African Journal of Mathematics and Computer Science Research Vol. 5(1), pp. 1-8, 9 January, 2012
Available online at http://www.academicjournals.org/AJMCSR
DOI: 10.5897/AJMCSR11.160
ISSN 2006-9731 ©2012 Academic Journals

Full Length Research Paper

Wu's algorithm and its possible application in cryptanalysis

T. L. Grobler[1,2]*, A. J. van Zyl[3], J. C. Olivier[4], W. Kleynhans[1,5], B. P. Salmon[1,5] and W. T. Penzhorn[1]

[1]Department of Electrical, Electronic and Computer Engineering, University of Pretoria, South-Africa.
[2]Defence, Peace, Safety and Security, Council for Scientific and Industrial Research, South-Africa.
[3]Department of Mathematics and Applied Mathematics, University of Pretoria, South-Africa.
[4]School of Engineering, University of Tasmania, Hobart, Australia.
[5]Meraka Institute, Council for Scientific and Industrial Research, South-Africa.

Accepted 5 December, 2012

In this paper we reviewed Wu's algorithm and introduced it as a cryptanalysis technique. This study reveals that when Wu's algorithm is used for cryptanalysis it simplifies. This is true because Wu's algorithm has to be applied to binary polynomials only, when used for cryptanalysis. To summarize, we gave a full description of Wu's algorithm in the binary case and also a basic example of using binary Wu to break an s-box.

Key words: Wu-Ritt decomposition algorithm, s-box, binary polynomials, cryptanalysis.

INTRODUCTION

Cryptographic algorithms use s-boxes to introduce non-linearity into a method. Such an s-box can be represented by a set of non-linear binary polynomials. A non-linear s-box is difficult to reverse. If the output is known it is difficult to derive the input. This is true due to the fact that reversing an s-box is the same as solving a set of non-linear polynomials (finding the roots of such a set). Wu's algorithm is a mechanical method to solve non-linear polynomial sets and can thus be used as a cryptanalysis technique.

The algorithm was originally developed by J. F. Ritt in the late forties (See his now classic book Differential Algebra: Ritt (1950). Later it was independently rediscovered and improved by the Chinese mathematician Wu Wen-Tsün in the late seventies (Wen-Tsün, 1978, 1984, 1986). Other resources describing Wu's algorithm include, Bayram and Celik, (2002), and Kapur and Mundy (1988). The cornerstone of this algorithm is the generation of a Ritt-Wu characteristic set, which is a triangular set that is constructed from a polynomial set so that the characteristic set shares some of the properties of the original set (Gallo and Mishra, 1990b; Ritt, 1950). The algorithm has found wide usage in mechanical theorem proving (Wen-Tsün, 1986; Kapur and Mundy, 1988) and solving polynomial sets (Bayram, 2002) to name a few applications. Improved variations on the original algorithm can be found in Gallo and Mishra (1990b), and Chou and Gao, (2008). The computational complexity of the algorithm is thoroughly analyzed in Gallo and Mishra (1990a, b).

It has been speculated that the reason Wu has failed to make a large impact is that it has been overshadowed by the Gröbner base algorithm (Buchberger, 1985). A debate regarding the respective power of these two algorithms has sprung up. Experimental data is available for both algorithms in Buchberger (1985), Chou (1985), and Kapur (1986). The complexity of the Gröbner base algorithm is analyzed in Möller and Mora (1984). This paper however does not investigate which of these two algorithms would

*Corresponding author. E-mail: trienkog@gmail.com. Tel: +27124303777.

make a better cryptanalysis method. It only introduces Wu's original algorithm (Wen-Tsün, 1986) as a cryptanalysis technique without the improvements of Gallo and Mishra (1990b), and Chou and Gao, (2008).

Wu's algorithm is used to solve polynomial sets in the decimal number system. This paper will investigate Wu's algorithm in the binary case with the hope that the algorithm will simplify greatly so that it can be used as an effective non-linear attack on s-boxes.

The paper begins by introducing multivariate polynomials. This is followed by a discussion of polynomial sets. These are then used to introduce Wu's algorithm. Once Wu's algorithm is introduced it is modified for the binary case. The paper ends with an example of how Wu can be used as a non-linear attack and a final conclusion discussing the future research required for possible successful implementation of this approach.

