanPolynomialRing(8)
sage: lhs = (1+A+H)*(B+H)*(1+C+H)*(D+H)*(E+H)*(1+F+H)*(G+H)
sage: rhs = (B+A)*(C+1+A)*(D+A)*(E+A)*(F+1+A)*(G+A)*(H+1+A)
sage: lhs == rhs
True
```

Our work would therefore be finished if $Y=W=1$ which they are not, as they are complex polynomials in 6 variables each. Now imagine that

Lemma 4.2. Under assumptions of Thm. 4.1 we have

$$Y\mu=\mu \quad \text{and} \quad W\mu=\mu\tag{6}$$

The proof of this lemma appears later below. If we admit this theorem, then because μ is a factor here and all the Y, W are elsewhere, we are actually **allowed** to replace Y by 1 because it is multiplied by μ anyway and $Y\cdot\mu$ by $1\cdot\mu$. All we need to do now is to show that $Y\mu=\mu$ and $W\mu=\mu$ which is somewhat surprising, because the sets of variables used in these two polynomials Y and W are expected to be entirely disjoint. We need a little Lemma.

[1] One working method to do this is to use the function F.annihilator(1, dim=True) in SAGE and divide by f each time we find a degree 1 annihilator f.

Lemma 4.3. If we have $(Z+1)*(a+d+b+c+1)(a+d+e+f+1)=0$ as in our assumptions then we also have

$$(Z+1)(f+e)(d+a)(b+c)=0 \tag{7}$$

Proof of Lemma 4.3. We observe that for any 3 Boolean polynomials $Wfg=0$ is equivalent to $W(f+h)g=0$ for each polynomial h which annihilates g. In particular when $h=g+1$ we get $Wfg=0 \Leftrightarrow W(f+g+1)g=0$. Accordingly:

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

$$(Z+1)*(a+d+e+f+1)(d+a)(d+a+b+c+1)$$

which means the latter polynomial must be zero for any input, because it is a multiple of our assumption $(Z+1)*(a+d+b+c+1)(a+d+e+f+1)=0$.

For convenience we recall all our definitions of $A-F$ below

$$\begin{cases} A \overset{def}{=} (i+m) & \text{which is bits } 24, 28 \\ B \overset{def}{=} (j+n) & \text{which is bits } 23, 27 \\ C \overset{def}{=} (k+o) & \text{which is bits } 22, 26 \\ D \overset{def}{=} (l+p) & \text{which is bits } 21, 25 \\ E \overset{def}{=} (y+O) & \text{which is bits } 8, 12 \\ F \overset{def}{=} (z+P) & \text{which is bits } 7, 11 \\ G \overset{def}{=} (M+Q) & \text{which is bits } 6, 10 \\ H \overset{def}{=} (N+R) & \text{which is bits } 5, 9 \end{cases}$$

and we recall that inputs of Y are in order $27, 6, 10, 23, 21, 25$ which is the same as n, M, Q, j, l, p and inputs of W are $26, 9, 5, 22, 7, 11$ or o, R, N, k, z, P. From (7) due to Lemma 4.3 applied twice for both W and Y we obtain:

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

$$(Y(n,M,Q,j,l,p)+1)(p+l)(j+n)(M+Q)=0.$$

$$(W(o,R,N,k,z,P)+1)(P+z)(k+o)(R+N)=0.$$

Here the order of the 6 variables matters and we obtain that:

$$CHF \cdot W = CHF \quad \text{and} \quad BDG \cdot Y = BDG \tag{8}$$

We then rewrite in a similar manner our initial assumption $(Z+1)*(d+a+b+c+1)(a+d+e+f+1)=0$ twice and we obtain:

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

$$(W(o,R,N,k,z,P)+1)*(k+o+R+N+1)(o+k+z+P+1)=0$$

$$(Y(n,M,Q,j,l,p)+1)*(j+n+M+Q+1)(n+j+l+p+1)=0$$

We have obtained two more absorption properties:

$$(C+H+1)(C+F+1)\cdot W=(C+F+1)(C+H+1) \tag{9}$$

and

$$(B+D+1)(B+G+1)\cdot Y=(B+D+1)(B+G+1) \tag{10}$$

Proof of Lemma 4.2. Now we are ready to prove also the earlier Lemma 4.2 which was omitted until this point (and was not used in the equations established above, therefore there is no circular argument). We need to show that under assumptions of our Thm. 4.1 we have:

$$Y\mu=\mu \quad \text{and} \quad W\mu=\mu$$

