Algebraic Description and Simultaneous Linear Approximations of Addition in Snow 2.0.[⋆]

Nicolas T. Courtois[1] and Blandine Debraize[2,3]

[1] University College of London, Gower Street, London, UK
[2] Gemalto, Meudon, France
[3] University of Versailles, France

Abstract. In this paper we analyse the algebraic properties over the field GF(2) of the addition modulo 2^n. We look at implicit quadratic equations describing this operation, and at probabilistic conditional linear equations. We show that the addition modulo 2^n can be partly or totally linearized when the output is fixed, and this for a large family of outputs. We apply these results to analyse the resistance of the stream cipher Snow 2.0 against algebraic attacks.

Key words: modular addition, multivariate quadratic equations, algebraic immunity, stream ciphers, Snow 2.0, algebraic cryptanalysis

1 Introduction

Many ciphers are based on mixing of S-Boxes, arithmetic and Boolean operations. One of the fastest arithmetic operations is addition modulo a power of 2, that is handled very efficiently on modern processors. We will adopt the notation '$\boxplus$' for this operation in this article, to differentiate it from the "exclusive-or" which we denote '$\oplus$'. The addition modulo 2^n is used in block ciphers such as TwoFish, hash functions such as MD5 and SHA 1 and in many stream ciphers. Among stream ciphers, the most prominent example is Snow 2.0, which is today a reference standardized software-oriented stream cipher.

The algebraic immunity is defined as the minimum degree for which one can write multivariate equations mixing input bits and output bits (see [17]). It is an important design criterion for S-boxes. However this criterion is not sufficient in itself to define the resistance of an S-box against algebraic attacks. Indeed, the modular addition is partly linear, thus it has the same algebraic immunity as the exclusive or. Yet '$\boxplus$' is stronger. For example, an interesting analysis of the security of the stream cipher Snow 2.0 has been given in [5] by Billet and Gilbert. They propose an attack on a modified version of the cipher, where the '$\boxplus$' are replaced by '$\oplus$' that is fully linear. But as '$\boxplus$' cannot be described entirely by linear relations, the attack could not be directly extended to the real cipher.

A generalisation of the notion of algebraic immunity has been studied in [1] and [14]. It is concerned with implicit equations conditioned on the value of the output or a part of the output. It has interesting applications in algebraic cryptanalysis of stream ciphers as shown by Fischer and Meier in [14].

[⋆] Partly supported by the EU Commission via the ECRYPT network of excellence.

Algebraic properties of the addition modulo a power of two have previously been studied in [5, 15, 19]. In [5], a quadratic description over GF(2) implying carry bits is proposed. In this paper we propose a new notion called describing degree to explore the algebraic properties of the '$\boxplus$' in a more refined way. We develop a description of this function as a set of implicit quadratic equations over GF(2) without any additional variable. Then we study the question of how equations can be partially linearized from the point of view of the attacker. For this we use conditioned equations described in [14] to introduce a new method to approximate the addition modulo 2^n, that is the main contribution of this paper. Actually one can view '$\boxplus$' as an S-box in two different ways, and both versions can be partially or totally linearized. One of the interests of these partial linearization techniques is that they can improve considerably the complexities of algebraic attacks, that can potentially be developed also for other ciphers.

One of the propositions of the authors in [5] to extend their method to the real Snow 2.0 is to guess the carries of the modular addition. In this paper we go further in the analysis of the security of this cipher against this type of attack. We implement their proposition, and show that by using our linearization techniques on '$\boxplus$' we obtain better results than by guessing the carries. In this paper we consider KGSnow 2.0, that is the keystream generator part of the cipher, where the initial state of the registers is considered as the key of the cipher. We compare our results to the classical time-memory trade-off attack on KGSnow 2.0.

In Section 2 we recall and define various notions of algebraic immunity, in Section 3 we study algebraic deterministic descriptions of '$\boxplus$', and these can be partially linearized as shown in Section 4. In Section 5 we explain how to handle algebraic cryptanalysis and our basic algorithm. Based on all these, an analysis of KGSnow 2.0 is presented in Section 6.

2 Preliminaries

2.1 Notation

Let us consider three n-bit words $(x_{n-1}, \ldots, x_0)$, $(y_{n-1}, \ldots, y_0)$ and $(z_{n-1}, \ldots, z_0)$ with z_0 being the low-order bit. The modular addition

$$(x, y) \mapsto z = x \boxplus y \mod 2^n$$

is a T-function (see [16]), as each bit z_i of the output only depend on the bits $x_0, \cdots, x_i, y_0, \cdots, y_i$. This T-function can be described the following way by $(*)$ and $(*')$, using new variables that are carry bits, represented by the $(n-1)$-bit word $c = (c_{n-1}, \ldots, c_1)$:

$$(*)\begin{cases} z_0 = x_0 + y_0 \\ z_1 = x_1 + y_1 + c_1 \\ z_2 = x_2 + y_2 + c_2 \\ \vdots \\ z_i = x_i + y_i + c_i \\ \vdots \\ z_{n-1} = x_{n-1} + y_{n-1} + c_{n-1}, \end{cases} \qquad (*')\begin{cases} c_1 = x_0 y_0 \\ c_2 = x_1 y_1 + (x_1 + y_1) c_1 \\ \vdots \\ c_i = x_{i-1} y_{i-1} + (x_{i-1} + y_{i-1}) c_{i-1} \\ \vdots \\ c_{n-1} = x_{n-2} y_{n-2} + (x_{n-2} + y_{n-2}) c_{n-2} \end{cases}$$

2.2 Descriptive Algebraic Representation Criteria for S-boxes

We first recall important notions in algebraic cryptanalysis:

Definition 1. *A system of equations is said to be overdefined if the rank of the system equations is strictly larger than the number of variables.*

Let $S : \{0,1\}^n \rightarrow \{0,1\}^m$ be an S-box.

Definition 2. *An I/O equation for S is a nonzero algebraic equation $r(x, y) = 0$ that holds with probability 1, i.e. for every pair (x, y) such that $S(x) = y$.*

The notion of Algebraic Immunity (also sometimes called Graph Algebraic Immunity or I/O degree) has been introduced by Carlet, Meier and Pasalic [17].