MULTIVARIATE POLYNOMIALS

Basic definitions

Multivariate polynomials are polynomials consisting of more than one variable, $x_1, x_2, \cdots, x_r$. Without loss of generality one may assume that the ordering of the above mentioned variables are $x_1 < x_2 < \cdots < x_r$. A multivariate polynomial *f* is denoted by (1) if x_m is selected as *master variable*

$$f = I_n x_m^n + I_{n-1} x_m^{n-1} + \cdots + I_0 \quad (1)$$

where $I_k \in \{x_1, x_2, \cdots, x_{m-1}, x_{m+1,} \cdots, x_r\}$, $0 \le k \le n$. I_k is a multivariate polynomial itself and may contain a number in its coefficient.

The *class* of a multivariate polynomial is defined as the greatest subscript *c* of any x contained in *f* and is denoted by $c = class(f)$. The class of a constant is defined to be naught.

When the master variable is equal to x_c (1) becomes

$$f = I_n x_c^n + I_{n-1} x_c^{n-1} + \cdots + I_0 \quad (2)$$

The variable x_c is the *leading variable* and I_n is the *leading or initial coefficient* of *f*. The leading variable is denoted by $x_c = lv(f)$, while the initial is denoted by $I_n = lc(f)$.

The *degree* of a polynomial is the highest degree of the leading variable. The degree of *f* is denoted by *n* = *deg*(*f*). One can also determine the degree of any variable; this is denoted by *deg*(*f*, x_i).

These definitions are described in greater detail in Bayram et al. (2002), Wen-Tsün (1986), and Kapur and Mundy (1986).

Reduction

A polynomial *g* is said to be *reduced* with respect to *f* if the highest degree of x_c $[lv(f)]$ in *g* (if any) is less than the *deg*(*f*). The above does not imply that *g* is of a lesser class than *f*, x_c is only the master variable of *g* and not necessarily its leading variable. If *g* is reduced with respect to *f* we denote it as *g red f*. If *g* is not reduced with respect to *f* we denote it as *g* ~~*red*~~ *f*. The action of reducing *g* with respect to *f* is known as *reduction*.

Reduction is provided by the *pseudo division algorithm*. In normal polynomial mathematics any polynomial $g(x)$ can be written as $g(x) = f(x) \cdot q(x) + r(x)$. When working with multivariate polynomials the above equation becomes

$$I_n^s g = q \cdot f + r \quad (3)$$

where $f = I_n x_c^n + I_{n-1} x_c^{n-1} + \cdots + I_0$ with $I_k \in \{x_1, x_2, \cdots, x_{c-1}\}$ and $g = L_j x_c^j + L_{j-1} x_c^{j-1} + \cdots + L_0$ with $L_k \in \{x_1, x_2, \cdots, x_{c-1}, x_{c+1}, \cdots x_r\}$. Note that $j \ge n$. The polynomial I_n is thus the leading coefficient of *f* and must be non-zero. Also *q* and *r* are multivariate polynomials and *s* is an integer with the following added condition $s \le n - j + 1$. If the integer *s* is the smallest possible power that satisfies (3) and *q* as well as *r* was uniquely determined by some means then *r* is known as the *pseudo remainder* of *g* with respect to *f*. The pseudo remainder has one additional property reduced with respect to *f*, meaning *deg*(*f*, x_c) > *deg*(*r*, x_c). If $n > j$ then *g* is already reduced with respect to *f* and $s = 0, q = 0$ are chosen so that $r = g$ and the above equation still holds.

Pseudo division algorithm

Input to the algorithm is the polynomials *g* and *f* in x_c, thus one can write *g* and *f* as $g = L_j x_c^j + \cdots + L_0$, $f = I_n x_c^n + \cdots + I_0$. Output of the algorithm is the pseudo remainder of *g* with respect to *f* and is denoted by $prem(g, f) = r$.

Set $r = g$.

While *k* = *deg*(r, x_c) $\ge n$

$r = I_n r - C_k x_c^{k-n} f$, where C_k is the leading coefficient of *r* in x_c.
Return *r*.

This algorithm is discussed in greater detail in Chou and Gao, (2008).

Partial order on polynomials

One can define a *partial order on polynomials*. The polynomial *g* has a higher rank than *f* if one of the following two cases holds:

1) $class(f) < class(g)$;
2) $class(f) = class(g)$ and the *deg*(*f*) < *deg*(*g*).

It is denoted as $g > f$ (*g* has a higher rank than *f*). The expression $g > f$ can also be interpreted as *f* has a lower rank than *g*. When neither of the above conditions is met one says that *f* and *g* are of the same rank. This is donated as $f \sim g$. This happens when $class(f) = class(g)$ and the *deg*(*f*) = *deg*(*g*) or when both polynomials are constants.