This part is non deterministic, and we find it surprising. Because factorisation is not unique, we have some freedom in how we factor[2] μ. We recall

$$\mu=(B+C)(G+H)(B+H)(B+F)(C+D) \tag{11}$$

We want to separate the B, D, G terms from the C, H, F terms, in order to absorb W and Y respectively. By the same randomized process using SAGE function .annihilator as above we try to factor this polynomial several times and in two different attempts we get that:

$$\mu=(C+H+1)(C+F+1)(BDG+H(B+D+1)(B+G+1)) \tag{12a}$$

$$\mu=(B+D+1)(B+G+1)(CFH+G(C+H+1)(C+F+1)) \tag{12b}$$

these two facts imply that both Y and W can be absorbed by μ. More precisely in (12a) we see that W is absorbed by the first two terms of the factorisation using (9) therefore $W\mu=\mu$. Similarly in (12b) Y is absorbed by the first factor using (10) therefore $Y\mu=\mu$. Interestingly there is another proof: Y can also be absorbed by the last factor of (12a) using both second part of (8) and (10) and W can also be absorbed by the last factor of (12b) using both the first part of (8) and (9). This ends the proof of earlier Lemma 4.2, which was the only thing which remained to prove.

[2] In Section 6 of [18] we will find the same μ with a different factorrisation.

4.4 Related Research, Redundancy and Equivalent Attacks

In our Thm. 4.1 we have

$$\mathcal{P} = (1+A+H)(B+H)(1+C+H)(D+H)(E+H)(1+F+H)(G+H)$$

and in [18] we find another attack of degree 7 with

$$\mathcal{P} = (A+B)\ (C+D)\ (D+F)(B+F)\ (E+F)(G+F)(G+H).$$

It is quite surprising and nevertheless true, that both polynomials are actually identical, which is possible because we do not have unique factorisation. Actually, both are attacks are completely identical. The attack in [18] has two annihilation conditions which needs to hold simultaneously:

$$(Z+1)*(f+e)(d+a)(b+c) = 0 \qquad (13a)$$

$$(Z+1)*(f+e+1)(d+a+1)(b+c+1) = 0 \qquad (13b)$$

It is possible to show that this is simply equivalent to our attack condition (1).

Theorem 4.5 (Equivalence of Two Attacks). Conjunction of conditions (13a) AND (13b) is equivalent to condition (1). A Boolean function works in our attack of Thm. 4.1 if and only if it works with the attack of Section 6 in [18].

Proof of Thm. 4.5. Following Lemma 4.3 the first condition is always true for our attack. The second implication is true for the same reason, if we negate exactly three variables f, d, b then the formula (1) which is $(Z+1)(d+a+b+c+1)(a+d+e+f+1)$ remains unchanged due to double negation. Finally the opposite implication holds because $(d+a+b+c+1)*(a+d+e+f+1)$ is simply equal to the sum:

$$(f+e+1)(d+a+1)(b+c+1) + (f+e)(d+a)(b+c)$$

We have two different proofs and two different formal algebra derivations for the exactly same attack. This observation has many interesting consequences.

4.6 Polynomial Algebra vs. Space Partitioning Attacks

Boolean polynomial arithmetic is highly redundant and these attacks could also be simply studied in terms of space partitioning attacks [21] and it will then be easier to see that they are identical. However multivariate polynomials do provide specific **insights** - they make attack more intelligible in the sense that they explain WHY certain attacks work. Having two different factorisations is quite instrumental and actually is used inside our proofs for both results. It is quite common in cryptanalysis with invariants that different attacks would be related to each other with intersections and with situations where one attack implies[3] another but not vice-versa. Here we have an equivalence result.