Definition 3. *The algebraic immunity AI is defined by the minimum degree of an I/O equation for S.*

The algebraic immunity of the modular addition is clearly 1, because of the linear equation mixing the least significant bits described at Section 2.1: $z_0 = x_0 + y_0$. The algebraic immunity of the exclusive or is also 1, yet typically $\boxplus$ will be cryptographically much stronger than $\oplus$. We see that the algebraic immunity is not always the best criterion to define the resistance of an S-box against algebraic attacks. Two other important properties of an S-box are:

1. The minimal degree d such that the S-box is entirely defined by equations of degree at most d.
2. The number of such linearly independent equations of degree at most d.

We define a new criterion to describe the first property :

Definition 4. *The minimal degree d such that the S-box is entirely defined by equations of degree at most d is called describing degree (DD) of S.*

The notion of algebraic immunity is based on the existence of an I/O degree equation. But if for a function F some equations of minimal degree d exist, however all these degree d equations may not define the function F. As we have seen, the algebraic immunity of '$\boxplus$' is 1 but as this function is not defined by the only one linear equation $z_0 = x_0 + y_0$, its describing degree is 2.

2.3 Criteria for Conditioned Algebraic Representation of S-boxes

Conditional algebraic I/O equations emerge as an important tool in cryptanalysis of stream ciphers as illustrated by Krause, Armknecht, Fischer and Meier [1, 14].

Definition 5. *Let us assume $n > m$. Given some fixed output y, a y-conditional I/O equation for S is a nonzero algebraic equation $r_y(x) = 0$ that holds with probability 1 for every x such that $S(x) = y$.*

The relevant notion of conditional algebraic immunity is defined by Fischer and Meier [14] as follows:

Definition 6. *Given some fixed output y, let d be the minimum degree of a y-conditional I/O equation. The conditional algebraic immunity CAI of S is the minimum of d over all y in $GF(2)^m$.*

Similarly, we adapt our describing degree criterion:

Definition 7. *Given some fixed output y, let d be the minimum degree such that the equation $S(x) = y$ is entirely defined by conditional I/O equations of degree at most d. The minimal d over all y in $GF(2)^m$ is called conditional describing degree (CDD) of S.*

3 Describing Degree of the Addition Modulo 2^n

In this section, we show that the addition modulo a power of two can be described by quadratic I/O equations over GF(2). We give a "describing" set of quadratic equations and compute many extra equations.

We note that there are several ways to consider $\boxplus$ as an S-box function. If we consider three n-bits words x, y and z, the equation

$$x \boxplus y = z$$

leads to two possible functions $\mathcal{P}$, $\mathcal{M}$ of type $\{0,1\}^{2n} \rightarrow \{0,1\}^n$:

- $\mathcal{P} : (x, y) \mapsto z$
- $\mathcal{M} : (x, z) \mapsto y$ and $\mathcal{M}' : (y, z) \mapsto x$ that is exactly the same function.

3.1 Equations with no Extra Variables

Proposition 1. *The Describing Degree is 2 for both $\mathcal{P}$ and $\mathcal{M}$.*

Proof. Looking at the equation $(*)$ and $(*')$ of Section 2.1, we notice that all the carry bits c_i from $(*')$, can be expressed as linear combinations of other variables using $(*)$, and eliminated. The resulting equations remain quadratic and there is no extra variable at all:

$$(\#) \begin{cases} z_0 = x_0 + y_0 \\ z_1 = x_1 + y_1 + x_0 y_0 \\ z_2 = x_2 + y_2 + x_1 y_1 + (x_1 + y_1)(x_1 + y_1 + z_1) \\ \vdots \\ z_i = x_i + y_i + x_{i-1} y_{i-1} + (x_{i-1} + y_{i-1})(x_{i-1} + y_{i-1} + z_{i-1}) \\ \vdots \\ z_{n-1} = x_{n-1} + y_{n-1} + x_{n-2} y_{n-2} + (x_{n-2} + y_{n-2})(x_{n-2} + y_{n-2} + z_{n-2}) \end{cases}$$

At this stage, these equations entirely define the function $\boxplus$, but are not overdefined (this will change below). Yet they are already quite sparse. Apart from the linear terms, only products of type $A_i B_i$ or $A_i B_{i-1}$ do appear with A being x or y, and with B being x, y or z. The number of terms is only $\mathcal{O}(n)$ instead of $\mathcal{O}(n^2)$ for a generic system of quadratic equations.

3.2 Additional Equations

Many other quadratic equations do exist for the adders modulo 2^n. Their existence can be derived as follows:

- For any n, from (#) we have 1 linear and $n - 1$ quadratic equations.
- Then there are $3n$ additional equations that come from the fact that you can multiply the linear equation by any variable and it becomes quadratic. However it turns out that the dimension of the vector space spanned by these equations is only $3n - 1$ because of the following linear dependency: $(z_0 + x_0 + y_0) = z_0(z_0 + x_0 + y_0) + x_0(z_0 + x_0 + y_0) + y_0(z_0 + x_0 + y_0)$
- We also have 2 equations that come from the fact that you can multiply $z_1 = x_1 + y_1 + x_0 y_0$ by x_0 or by y_0.

– Then for each of the $n-2$ remaining equations, one gets 3 additional quadratic equations. This is because the equation:

$$\begin{aligned} z_i &= x_i + y_i + x_{i-1}y_{i-1} + (x_{i-1} + y_{i-1})(x_{i-1} + y_{i-1} + z_{i-1}) \\ &= x_i + y_i + x_{i-1} + y_{i-1} + x_{i-1}y_{i-1} + x_{i-1}z_{i-1} + y_{i-1}z_{i-1} \end{aligned}$$

can be multiplied by $(x_{i-1}+y_{i-1})$, by $(x_{i-1}+z_{i-1})$ and also by $(y_{i-1}+z_{i-1})$. These three new equations are linearly dependent and their rank is two. Thus we get $2(n-2)$ additional equations.

In the extended version of this paper, we prove that all the $6n-3$ quadratic equations described above are linearly independent.

4 Conditional Linear Equations for '$\boxplus$'

By fixing z for $\mathcal{P}$, and y for $\mathcal{M}$, we obtain conditional equations of degree at most 2, with at least 2 linear equations. In some cases, when the $n-2$ least significant bits of z are 1 for $\mathcal{P}$, and when the $n-2$ least significant bits of y are 0 for $\mathcal{M}$, it can be completely linearized. That is what we show in section 4.1.