One has to note here that reduction is not parallel to the partial order. If $g > f$ it does not mean that *g* is not reduced with respect to *f*. Reduction concentrates on a single variable, while the partial order concentrates on the entire polynomial.

POLYNOMIAL SETS

A *polynomial set* is defined as a collection of polynomials and is denoted by $F = \{f_1, f_2, \cdots, f_p\}$.

Polynomial ideal

Let *F* be a set of polynomials in variables $x_1, x_2 \cdots, x_r$. Let P_i be the totality of linear combinations of polynomials in *F* with polynomials in *K* as coefficients, where *K* consist of all possible polynomials in variables $x_1, x_2 \cdots, x_r$. Then P_i is a *polynomial ideal* generated by *F*.

Let P_i be a prime ideal in variables $x_1, x_2 \cdots, x_r$. Then there may be some x_i such that no non-zero polynomial in P_i involves only x_i, that is every polynomial in which x_i appears effectively also involves some x_j with $i \neq j$. If there exist such a x_i, let us pick one of them and call it u_1. There may be another x_k distinct from u_1 such that no non-zero polynomial in P_i involves only u_1 and the new x_k. If there exist such a x_k, pick one of them and call it u_2. Continuing, one finds a set $u_1, \cdots, u_w (w < r)$. This set is known as *the parametric set of indeterminates* for P_i. The value *w* is known as the *dimension* of P_i and is denoted as *dim* P_i. More information on polynomial ideals can be found in Ritt (1950).

Ascending set

A polynomial set $F = \{f_1, f_2, \cdots, f_p\}$ is said to be an *ascending set* if either of the following two conditions hold:

1) $p = 1$, and $f_1 \neq 0$.
2) $p > 1$, $0 < class(f_1) < class(f_2) < \cdots < class(f_p)$, and f_j is reduced with respect to f_i for each pair $j > i$.

An ascending set is said to be *contradictory* if $p = 1$, $f_1 \neq 0$ with $class(f_1) = 0$. It is clear from the above that $p \leq r$.

Reducing a polynomial with respect to an ascending set

Let $F = \{f_1, f_2, \ldots, f_r\}$ be an ascending set with $class(f_1) > 0$ and *g* a polynomial. We define $prem(g; F)$ inductively to be $prem(prem(g, f_r); f_1, f_2, \ldots, f_{r-1})$. Let the answer be *R*. Note that $prem(g; F)$ means we reduce *g* with respect to *F*. In other words *g* is reduced with respect to all polynomials in *F*. This definition leads to the *remainder equation*.

$$\prod_{i=1}^{r} I_i^{s_i} g = \sum_{i=1}^{r} Q_i f_i + R \qquad (4)$$

which is derived in Wen-Tsün (1986).

Partial order for ascending sets

One can extend the partial order on polynomials to provide a *partial order for ascending sets*. Let $F = (f_1, f_2, \ldots, f_p)$ and $G = (g_1, g_2, \ldots g_s)$ be ascending sets. We define $F < G$ whenever there is either a $j \leq \min\{p, s\}$ such that $f_i \sim g_i$ for all *i* < *j*, but $f_j < g_j$, or *p* > *s* and $f_i \sim g_i$ for all $i \leq s$. This means that $F < G$ if either of two cases holds:

1) When f_i and g_i are incomparable up to a point *j*

($j \le \min\{p, s\}$), but $f_j < g_j$.

2) When *F* is longer than *G*, and each g_i is incomparable to the corresponding f_i .

For incomparable ascending sets we write *F* ~ *G*. When $F < G$ we say that *G* has a higher rank than *F* or that *F* has a lower rank than *G*. When *F* ~ *G* we say *F* and *G* are of the same rank. The ascending sets will be of the same rank when *p* = *s*, $f_1 \sim g_1, \cdots, f_p \sim g_p$. Note from above that the larger ascending set may have the lower rank.

Minimal ascending set

A *minimal ascending set* is defined as any set from a collection of ascending sets that has the lowest rank of all the sets in the collection. What is meant by the lowest rank is that no other set in the collection may be of a lower rank than a minimal ascending set. There may however be some sets that have the same rank than a minimal ascending set. These sets are minimal ascending sets of the collection as well. Thus one can say a set F_i is a minimal ascending set of the collection of ascending sets $F_1, \cdots, F_v$ if $F_i < F_j$ or $F_i \sim F_j$ for all $1 \le j \le v, i \ne j$.

The formal definition of a minimal ascending set is given by the following lemma (Wen-Tsün, 1986).