We are now going to argue that we do not need to "fall back" and study partitioning attacks and drop the redundant formal algebraic approach of this paper. On the contrary, we claim that we need **more** of redundant formal algebraic approach. In Section 12 and Appendix B in [9] and Sections 4 and Section 7 in a new paper [19] we provide an even more redundant attack methodology. We describe a new comprehensive framework for the construction of non-linear attacks on block ciphers. It is based on the study of sets of imperfect cycles on basic polynomials obeying simple rules. Here the redundancy potentially increases and we expect to find even more ways to obtain the same attack. The new methodology is deterministic and prescriptive. It allows us to show the existence of a large number of attacks, in such a way that if an initial set of constraints on polynomials is true, we don't need a proof that the attack works. Thus we gain more insights about **why** a given attack works. Moreover if a certain attack could be constructed in several different inequivalent ways, and if these are not identical, this could increase the probability of success of certain attacks.

Another interesting question is whether our attack of Thm. 4.1 we have could also be obtained in a third way, as just a special case of this new general cyclic framework? Potentially yes, however we do not know this yet. In fact this construction involves multiplying many polynomials together and for this reason it has a **one-way** property. Given a concrete polynomial $\mathcal{P}$ there could be many ways (but also possibly zero ways) to obtain this $\mathcal{P}$ from the general framework of [19].

[3] An elaborate set of examples for this can be found in [17] and Section 10 in [8].

5 On Success Probability of Our Attack

In this section we evaluate the exact probability that a random Boolean function is such that the attack of Thm. 4.1 works when the Boolean function is chosen at random. We have the following result:

Theorem 5.1 (Specific Annihilator Probability). The probability that $(Z+1)*(d+a+b+c+1)(a+d+e+f+1)=0$ when Z is a random Boolean function with 6 variables is exactly 2^{-16}.

This result is not trivial: it potentially depends not only on two linear functions but also potentially on how they interact (repeated variables). In order to show this we need several more basic results. Let $\mathbf{B}_N$ be the ring of all Boolean polynomials with N variables represented by their ANF. We denote by $\text{Ann}(f)$ the set of annihilators of a Boolean function f, and since this is a linear space which also includes 0, let $\text{DimA}(f)$ be the dimension of this space with:

$$|\text{Ann}(f)| = 2^{\text{DimA}(f)}$$

Now it is maybe less obvious that we have:

Theorem 5.2 (Annihilators and Weight).

$$|\text{Ann}(f)| = \frac{|\mathbf{B}_N|}{2^{wt(f)}}$$

where $wt(f)$ denotes the Hamming weight of the Boolean function f, meaning the number of $1's$ in its truth table or, in other words, the cardinality of the support of f. The proof of this result is found in Appendix A.

We study the probability that given a specific Boolean function f, a random Boolean function Z will be in the annihilator set of f. There are several ways to derive our result. the simplest method is to look at the weight of the product $(d+a+b+c+1)(a+d+e+f+1)$, which is $2^N/4$ and applying Thm. 5.2 directly, which gives 2^{-2^4} immediately. However we want to understand how this type of result changes when we multiply factors, and establish convenient notations and tools which can be used to solve similar problems, even in more complex cases. For this reasons another method is studied in detail in Appendix C. A more detailed study showing three methods to establish a similar result at a higher degree equal to 3 can be found in Appendix C of [9]. A fourth method could be through our equivalence result of Thm. 4.5.

6 Applications, Backdoors and Defensive Considerations

In this article the main attack scenario is that the Boolean function is chosen at random or have been chosen by the adversary. One interesting question is could a block cipher be backdoored in a precise sense that $(Z+1)*(d+a+b+c+1)(a+d+e+f+1)=0$ and does it conflict with known cipher design criteria. In Section 5 we compute the probability that the attack of Thm. 4.1 works when the Boolean function is chosen at random. In practice Boolean functions are expected to be very strong, and satisfy multiple stringent requirements such as good non-linearity, correlation immunity, etc. cf. [5,3]. Under these assumptions could the cipher nevertheless be broken due to Thm. 4.1 just because we are not lucky and $(Z+1)*(d+a+b+c+1)(a+d+e+f+1)=0$? With complex combinations of non-linearity assumptions the probability is overall probably different than the result of Thm. 5.1 in Section 5 below.