At section 4.2, we refine this result by showing that for both $\mathcal{P}$ and $\mathcal{M}$, if the output contain r consecutive bits of the same value 0 or 1 (whatever the value it is for both $\mathcal{P}$ and $\mathcal{M}$), we obtain a set of $r+2$ linear equations that are *simultaneously* true with a certain probability.

4.1 Conditional Describing Degree of '$\boxplus$'

Proposition 2. *The Conditional Describing Degree is 1 for both $\mathcal{P}$ and $\mathcal{M}$.*

Proof. For $\mathcal{M}$, this result is straightforward: we put $y = 0$, and the relation becomes completely linear as we have $x = z$.

For $\mathcal{P}$, we put $z = 2^n - 1$, and we see that the first carry $c_1 = 0$, because $x_0 \oplus y_0 = 1 \Rightarrow x_0 \boxplus y_0 < 2$. Then we prove recursively that all the carries of this function are zero and that: $\forall i \; x_i \oplus y_i = 1$. Finally, these equations clearly describe exactly all possible input values that lead to the chosen fixed output for this S-box (for both $\mathcal{M}$ and $\mathcal{P}$).

In fact $y = 0$ is not the only value for y such that the relation $\mathcal{M}(x, z) = y$ can be entirely described by linear equations. Our simulations showed that exactly 4 values of y have this property: the values for which the $n-2$ least significant bits of y are zero. This can be proven as follows : as the carry c_{n-2} is 0, the equation of the second most significant bit of the modular addition is $z_{n-2} = x_{n-2} \oplus y_{n-2}$. If $y_{n-2} = 0$, the same result holds for the most significant bit equation: $z_{n-1} = x_{n-1} \oplus y_{n-1}$. If $y_{n-2} = 1$, we have: $c_{n-1} = \lfloor \frac{x_{n-2}+y_{n-2}+c_{n-2}}{2} \rfloor = \lfloor \frac{x_{n-2}+1}{2} \rfloor = x_{n-2}$. Then the most significant bit equation is $z_{n-1} = x_{n-1} \oplus y_{n-1} \oplus x_{n-2}$.

The same way, $z = 2^n - 1$ is not the only value for z such that $\mathcal{P}(x, y) = z$ can be fully described by linear equations. Our simulations showed that the 4 values of z such that its $n-2$ least significant bits are 1 have the same property. This can be proven in a similar way as for $\mathcal{M}$.

When we fix the output, the number of linear equations describing $\mathcal{P}$ and $\mathcal{M}$ is at least 2 and in many cases it is more than 2. One can observe that (this point will be developed later) :

– For $\mathcal{P}$, this depends on the number of consecutive 1 in the least significant bits of the binary representation of z. If there are r consecutive 1 and $r \leq n-2$, the number of linear equations is $r+2$.
– For $\mathcal{M}$, this depends on the number of consecutive 0 in the least significant bits of the binary representation of y. If there are r consecutive 0 and $r \leq n-2$, the number of linear equations is $r+2$.

4.2 Probabilistic Conditional Properties of '$\boxplus$'

We will now give two general theorems on the number of probabilistic conditional equations for $\mathcal{P}$ and $\mathcal{M}$. Let $S : \{0,1\}^n \to \{0,1\}^m$ be an S-box.

Definition 8. *A set $\mathcal{E}$ of equations is said to be p-probable for S if the probability that all equations in $\mathcal{E}$ are* simultaneously *true, taken over the set of all pairs (x, y) such that $S(x) = y$, is equal to p.*

Theorem 1. *Let z be a fixed output for $\mathcal{P}$.*

– *If z has r consecutive 1s from the bit $i \geq 0$ to the bit $i+r-1 \leq n-1$ in its binary representation, then there is a p-probable set $\mathcal{E}$ of $r+2$ (only $r+1$ if $i+r-1 = n-2$, and r if $i+r-1 = n-1$) linear equations for $\mathcal{P}$, with $p = \frac{1}{2} + \frac{1}{2^{i+1}}$.*
– *If z has r consecutive 0s from the bit 0 to the bit $r-1 \leq n-1$ in its binary representation, then there is a p-probable set $\mathcal{E}$ of $r+2$ (only $r+1$ if $r-1 = n-2$, and r if $r-1 = n-1$) linear equations for $\mathcal{P}$, with $p = \frac{1}{2}$.*
– *If z has r consecutive 0s from the bit $i \geq 0$ to the bit $i+r-1 \leq n-1$ in its binary representation, then there is a p-probable set $\mathcal{E}$ of $r+2$ (only $r+1$ if $i+r-1 = n-2$, and r if $i+r-1 = n-1$) linear equations for $\mathcal{P}$, with $p = \frac{1}{2} - \frac{1}{2^{i+1}}$.*

A symmetrical result holds for $\mathcal{M}$ when replacing 0s by 1s:

Theorem 2. *Let y be a fixed output for $\mathcal{M}$.*

– *If y has r consecutive 0s from the bit i to the bit $i+r-1 \leq n-1$ in its binary representation, then there is a p-probable set $\mathcal{E}$ of $r+2$ (only $r+1$ if $i+r-1 = n-2$, and r if $i+r-1 = n-1$) linear equations for $\mathcal{M}$, with $p = \frac{1}{2} + \frac{1}{2^{i+1}}$.*
– *If y has r consecutive 1s from the bit 0 to the bit $r-1 \leq n-1$ in its binary representation, then there is a p-probable set $\mathcal{E}$ of $r+2$ (only $r+1$ if $r-1 = n-2$, and r if $r-1 = n-1$) linear equations for $\mathcal{M}$, with $p = \frac{1}{2}$.*
– *If y has r consecutive 1s from the bit $i \geq 0$ to the bit $i+r-1 \leq n-1$ in its binary representation, then there is a p-probable set $\mathcal{E}$ of $r+2$ (only $r+1$ if $i+r-1 = n-2$, and r if $i+r-1 = n-1$) linear equations for $\mathcal{M}$, with $p = \frac{1}{2} - \frac{1}{2^{i+1}}$.*

For proofs of Theorem 1 and 2 we refer to Appendix A.