Lemma 1

Let $F_1, F_2, \cdots, F_q, \cdots$ be a sequence of ascending sets F_q for which the rank never increases, or for any q we have either $F_{q+1} < F_q$ or $F_{q+1} \sim F_q$. Then there is an index q' such that for any $q > q'$ we have $F_q \sim F_{q'}$. In other words, there is some q' such that any F_q for which $q \ge q'$ is a minimal ascending set of the above sequence.

Because any ascending collection can be written as the sequence described in the above lemma (due to the partial order for ascending sets) one can deduce that every ascending collection has at least one minimal ascending set.

Basic set

The *basic set* of a non-empty polynomial set *F* is any ascending subset from *F* that is a minimal ascending set of all the possible ascending subsets that can be constructed of *F*. In other words there will be no other ascending subset from *F* with a lower rank than the basic set of *F*. Any ascending subset of *F* that has the same rank as a basic set of *F* is a basic set of *F* as well. Please note that a basic set is an ascending set by construction. So let $\Omega = \{F_1, F_2, \cdots, F_v\}$ be a set consisting of all possible ascending subsets of *F*, so that $F_j \subset F$ for all $(1 \le j \le v)$. Then any F_i such that $F_i < F_j$ or $F_i \sim F_j$ for all $(1 \le j \le v)$ is a basic set of *F*. The main difference between a basic set and a minimal ascending set is the fact that a basic set is an ascending set one constructs from a single polynomial set, while a minimal ascending set is a set belonging to an existing collection of ascending sets.

Constructing a basic set of a polynomial set

The input to the algorithm is a polynomial set *F*. The output of the algorithm is *B* a basic set of *F*. This is denoted as *B* = *basic_set*(*F*).

Set $F_1 = F$, *B* is equal to an empty set.
k = 1
While (*B* not the basic set of *F*)
Find the first polynomial *f* from F_k of lowest rank and add it to *B*.
F_{k+1} is equal to an empty set.
If (*class*(*f*) = 0 and *k* = 1) then *B* is a basic set of *F,*
Else
For (*i* = 1, $i \le size(F_k)$, *i*++)
If ((f_i *red* f) and ($f_i \ne f$)) add f_i to F_{k+1}
end{for}
If (F_{k+1} is empty) *B* is a basic set of *F*.
k++
end {else}
end {while}
Return *B*.

In simpler terms what this algorithm does is it constructs the longest possible ascending set of lowest rank. Remember there is more than one basic set, this algorithm uses the first polynomial of lowest rank; if you use the second you would get a completely different basic set.

Characteristic set

The *characteristic set* of a polynomial set *F* is a set that is constructed in a special manner so that this new set contains all the zeros (solutions) of *F* and is triangular in form. What is meant by the solutions of *F* is all those numbers that would make *F* = 0 true. A triangular form (ascending set) is an easy form to solve for instance $\{y^3 - y^2, xy + x + y\}$ is easier to solve than $\{xy + x + y, xy^2 + x + y\}$,

because it is triangular in form, in other words direct back substitution is possible. A more formal definition of a characteristic set is as follow. Let P_i be the prime ideal generated by *F*. Then the characteristic set of *F* is the basic set of P_i.

Constructing a characteristic set from a polynomial set

The input to the algorithm is a polynomial set *F*. The output of the algorithm is *C* a characteristic set of *F*. This is denoted as *C* = *char_set*(*F*).

C is an empty set, *T* is an empty set.
While (*C* not the characteristic set of *F*)
B = *basic_set*(*F*).
If (*class*(*B*) = 0) then *C* = {1} and is the characteristic set of *F*,
Else
T = *F* - *B*, *T* is thus the difference between the sets *F* and *B*. Those polynomials that are in *F* but not in *B*.
T_2 is an empty set, and *R* is a polynomial.
For (*i* = 1, $i \le size(T)$, *i*++), {Forms the set of remainders of polynomials in *T* with respect to *B*}
$R = prem(t_i; B)$, where t_i is the i[th] polynomial in *T*.
If ($R \neq 0$) then add *R* to T_2.
end{for}
If (T_2 is empty) then *C* = *B* and is the characteristic set of *F*.
Else add T_2 to *F*.
end{else1}
end{while}
Return *C*.

The following theorem from Wen-Tsün (1986) is thus briefly summarized.