An interesting question would be to look back at how the Boolean function of T-310 was generated, two precise sets of historical design criteria are given in [13], and to re-evaluate this probability in more detail under these assumptions. Our current understanding is that the Boolean function of T-310 was extremely well chosen and we do not expect it to be weak in any way, not even accidentally[4]

6.1 Applications - Decryption and Key Recovery Attacks

An interesting question is how do we use such properties in decrypting T-310 communications. The answer is quite complex due to the highly secure stream cipher mode in which this cipher operates, cf. [13], where key bits used in encryption are extracted at a very low rate, of less than 1 bits every 127 rounds. We refer to Section 9 in [9] and Section 6 in [10] for an introduction to this topic. Some invariants (not all) introduce pervasive biases made of higher order correlation properties which do not degrade as the number of rounds grows. Other invariants do directly involve some key bits, see for example Section 9.1. in [9] and [15]. This is a complex question which we consider to be out of scope for the present paper which is a proof of concept for a single attack of degree 7. Most likely, the peculiar invariant presented in this paper is not quite the one which we would need.

6.2 Post-scriptum: Prime period and prime degree questions

Why is it interesting to study invariants such as in this paper? Interestingly 7 is a prime and 127 is also a prime. There are several important reasons why we study invariants with a prime period[5]. Unlike the attack of degree 8 from [9], they do not seem to be easily constructed following the structure of our cipher with 4 branches. It is possible they would be in some sense "primitive" rather that not derived from some simpler attacks. Is it possible that an invariant property of period 127 exists? Such a property could be highly relevant due the period 127 being actually used in real-life encryption. A first proof of concept showing that this is actually possible can be found in Section 3 of [18].

[4] Our current experience looking for interesting annihilation events in various cipher S-boxes suggests that Boolean functions which have many zeros inside their Walsh spectrum are going to be vulnerable to this type of annihilations less frequently, than with random Boolean functions. This is related to the study of the Walsh spectrum and k-normal Boolean functions, cf. [6].

[5] The relation between the degree and period in invariant attacks is not straightforward. Many complex attacks become simpler when we increase the period, while some attacks with period k, can be constructed from cycles on simple polynomials with a period of exactly k, cf. Section 5.6 and Fig. 5 in [9]. A non-linear attack of degree k can be constructed by multiplying polynomials lying a cycle for a hypothetical set of transitions with period k - this topic is studied in [19].

7 Conclusion

Block ciphers have been with us since the 1970s and have iterated periodic structure. Extremely few attacks in symmetric cryptanalysis benefit from this periodicity really well and work when the number of rounds is large. A major question is how to discover new round-invariant properties. The space of solutions is double exponential and systematic exploration is not feasible [2]. For some 50 years researchers found very few attacks of this type except recently [8,26]. Following [8] polynomial invariant attacks are feasible when two Boolean polynomials become equal, or the so called Fundamental Equation vanishes. In this article we present a new very specific construction a polynomial invariant attack. Our goal is to make two different products of 7 factors equal (as polynomials). Our construction relies on several algebraic simplification or annihilation/absorption events. In fact such equality with two products would be rather unthinkable if the ring of multivariate polynomials B_n had unique factorisation. Our construction is compared to other similar attacks in [9] and [18] which use the same basic polynomials.

One way to interpret the result in this article is showing how to "backdoor" the Cold War block cipher T-310, making it weak on purpose. However it is also more generally about "proper" cryptanalysis of block ciphers. Each and every property we require about the cipher wiring or the Boolean function happens with a probability which is not too small. Such attacks can therefore happen accidentally, also when the whole cipher specification and the non-linear components are the strongest possible and were not chosen by the attacker.

How good is our result? We do not claim that the probability of 2^{-16} in Thm. 5.1 is very large and we must recognise that a Boolean function with an annihilator of degree 2, is not likely to be admitted as a legitimate choice in a block cipher. This is because a typical function on 6 bits has annihilators of degree 3 but not 2, cf. [7], Therefore the invariant attack described in this article is yet weak. The present attack does not work for any real-life long-term keys (LZS) for T-310 [20]. However if this polynomial invariant $\mathcal{P}$ of degree 7 does not work, another similar property of degree 7 may eventually work. In future works we intend to demonstrate that as the degree of $\mathcal{P}$ increases, the power of our attack increases, and the success probabilities go up. For example the "product attack" of degree 8 described in [9] has a higher success probability. This previous attack in [9] involved two cycles with 4 polynomials closely following natural transitions in a Feistel cipher with 4 branches. This article shows that there exists yet less trivial and more complex periodic polynomial invariant attacks on block ciphers where the polynomial is a particular product of well chosen 7 linear polynomials. A factorisation of a Boolean polynomial is not unique and the same attack can be obtained in more than one way, cf. Section 4.4. This provides valuable insights about why certain attacks work. In Section 6.1 we explain how invariant attacks could potentially be used to decrypt encrypted communications, and why it is potentially important to search for periodic properties with a degree and/or period being a prime number such as 7 or 127.