Remark. In both theorems 1 and 2, the assertions are true with probability 1 if we fix another constraint. For assertion 1 and 3, this constraint consists in fixing the carry c_i to 0. For assertion 2 of Theorem 1, it consists in fixing x_0 to 0, and for assertion 2 of Theorem 2 , in fixing x_0 to 1 (this also implies that the first carry bit c_1 is 0). This can be seen in the proof in Appendix A. A randomly chosen linear equation is true for $\mathcal{M}$ or $\mathcal{P}$ with probability close or equal to $\frac{1}{2}$. The interest of our theorems is that they allow to obtain r, $r+1$ or $r+2$ linear equations that are *simultaneously* true with probability close to $\frac{1}{2}$. Thus in cryptanalysis, one is able to, if certain constraints are added, to partially linearize the modular addition. Adding a large number of linear equations to a quadratic system allows to eliminate many variables, and is expected to make it in general easier to solve, at least for known Gröbner bases algorithms. This concept of simultaneous linear approximations is very different from Linear Cryptanalysis with multiple characteristics, and to the best of our knowledge has not been studied before.

5 Algebraic Cryptanalysis and Application to KGSnow

5.1 Overview of Algebraic Cryptanalysis

Algebraic cryptanalysis is usually made of two stages:

1. **Writing the equations** describing the problem of the recovery of the key. This stage determines the complexity of the second stage.
2. **Solving the polynomial system**. The most usual family of algorithm for solving polynomial systems is the XL and Gröbner bases family. XL ([7]) and Gröbner bases algorithms like F4 ([12]) and F5 ([13]) are based on the same idea of *expansion* of the system followed by an *elimination* step.
   - During the *expansion* the equations are multiplied by monomials of a chosen degree.
   - The *elimination* is a Gaussian elimination applied to the expanded system where each monomial is considered as a variable.

   This principle is applied once in XL whereas it is applied several times for F4 and F5 with additional clever tricks to decrease the number of equations in the Gaussian elimination.

The theoretic bound for the complexity of XL,F4 and F5 algorithm depends on the maximal degree d of the polynomials manipulated during the computation of the algorithm. The most important cost in these algorithms is the complexity of the (most costly step) Gaussian elimination, that is bounded by $\mathcal{M}^3$, where $\mathcal{M} = 1 + n + \binom{n}{2} + ... + \binom{n}{d}$ is the number of monomials of degree less than d (n being the number of variables).

Some work has been done to compute the value of this degree for a given system by Diem in [10] and Bardet, Faugère and Salvy in [4]. But it is not relevant for non random systems such as the systems derived from cryptographic systems. What is well-known and proved by [4] is that the more overdefined the system is, the lower this degree is. Thus, experimentation plays still an important role in evaluating which overdefined systems of equations derived from cryptographic systems can be solved and how.

5.2 Description of ElimLin and Simulations on KGSnow 2.0

In our experiments on KGSnow 2.0. we are heavily limited by the computational power available and in this paper we limit the maximal degree for the polynomials used during our computations to a very small value that is 2. We handle all our computations with a very simple algorithm for solving polynomial systems over GF(2) that is called ElimLin. A high-level description of ElimLin is given in Algorithm 5.1. It is hard to make a fast and memory efficient implementation of this algorithm, and serious research is needed about how to handle sparse Gaussian elimination and how do we store equations in memory in ElimLin. It appears that (with our current version already) we do not obtain better results with the F4 version of the computer algebra system MAGMA (see [21]) than with the simple ElimLin.

Algorithm 5.1 ElimLin algorithm

```
INPUT: a system $\mathcal{S}$ of GF(2)-equations $\{p_1, ..., p_m\}$ describing an ideal $\mathcal{I}$
Apply a total order on the monomials of $\mathcal{S}$
$\mathcal{S} \longleftarrow Gaussian\quad elimination(\mathcal{S})$
$L \longleftarrow$Number of linear equations in $\mathcal{S}$
while $L > 0$:
        for $i = 1$ to L:
                $v \longleftarrow$ greatest variable of the linear equation $l_i$
                $l'_i \longleftarrow l_i \oplus v$
                Substitute $v$ by $l'_i$ in all the equations of $\mathcal{S}$ except from $l_i$
        Apply a total order on the monomials of $\mathcal{S}$
        $\mathcal{S} \longleftarrow Gaussian\quad elimination(\mathcal{S})$
        $L' \longleftarrow$ Number of linear equations in $\mathcal{S}$
        $L \longleftarrow L - L'$
return $\mathcal{S}$
```

ElimLin in Cryptanalysis of KGSnow 2.0

In our simulations on KGSnow 2.0 we write the I/O equations describing the update of the LFSR and FSM, and the output of the cipher occurring for some consecutive clocks (from 11 to 18 consecutive clocks). The equation describing the addition modulo 2^{32} are those described at section 3. The equations describing the 32 bits S-box are directly derived from the 39 I/O quadratic equations describing the AES S-box (see [8]). We fix some bits belonging to the states of the FSM, and apply ElimLin on this system.

We consider here that brute force is the exhaustive search of the LFSR and FSM initial states (576 bits). If we fix all but a key bits, an attack will be faster than brute force if the running time is less than $2^a E$, where E is the time to check one potential possibility for the initial state. Given a sufficient number of output bits, heuristically about $|LFSR| + [R1| + |R2| + \varepsilon$, (as in [2]), this system has a unique solution that gives these 'key' bits. Exact figures are hard to evaluate because they depend on an optimised implementation of the cipher. Here we will assume that one encryption takes 300 CPU clocks and that the CPU runs at 3 GHz. Then $E \approx 2^{-35}$ hours. Thus, if we fix all key bits except 35, an attack done in less than 1 hour on a PC will be faster than brute force.

If we fix all except 40 key bits, any attack done in less than (approximately) 1 day, will be faster than the exhaustive search of the initial LFSR and FSM bits.

6 Analysis of Snow 2.0 and KGSnow 2.0

KGSnow 2.0 is the keystream generator part of Snow 2.0. In this part we give a short description of Snow 2.0 and recall the analysis of the security of this cipher given in [5] and propose new ways to investigate this security by studying KGSnow 2.0 and using our results of sections 3 and 4.

6.1 Description of Snow 2.0

Snow 2.0 is a reference standardized software-oriented stream cipher. It is believed quite secure and is quite fast: less than 6 CPU cycles per byte on a PC, which is roughly about 3 times faster than RC4 and 4 times faster than AES, cf. [20]. We give a brief description of Snow 2.0; see [11] for details. Snow 2.0 stream cipher is based on one linear feedback shift register made of 16 elements from $GF(2^{32})$ (that can also be seen as a binary LFSR with 512 bits), and a finite state machine composed of 2 states of 32 bits each, nonlinearly clocked.