References

1. Arnaud Bannier, Nicolas Bodin, and Eric Filiol: *Partition-Based Trapdoor Ciphers,* `https://ia.cr/2016/493`.
2. C. Beierle, A. Canteaut, G. Leander, Y. Rotella: *Proving resistance against invariant attacks: how to choose the round constants,* in Crypto 2017, Part II. LNCS, 10402, pp. 647–678, Springer 2017.
3. Joan Boyar, Magnus Find, René Peralta: *Four Measures of Nonlinearity,* In Algorithms and Complexity, CIAC 2013, LNCS 7878, pp. 61-72, Springer,
4. Marco Calderini: *A note on some algebraic trapdoors for block ciphers,* last revised 17 May 2018, `https://arxiv.org/abs/1705.08151`
5. Claude Carlet: *The complexity of Boolean functions from cryptographic viewpoint,* In Complexity of Boolean Functions (Dagstuhl, Germany), Dagstuhl Seminar Proceedings, no. 06111, 2006.
6. Pascale Charpin: *Normal Boolean functions,* Journal of Complexity, vol. 20, Issues 2–3, pp 245–265, 2004.
7. 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`.
8. 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.
9. Nicolas T. Courtois: *Structural Nonlinear Invariant Attacks on T-310: Attacking Arbitrary Boolean Functions,* `https://ia.cr/2018/1242`, revised 12 Sep 2019.
10. 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`
11. Nicolas Courtois: *Feistel Schemes and Bi-Linear Cryptanalysis,* in Crypto 2004, LNCS 3152, pp. 23–40, Springer, 2004.
12. 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.
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`
14. Nicolas Courtois, Maria-Bristena Oprisanu and Klaus Schmeh: *Linear cryptanalysis and block cipher design in East Germany in the 1970s,* in Cryptologia, 2018.
15. Nicolas Courtois, Marios Georgiou and Matteo Scarlata: *Slide attacks and LC-weak keys in T-310,* in Cryptologia, vol 43, Iss. 3, 2019, pp. 175–189.
16. 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.
17. Nicolas T. Courtois: *Invariant Hopping Attacks on Block Ciphers,* presented at WCC'2019, Abbaye de Saint-Jacut de la Mer, France, 31 March - 5 April 2019. Extended version available at `https://arxiv.org/pdf/2002.03212.pdf`, 8 February 2020.
18. Nicolas T. Courtois, Aidan Patrick: *Lack of Unique Factorization as a Tool in Block Cipher Cryptanalysis,* Preprint, `https://arxiv.org/abs/1905.04684` 12 May 2019.

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. Jörg Drobick: *T-310 Schlüsselunterlagen,* a web page which enumerates several different known long-term keys for T-310 from 1973-1990, consulted 21 January 2017, `http://scz.bplaced.net/t310-schluessel.html`
21. Carlo Harpes: *Partitioning Cryptanalysis,* Post-Diploma Thesis, Signal and Information Processing Lab., Swiss Federal Institute of Technology, Zurich, March 1995.
22. 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.
23. Lars R. Knudsen, Matthew J. B. Robshaw: *Non-Linear Characteristics in Linear Cryptoanalysis,* Eurocrypt'96, LNCS 1070, Springer, pp. 224–236, 1996.
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. 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.
27. Yongzhuang Wei, Tao Ye, Wenling Wu, Enes Pasalic: *Generalized Nonlinear Invariant Attack and a New Design Criterion for Round Constants,* In IACR Tr. on Symm. Crypt. Vol. 2018, No. 4, pp. 62-79. `https://tosc.iacr.org/index.php/ToSC/article/view/7361/6531`
28. Zhegalkin polynomial, Wikipedia entry, `https://en.wikipedia.org/wiki/Zhegalkin_polynomial`

A Our Formula For Annihilator Space Cardinality

In this appendix we provide a proof of the formula of Thm. 5.2:

$$|\mathrm{Ann}(f)| = \frac{|\mathbf{B}_N|}{2^{wt(f)}} \tag{12}$$

where $wt(f)$ denotes the Hamming weight of the Boolean function f, i.e. the number of $1's$ in its truth table. We start by observing that:

$$g \in \mathrm{Ann}(f) \Leftrightarrow gf = 0 \Leftrightarrow (g = 0 \ \vee f = 0) \Leftrightarrow (g = 1 \rightarrow f = 0)$$

$$\Leftrightarrow \mathrm{supp}(g) \subseteq \mathrm{supp}(1 + f) \tag{13}$$

We are going to prove the formula by showing a bijection between two finite sets, one with cardinality $|\mathrm{Ann}(f)|$, the other with cardinality equal to $\frac{|\mathbf{B}_N|}{2^{wt(f)}}$. Let's start by defining our map in the general case.

Lemma A.1. Given any subset $K \subseteq \mathbb{F}_2^N$, the following map is a bijection:

$$\begin{aligned} \Lambda : \{g : \mathbb{F}_2^N \rightarrow \mathbb{F}_2 : \mathrm{supp}(g) \subseteq K\} &\rightarrow \{\chi : K \rightarrow \{0, 1\}\} \\ g &\longmapsto \chi_{\mathrm{supp}(g)} \end{aligned}$$

Where the map $\chi_{\mathrm{supp}(g)}$ is the characteristic map of the support of g defined on the set K:

$$\chi_{\mathrm{supp}(g)}(x) = \begin{cases} 1 & x \in \mathrm{supp}(g) \\ 0 & x \notin \mathrm{supp}(g) \end{cases}$$

Proof of Lemma A.1.

1. Injectivity:

$$\Lambda(g_1) = \Lambda(g_2) \Leftrightarrow \chi_{\mathrm{supp}(g_1)} = \chi_{\mathrm{supp}(g_2)} \Leftrightarrow \mathrm{supp}(g_1) = \mathrm{supp}(g_2) \Leftrightarrow g_1 = g_2$$

2. Surjectivity: trivial because every Boolean function coincides with the characteristic function of its support.

□

From this lemma the formula (12) follows directly, when we consider it in the case $K = \mathrm{supp}(1 + f)$. The set on the left has cardinality equal to $|\mathrm{Ann}(f)|$ thanks to (13), while the set on the right has of course cardinality $2^{|\mathrm{supp}(1+f)|}$. Therefore we obtain the result claimed:

$$|\mathrm{Ann}(f)| = 2^{|\mathrm{supp}(1+f)|} = 2^{2^N - |\mathrm{supp}(f)|} = \frac{|\mathbf{B}_N|}{2^{wt(f)}}$$

□

B Supports or Boolean Functions and HW Sum Formula

In this section we show an alternative interpretation of Boolean functions. In what follows denote the power set of a set X by $\mathcal{P}(X)$, which is defined as the set of all subsets of X. The symbol $\triangle$ will denote the symmetric difference between sets, i.e. $A \triangle B = (A \setminus B) \cup (B \setminus A)$ which is the set operation equivalent of logical XOR. The main idea is that $\mathbf{B}_N$ and $\mathcal{P}(\mathbb{F}_2^N)$ are Boolean rings with their own operations which are in a one to one correspondence.

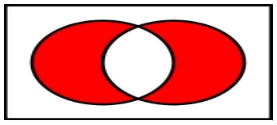
Fig. 5. Venn diagram for the set symmetric difference.

Fact B.1. We have an isomorphism between two Boolean rings $(\mathbf{B}_N, +, *)$ and $(\mathcal{P}(\mathbb{F}_2^N), \triangle, \cap)$ defined as follows with the following four properties:

$$(\mathbf{B}_N, +,*) \longrightarrow (\mathcal{P}(\mathbb{F}_2^N), \triangle, \cap)$$
$$f \longmapsto \mathrm{supp}(f)$$

1. The map is a bijection.
2. $f + g \longmapsto \mathrm{supp}(f) \triangle \mathrm{supp}(g)$
3. $f * g \longmapsto \mathrm{supp}(f) \cap \mathrm{supp}(g)$
4. The inverse map respects the operations as well

We will now use this isomorphism to prove the theorem needed in this paper:

Theorem B.2. Given any two Boolean functions $f, g \in \mathbf{B}_N$, we have the following relation between the Hamming weights:

$$wt(f + g) = wt(f) + wt(g) - 2 \cdot wt(fg)$$

Proof of Thm. B.2. Hamming weight is defined cardinality of the support, therefore due to isomorphism above, our result becomes the well known fact about sets:

$$|A \triangle B| = |A| + |B| - 2|A \cap B|$$

where we take A and B are $\mathrm{supp}(f)$ and $\mathrm{supp}(g)$ respectively. This ends the proof.