The output of the FSM is given by the equation:

$$z_t \oplus s_t = (s_{t+15} \boxplus R1_t) \oplus R2_t \tag{1}$$

and the update of the FSM is given by:

$$R1_{t+1} = s_{t+5} \boxplus R2_t \tag{2}$$

$$R2_{t+1} = S(R1_t), \tag{3}$$

where S represents the S-box. This 32 bits S-box is made of four parallel Rijndael S-boxes followed by the MixColumn operation (see [9]). The Rijndael S-box can be described by 39 I/O quadratic equations, see [8]. The MixColumn transformation is GF(2) linear then the entire S-box can be described by 156 quadratic I/O equations. The length of the key is 128 or 256 bits and the initialisation of the key is nonlinear. Our contribution, in section 6.3 essentially consists in analysing KGSnow 2.0.

KGSnow 2.0. has the same design as Snow 2.0, but we ignore the key and IV setup, this meaning that we consider that the key of KGSnow 2.0. is the initial state of the registers when the first keystream bits are produced. This is because with algebraic attacks, the values we are looking for are the values of these registers as they are the solution of the systems of equations, and especially because we do not even know today concerning the security of Snow 2.0 against algebraic attacks if these attacks are able to recover this state faster than its exhaustive search.

6.2 Previous Work

The best known attacks on Snow 2.0 are distinguishing attacks. In [18], it is shown that it possible to distinguish an output keystream of Snow 2.0 of length

2^{174} words from a truly random sequence with workload 2^{174}. No key recovery attack on this cipher have been found so far.

In [5], Billet and Gilbert analyse the security of the cipher by replacing the addition modulo 2^{32} by '$\oplus$'. It is then possible to break the modified cipher by linearization with a complexity of 2^{51}. They use the fact that the describing degree of the S-box is 2, added to the fact that with the replacement of the '$\oplus$' by '$\boxplus$', it is possible to eliminate all the FSM memory bits except the initial ones. They proposed two ways to exploit this result for the real Snow 2.0.:

– The first one consists in guessing the carries of the addition modulo 2^{32}, or to look for the most probable case, that is when all the carries are 0. We have implemented this approach by fixing some carries in the system of equations. But by applying our algorithm (see section 5.2) on such systems we could never recover the initial state fast enough to perform an attack more efficient than the exhaustive search. In section 6.3 we improve this method by using our results of section 4.1.
– The second one consists in introducing the carry bits of the two modular additions '$\boxplus$' at each clock. This allows to build a system of quadratic equations describing the initial state. But in this case it is not possible anymore to totally linearize the system of equations, the attacker has to apply a polynomial system solving algorithm like Gröbner bases algorithms to recover the initial state. We give an overview of this kind of algorithm and their complexities in Section 5. This second approach can be much improved by our analysis of the modular addition proposed in part 3. Indeed, as we explain in Section 5, the more overdefined the polynomial set is, the better the complexity of solving the system becomes. The equations produced by our method implies exactly the same number of variables and provides $189n$ quadratic equations for each $\boxplus$ instead of $31n$. Another advantage comes from the fact that these equations are very sparse.

As the theory is very poor concerning the complexity of algorithms like Gröbner Bases algorithms, the effectiveness of this kind of attacks remains extremely unclear. An interesting question that has not been answered so far concerning this type of attack is the following: is an algebraic attack able to break KGSnow 2.0? We show at section 6.3 that by using the algebraic properties of '$\boxplus$', the answer is yes.

6.3 Towards an Optimal Linearizing Attack

In [5], the authors propose to linearize the $\boxplus$ by guessing the carries. We fixed $17 \times 31 + 17 \times 31$ carries of consecutive clocks and applied ElimLin on the system of the linearized equations coming from the 34 $\boxplus$ and the quadratic I/O equations describing the 17 uses of the S-box. We suppose that when the carries are fixed, the solving part behaves the same way as if all the carries were zero. The probability for this event to happen for 16 consecutive clocks, as proposed in [5], is : $(\frac{3}{4})^{31*17}(\frac{2}{3})^{31*15} \simeq 2^{-497}$. (The approximation of this probability in [5] is too large, as all the random variables are not independent). ElimLin had not finished its computations after 80 hours. Then this attack is not faster than the exhaustive search of the initial state.

We show in this part that guessing the consecutive values of the register R1 and using the properties of '$\boxplus$' described at section 4 seems to be a much better strategy. This is our contribution. We use here the word 'linearizing' to differentiate from linearization attacks where the term linearization means considering each monomial as a variable. Here the concept is completely different, as we do not increase the number of variables.

First Attack We observe that in the design of KGSnow 2.0 (this coming from the design of Snow 2.0) , the fact that for a given t, $R2_{t+1}$ only depends on $R1_t$: by guessing $R1_t$, one gets immediately $R2_{t+1}$. Then, by guessing several consecutive values of $R1$, we obtain also several consecutive values of $R2$. With this method we still do not completely linearize the equations: each time the value of $R1_t$ is known, by equation 1 we just know that we get at least 2 linear relations, the other ones remaining quadratic. But each time the values of $R2_t$ and $R1_{t+1}$ are simultaneously known, we obtain the values of 32 bits of the internal state. The known values and linear relations between the internal state bits are also linear relations between the bits of the initial state of the LFSR as each bit of the internal sequence can be expressed as a linear expression of the initial LFSR state bits. The same way, the quadratic relations between the internal state bits means that we have quadratic equations between bits of the initial state of the LFSR as a composition $f \circ l$ where f is a degree 2 boolean function and l a linear boolean function is still a degree 2 function.

If we guess 10 consecutive $R1s$ (320 bits to guess), we obtain an overdefined system of linear equations and quadratic equations implying only the bits of the 512 initial state bits of the LFSR. These equations come from 17 additions modulo 2^{32}. As each '$\boxplus$' provides an amount of information of 32 bits, the information provided by this quadratic and linear boolean function is enough to recover the 512 initial state bits of the LFSR.

By applying any Gröbner bases algorithm on this system of equations, we obtain the initial state bits. No method is known to compute compute a theoretical complexity for this multivariate polynomial system solving part. Under the hypothesis that this system can be solved at degree 2, a theoretical complexity would be $\mathcal{O}(2^{51})$. In practice we could solve such systems with an algorithm called ElimLin described in Section 5.2 in 2.4 minutes on a PC, which is much faster than the theoretical complexity.