□

We also provide an example to show how Thm. B.2 works in practice. We have generated two random Boolean functions and we see how the truth tables for $f, g, f+g$ and fg are related. We see that $wt(f+g) = wt(f)+wt(g)-2\cdot wt(fg)$.

f	0	1	1	0	1	1
g	1	1	0	0	0	1
$f+g$	1	0	1	0	1	0
fg	0	1	0	0	0	1

C Another Method to Derive our Success Probability

We study the probability that given a specific Boolean function f, a random Boolean function Z will be in the annihilator set of f. We assume that the choice of the function Z is uniformly random inside the space of Boolean functions $\mathbf{B}_N$. Then this probability is equal to the relative size of the set of $\mathrm{Ann}(f)$ inside $\mathbf{B}_N$:

Definition C.1. Given a Boolean function f, we define the probability value associated to it as:

$$\mathrm{AP}_f \overset{def}{=} \mathbb{P}\left(Z \text{ Random in } \mathbf{B}_N,\, Z \in \mathrm{Ann}(f)\right) = \frac{|\mathrm{Ann}(f)|}{|\mathbf{B}_N|}$$

C.2 Annihilator Combination Theorem

We recall that due to Thm. 5.2 our probability is always directly related to the Hamming weight of the function f, in particular we see that $\mathrm{AP}_f = (\frac{1}{2})^{wt(f)}$. An interesting question is what happens when f is a product of two Boolean functions like $f = f_1 f_2$, whose decomposition is not unique.

Theorem C.3 (Combination Formula). Given any two Boolean functions f_1 and f_2 we have:

$$\mathrm{AP}_{f_1 f_2} = \sqrt{\frac{\mathrm{AP}_{f_1}\mathrm{AP}_{f_2}}{\mathrm{AP}_{f_1+f_2}}} \tag{14}$$

Proof of Thm. C.3. This formula follows from the following formula which is formally proven in Thm. B.2 in Appendix B:

$$wt(f_1 + f_2) = wt(f_1) + wt(f_2) - 2 \cdot wt(f_1 f_2) \tag{15}$$

Rearranging the formula (15) we can write:

$$wt(f_1 f_2) = \frac{1}{2}\left(wt(f_1) + wt(f_2) - wt(f_1 + f_2)\right) \tag{16}$$

which yields the formula (14) in the following way:

$$\mathrm{AP}_{f_1 f_2} = \left(\frac{1}{2}\right)^{wt(f_1 f_2)} = \left(\frac{1}{2}\right)^{\frac{1}{2}(wt(f_1)+wt(f_2)-wt(f_1+f_2))} = \sqrt{\frac{\mathrm{AP}_{f_1}\mathrm{AP}_{f_2}}{\mathrm{AP}_{f_1+f_2}}}$$

From here we obtain another derivation for our Thm. 5.1. We are going to apply Thm. C.3:

$$p = \mathrm{AP}_{(d+a+b+c+1)(a+d+e+f+1)} = \sqrt{\frac{\mathrm{AP}_{(d+a+b+c+1)}\mathrm{AP}_{(a+d+e+f+1)}}{\mathrm{AP}_{(d+a+b+c+1)+(a+d+e+f+1)}}}$$

Each affine function here is balanced and we have We start by noticing that

$$\mathrm{AP}_{(d+a+b+c+1)} = \left(\frac{1}{2}\right)^{wt(d+a+b+c+1)} = \left(\frac{1}{2}\right)^{2^6 - wt(d+a+b+c)} = \left(\frac{1}{2}\right)^{2^5}$$

Putting everything together we obtain the desired result of Thm. 5.1:

$$p = \sqrt{\frac{\frac{1}{2^{2^5}}\frac{1}{2^{2^5}}}{\frac{1}{2^{2^5}}}} = \sqrt{\frac{1}{2^{2^5}}} = \frac{1}{2^{2^4}} = 2^{-16} \approx 1.53 * 10^{-5}$$