Even if this method seems much better than the guess of the carries, it does not completely linearize the equations. Actually we are able to do it by using the conditional properties of '$\boxplus$' described at Section 4.2.

Improvement of the Attack In this part the idea is to go through the keystream to look for the most interesting case instead of guessing the information. We will use the facts described in theorem 1 and theorem 2 that $11...11 = 2^{32} - 1$ being an output of $\mathcal{P}$ and $\mathcal{M}$ simultaneously linearize both operations with almost the same probability, (as $11...11 \boxplus 1 = 00...00 \mod 2^{32}$), and the trivial fact that 0 being an output of $\mathcal{P}$ linearizes the operation.

We look for the case when $R1_1 = 0, R1_2 = 2^{32} - 1, R1_3 = 0, R1_4 = 0, R1_5 = 0, R1_6 = 0, R1_7 = 0, R1_8 = 0, R1_9 = 0$ by going through the keystream. These constraints on the $R1_is$ are essentially constraints on the internal sequence: we need that

- $s_5 = -R2_0 \mod 2^{32}$,
- $s_6 = -1 - R2_1 \mod 2^{32}$,
- $s_7 = -S(0) \mod 2^{32}$,
- $s_8 = -S(2^{32}-1) \mod 2^{32}$,
- for $9 \leq i \leq 13$, $s_i = -S(0) \mod 2^{32}$.

As the LFSR is clocked by a primitive feedback polynomial, the period of the internal sequence is $2^{512}-1$ and the probability for this constraint to happen is 2^{-288}. We know that we are at the right place on the keystream when the system of equations is solved and provides the right initial state of the LFSR.

Let us suppose these constraints are verified. The same way as the previous attack, we obtain 224 GF(2) linear relations implying only initial LFSR state bits by equation (2). Each time $R1$ is 0, from clock 3 to 9, we obtain 32 GF(2) linear relations in the initial LFSR state bits by equation (1), that is an amount of 224 linear equations. By equation (2) at clock 1 we have:

$$s_6 \boxplus R2_1 = 111\cdots1.$$

Then from the proof of proposition 4.1 we obtain :

$$R2_1 = s_6 \oplus 111\cdots1.$$

By replacing $R2_1$ by $s_6 \oplus 111\cdots1$ and $R1_1$ by 0 in equation (1), we obtain 32 new GF(2) linear relations in the initial LFSR state bits. Finally by using Theorem 2 (Section 4) for equation (1) at clock 2, we obtain 32 linear relations in the LFSR initial state bits with probability $\frac{1}{2}$. To obtain a probability 1, it is enough to add another constraint on the internal sequence: we need that the least significant bit of $s_2 \oplus z_2 \oplus S(0)$ is zero. This is because we have by equation (1) at step 2:

$$s_2 \oplus z_2 \oplus S(0) = s_{17} \boxplus (2^{32}-1),$$

then, as in proof of theorem 1, we obtain:

$$(s_2 \oplus z_2 \oplus S(0)) \boxplus 1 = s_{17}.$$

If the least significant bit of $s_2 \oplus z_2 \oplus S(0)$ is zero, there is no carry with probability 1 in this modular addition. As z_2 is known, it is a constraint on the least significant bit of s_2.

The total number of linear equations is 512. If we assume that these equations are linearly independent, the system can be solved by a simple Gaussian reduction. But this Gaussian reduction can be performed as a precomputation step: during the precomputation the keystream bits are not known and become variables. We then obtain each initial LFSR state bit as a linear combination of the 9×32 keystream bits variables that are used in the 512 equations described above. Then the final step consists in replacing the keystream variables by their real values and verifying that the LFSR initial state bits are correct. This would give a final time complexity of about 2^{288}.

If the rank of the system is $r < 512$, we precompute r initial state bits as linear boolean functions of the keystream variables and the last $512-r$ initial state variables. The complexity is then multiplied by 2^{512-r} as we have to guess the last $512-r$ variables. Actually our simulations showed that this rank is 504.

We show in Appendix B, that by using theorem 1, it is possible to compute at least two more linear equations in the LFSR initial state bits. The final time complexity of this attack is then at most 2^{294}.

The drawback of this method compared to the first one we have proposed is of course the huge amount of necessary keystream bits, about $2^{288} \times 32 = 2^{293}$. This amount can be reduced because, as we explain in section 4, fixing only the 30 least significant bits of the 9 states $R1_t$ have the same linearizing properties as fixing all the bits. We just guess the 2 most significant bits of $R1_1, \cdots, R1_9$ instead of looking for their right value. We store a 2^{18} entries lookup table in which we set the values of the key depending on the keystream variables for each value of the 18 bits coming from the 2 most significant bits of each $R1_1, \cdots, R1_9$. The time complexity is the same but the keystream requirements are lower, about 2^{275} bits. The space complexity is quite small, (about 16 Gb to store the table).

In [3], a time-memory trade-off is proposed, with $TM = N$ and $D = T$, where T is the time complexity, M the memory, D the data and N the number of possible states. We can compare our method with this attack on KGSnow 2.0 with the precise parameters of our method. We show in Table 1 that we obtain better results with our attack. In [6] a better time-memory-data trade-off is proposed: $N^2 = TM^2D^2$, with the following constraint: $D^2 \leq T$. Because of this constraint, in principle it is impossible to compare this attack with ours.

Table 1. Our attack on KGSnow 2.0 vs. general time-memory trade-off

	time	memory	keystream
Babbage time-memory trade-off	2^{302}	2^{295}	2^{287}
Our best attack	2^{294}	2^{37}	2^{275}

Remark We have made some computations on KGSnow 2.0 to try to improve this attack by guessing fewer variables. We have written the system of equations describing 9 consecutive clocks of KGSnow 2.0. In this system we fixed all the bits of $R1_1$, $R1_3$, $R1_4$, $R1_5$, $R1_6$, $R1_7$, $R1_8$, $R1_9$ to 0 except from the most significant bits, and the five most significant bits of $R1_9$, and we fixed the 31 least significant bits of $R1_2$ to 1. We were able to recover the initial LFSR state bits in 2.08 hours. The total complexity is here 2^{311} (see Section 5.2 for details on the computation of the total complexity). We could not obtain a better complexity than 2^{311} by fixing less variables, this meaning that we were not able to improve the complexity of the theoretical attack described above.

7 Conclusion

In this paper we study multivariate linear and quadratic properties over GF(2) of the addition modulo 2^n. We propose a new method for describing this operation as an overdefined system of implicit boolean equations of degree 2. We also introduce the concept of multiple and simultaneous linear approximations. This concept is different from Linear Cryptanalysis with multiple characteristics: we show how to partially or completely linearize the boolean equations describing this function by setting appropriate specific constraints on the output or/and one of the inputs.

These properties can be used to design conditional linearizing attacks on ciphers that use additions modulo 2^n. We propose an example of application in cryptanalysis of KGSnow 2.0, the keystream generator part of Snow 2.0. Given the specific structure of KGSnow 2.0, we have found a combination of constraints, such that the number of linear equations obtained is large compared to their cost. This allows us to recover the key of KGSnow 2.0 within 2^{294} operations. This is not much more than the exhaustive search of the 256 bits key of Snow 2.0 compared to what we could have expected from an extension of the Billet-Gilbert attack of [5] on the real cipher, or more generally to what we could have expected from an algebraic attack on this type of cipher. It is also more efficient than the classical time-memory trade-off attack $TM = N$. This shows that the key of Snow 2.0 should not be longer than in the current specification.

Acknowledgments: It is clear that the addition modulo 2^n can be described by a system of quadratic equations with extra variables (carries), see [5]. However the idea that this can also be achieved without introducing any extra variable is not trivial and was owe it to Josef Pieprzyk and (independently) Philip Hawkes.

References

1. Frederik Armknecht, Matthias Krause: *Constructing Single- and Multi-output Boolean Functions with Maximal Algebraic Immunity.* ICALP (2) 2006: Springer LNCS 4052, pp. 180-191
2. Gwenolé Ars, Jean-Charles Faugère: *An Algebraic Cryptanalysis of Nonlinear Filter Generators using Gröbner Bases,* INRIA research report, available at `https://hal.ccsd.cnrs.fr/`.
3. S. Babbage. *A Space/Time Tradeoff in Exhaustive Search Attacks on Stream Ciphers* European Convention on Security and Detection, IEE Conference Publication No. 408, 1995.
4. M. Bardet, J-C. Faugère and B. Salvy, *On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations,* in Proc. International Conference on Polynomial System Solving (ICPSS,Paris,France), pp.71-75.
5. Olivier Billet, Henri Gilbert: *Resistance of Snow 2.0 against Algebraic Attacks.* CT-RSA 2005, Springer LNCS 3376, pp. 19-28.
6. Alex Biryukov, Adi Shamir. *Cryptanalytic Time/Memory/Data Tradeoffs for Stream Ciphers.* ASIACRYPT 2000: Springer LNCS 1975, pp. 1-13.
7. Nicolas T. Courtois, Adi Shamir, Jacques Patarin, Alexander Klimov: *Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations.* Eurocrypt 2000, Springer LNCS 1807, pp. 392-407.
8. Nicolas Courtois and Josef Pieprzyk: *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations,* Asiacrypt 2002, Springer LNCS 2501, pp. 267-287.
9. J. Daemen, V. Rijmen, *The Block Cipher Rijndael* , Smart Card Research and Applications, Springer LNCS 1820, pp 277-284.
10. Claus Diem, *The XL-algorithm and a conjecture from commutative algebra,* ASIACRYPT 2004, Springer LNCS 3329, pp 323-337.
11. Patrik Ekdahl, Thomas Johansson, *A new version of the stream cipher SNOW,* SAC 2002, Springer LNCS 2595, pp. 47-61. Available from `http://www.it.lth.se/cryptology/snow/`
12. Jean-Charles Faugère: *A new efficient algorithm for computing Gröbner bases (F_4),* Journal of Pure and Applied Algebra 139 (1999) pp. 61-88. See `www.elsevier.com/locate/jpaa`

13. Jean-Charles Faugère: *A new efficient algorithm for computing Gröbner bases without reduction to zero (F5),* Workshop on Applications of Commutative Algebra, Catania, Italy, 3-6 April 2002, ACM Press.
14. Simon Fischer and Willi Meier: *Algebraic Immunity of S-boxes and Augmented Functions,* FSE 2007, Springer LNCS 4593, pp 366-381.
15. Louis Goubin: *A Sound Method for Switching between Boolean and Arithmetic Masking.* CHES 2001: Springer LNCS 2162, pp. 3-15.
16. Alexander Klimov, Adi Shamir: *A New Class of Invertible Mappings.* CHES 2002: Springer LNCS 2523, pp 470-483.
17. Willi Meier, Enes Pasalic, Claude Carlet: *Algebraic Attacks and Decomposition of Boolean Functions.* In Eurocrypt 2004, Springer LNCS 3027, pp. 474-491
18. Kaisa Nyberg, Johan Wallén: *Improved Linear Distinguishers for SNOW 2.0.* FSE 2006: Springer LNCS 4047, pp 144-162.
19. Joseph H. Silverman, Nigel P. Smart, Frederik Vercauteren: *An Algebraic Approach to NTRU (q = 2n) via Witt Vectors and Overdetermined Systems of Nonlinear Equations.*SCN 2004: Springer LNCS 3352, pp 278-293.
20. Results of ESTREAM project benchmarks of ESTREAM stream ciphers compared to AES-CTR, RC4 and Snow 2.0, available at `http://www.ecrypt.eu.org/stream/perf/#results`
21. MAGMA, High performance software for Algebra, Number Theory, and Geometry, — a large commercial software package: `http://magma.maths.usyd.edu.au/`

A Proofs of Theorem 1 and 2

Proof (Proof of Theorem 1.). We first prove by induction on $i \geq 1$ that, the carry bit c_i in the addition $x \boxplus y = z$ (see section 2.1 for notations) is equal to 0 with probability $\frac{1}{2} + \frac{1}{2^{i+1}}$. This assertion is true when $i = 0$, as we always have $c_0 = 0$. Let us call p_i the probability that the carry bit $c_i = 0$. We have: $c_i = \lfloor \frac{x_{i-1}+y_{i-1}+c_{i-1}}{2} \rfloor$ (the condition on z does not affect the bits at positions $0 \dots i-1$ that are random and independent bits). If c_{i-1} is 0, then p_i has a probability $\frac{3}{4}$ to be 0. If c_{i-1} is 1, then p_i has a probability $\frac{1}{4}$ to be 1. Thus we have :

$$p_i = \frac{3}{4} \times p_{i-1} + \frac{1}{4} \times (1 - p_{i-1}) = \frac{1}{2} p_{i-1} + \frac{1}{4} \tag{4}$$

Now if the assertion is true at rank $i-1$, by using equation (4), we compute $p_i = \frac{1}{4} + \frac{1}{2 \cdot 2^i} + \frac{1}{4} = \frac{1}{2} + \frac{1}{2^{i+1}}$, i.e. the assertion is true at level i.

If $i > 0$ and the carry c_i of the addition $x \boxplus y = z$ is 0, and $z_i = 1, ..., z_{i+r} = 1$, we show that we obtain $r+2$ linear I/O equations for $\mathcal{P}$ if $i + r - 1 \leq n - 3$, $r + 1$ linear equations if $i + r - 1 = n - 2$ and r if $i + r - 1 = n - 1$ in the same way as in the proof of Proposition 4.1, namely we simply have $x_i \oplus y_i = 1$ because $c_i = 0$, and then successively we can show that $x_{i+1} \oplus y_{i+1} = 1$ which in turn implies $c_{i+1} = 0$, which in turn gives $x_{i+2} \oplus y_{i+2} = 1$, then $c_{i+2} = 0$ etc. This gives $r+1$ equations as it goes up to the position $i+r$, where we have $x_{i+r} \oplus y_{i+r} = z_{i+r}$. Moreover, the carry bit $c_{i+r+1} = \lfloor \frac{x_{i+r}+y_{i+r}+c_{i+r}}{2} \rfloor$. Then if $z_{i+r} = 1$, $c_{i+r+1} = 0$. If $z_{i+r} = 0$, $c_{i+r+1} = \lfloor \frac{2x_{i+r}}{2} \rfloor = x_{i+r}$, and we obtain $z_{i+r+1} = x_{i+r+1} \oplus y_{i+r+1} \oplus x_{i+r}$. In both cases we obtain one more linear equation, that gives a total of $r+2$ linear equations, except when $i+r-1 = n-1$ in which case we get only r, and when $i + r - 1 = n - 2$ we get $r + 1$.

We can observe that when $i = 0$, $p = 1$ and we get the same equations as we always have $c_0 = 0$, in particular when $i = 0$ and $r \geq n - 2$ we get again the Proposition 4.1.

To prove the second and third assertion of Theorem 1, we need to observe that $y \boxplus (\bar{y} \boxplus 1) = 0$, where $\bar{y}$ is the result of bitwise complementation of y. Thus we have:

$$x \boxplus y = z \;\Leftrightarrow\; z \boxplus (\bar{y} \boxplus 1) = x \;\Leftrightarrow\; (z \boxplus 1) \boxplus \bar{y} = x \tag{5}$$

In the case of the second assertion, the r least significant bits of z are 0. Then $z \boxplus 1$ becomes $z \oplus 1$ and $c'_1 = \lfloor \frac{\bar{y}_0 + 1}{2} \rfloor$, that is 0 with probability $\frac{1}{2}$. We obtain by induction that if $c'_1 = 0$, $y_j \oplus x_j \oplus 1 = 0$ for $1 \leq j \leq r - 1$ and $y_r \oplus x_r \oplus z_r = 0$. With the least significant bit equation $y_0 \oplus x_0 = 0$ we obtain $r + 1$ equations. Another equation comes from the fact that the carry $c_{r+1} = \lfloor \frac{x_r + y_r + c_r}{2} \rfloor$. This implies that if $z_r = 1$, $c_{r+1} = 0$ and if $z_r = 0$, $c_{r+1} = x_r$. As the carry c_{r+1} can be linearly described in both cases, we obtain one more linear equation $z_{r+1} = x_{r+1} \oplus y_{r+1} \oplus c_{r+1}$. We then obtain a total of $r + 2$ linear equations, except when $r - 1 = n - 2$ in which case we get one less, and when $r - 1 = n - 1$ in case we get two less.

For the third assertion, we consider the carry c''_i in the addition of three numbers: $z \boxplus 1 \boxplus \bar{y} = x$. We assume that $c''_i = 0$ and $z_i = 0, ..., z_{i+r-1} = 0$, we also obtain by induction $r+2$ (or $r+1$ if $i+r-1 = n-2$, or r if $i+r-1 = n-1$) linear I/O equations: $y_j \oplus 1 \oplus x_j = 0$ for $i \leq j \leq i + r - 1$, $y_{i+r} \oplus z_{i+r} \oplus x_{i+r} = 0$ and $y_{i+r+1} \oplus z_{i+r+1} \oplus x_{i+r+1} \oplus c_{i+r+1} = 0$ where the carry c_{i+r+1} is zero or x_{i+r}, depending on the value of z_{i+r}.

Here the first carry $c''_1 = \lfloor \frac{z_0 + \bar{y}_0 + 1}{2} \rfloor$ has a probability $\frac{1}{4}$ to be 0. The other carries c''_j can be computed as : $c''_j = \lfloor \frac{z_{j-1} + \bar{y}_{j-1} + c''_{j-1}}{2} \rfloor$. The probability that $c''_i = 0$ for $z \boxplus \bar{y} \boxplus 1 = x$ is computed by induction in the same way as for the first assertion and gives a probability $p''_i = \frac{1}{2} - \frac{1}{2^{i+1}}$.

Proof of Theorem 2. Appears in the extended version of this paper.

B Additional Linear Equations in Attack on KGSnow 2.0

Let us consider the equation (2) (section 6.1) at step 9. According to theorem 2, as $R2_9$ is known, the two least significant bits of $R1_{10}$ can be expressed as linear expressions of the initial state bits. Depending on the value of $R2_9$, more bits of $R1_{10}$ may be expressed in such a way.

In equation (1) at step 10, we can now substitue the two least significant bits of $R1_{10}$ by these linear expressions. Now we observe that in this equation (1), the value of s_{10} is known as it depends on the value of $R1_6$ that has been guessed. Then the value of $z_{10} \oplus R2_{10} \oplus s_{10}$ is know. This implies by theorem 1 that we have two linear boolean equations mixing only initial state bits. Depending on the value of $R2_9$ and $z_{10} \oplus R2_{10} \oplus s_{10}$, we may have more linear expressions.