Ritt's theorem also known as the Well-Ordering theorem

There is an algorithm which permits us to get, after a mechanically finite number of steps, either a polynomial set C = {1}, or a non-contradictory ascending set $C = \{c_1, \cdots, c_v\}$ with initials $I_1, \cdots, I_v$ such that any zero of F is also a zero of C, and any zero of C which is not a zero of any of the initials I_i, will also be a zero of F. The set C obtained in this manner is known as the characteristic set of F.

The size of a characteristic set *C* can be calculated using Equation 5 as stated by Gallo and Mishra (1990b)

$$n \ge |C| \ge n - \dim P_i \quad (5)$$

where *C* is the characteristic set of *F*, *n* is the size of *F*, and P_i is the polynomial ideal generated by *F*.

WU'S ALGORITHM

So far, we know that the roots of *F* are pretty much the roots of *C* if $I \neq 0$. But how do we find those zeros of *F* for the case when *I* = 0? This is quite easy when one calculates the characteristic set of $F \cup \{I\}$. We recursively call the characteristic set algorithm to investigate this case. It may produce more roots, or it may proclaim that the set $F \cup \{I\}$ has no roots. Hence

$$Zeros(F) = [Zeros(C) - Zeros(I)] \cup \bigcup Zeros(F \cup \{I_i\}) \quad (6)$$

where I_i is an initial of $c_i \in C$ and $I = \prod I_i$. One does not know how deep this recursion goes but that it must terminate is for certain due to the fact that the basic sets decrease in rank. We write $[Zeros(C) - Zeros(I)]$ for the set of zeros of *C* subject to the condition that $I \neq 0$. So the zeros of *F* are those zeros of *C* which are not zeros of *I*, together with the zeros of $F \cup \{I_i\}$. The last term captures those zeros that are simultaneous solutions of *F* and *I*. We stop pursuing a branch whenever we get that the *Zeros*($F \cup \{I_i\}$) is an empty set. Wu's complete algorithm is described below. Given a set of polynomials *F*, return a set of ascending sets of solutions *Z*.

Z is an empty set, *G* = *F*.
While (*G* is non-empty)
Pick a set of polynomials F' from *G*.
G = *G* - F', Let *G* be the polynomials that are in *G* but not in F'.
$C = Char\_set(F')$.
If ($C \neq \{1\}$) $Z = Z \cup C$.
T is an empty set.
For (All initials I_i of *C*)
If (I_i are non-constant) Add I_i to *T*.
end{for}
If (*T* is not empty) $G = G \cup F' \cup T$.
end{while}
Return *Z*.

A complete theoretical analysis of Wu's method can be found in Gallo and Mishra (1990a, b).

WU'S ALGORITHM IN THE BINARY CASE

There are quite a few simplifications that occur when Wu's algorithm is applied to binary multivariate polynomials.

Binary multivariate polynomials

A *binary multivariate polynomial* consists of binary variables $x_1, x_2 \cdots, x_r$. These variables can assume only one of two values a '0' or a '1'. Equation (2) transforms to (10) when the variables $x_1, x_2 \cdots, x_r$ are binary.

$$f = I_1 x_c \oplus I_0 \tag{10}$$

Note that plus becomes XOR and multiplication becomes AND. The value of *f* can thus either be a '1' or a '0'. Also the degree of *f* is always one. All the definitions stay exactly the same in the binary case except for a few which will be highlighted below.

The definition of reduction does simplify. A binary polynomial *g* is said to be *reduced* with respect to a binary polynomial *f* if x_c $[lv(f)]$ is not present in *g.* The above facts simplify the pseudo division algorithm to one single equation

$$bprem(g, f) = r = I_1 g \oplus L_1 f \tag{7}$$

Where *g* and *f* are binary polynomials in x_c $[lv(f)]$, I_1 is the initial of *f* and L_1 is the initial of *g*. Also the variable *r* denotes the binary pseudo remainder. The notation $bprem(g, f)$ is used to show that the binary pseudo remainder is being calculated. Also remember if *g* is already reduced with respect to *f* then *g* is the pseudo remainder and not (7). The partial order for polynomials is also affected by the binary condition; and is redefined as follow. The binary polynomial *g* has a higher rank than a binary polynomial *f* if the $class(f) < class(g)$ and is denoted by *f* < *g*. When the $class(f) = class(g)$ we say *g* and *f* are of the same rank, and denote it by $f \sim g$.

All other algorithms and definitions stay exactly the same except when it uses the original definitions of the above. Then the algorithm or definition must use the above instead. For instance when an algorithm or definition uses *prem* one should use *bprem*. Also for linear combinations one should use XOR and not standard addition and AND instead of multiplication.

Influence on WU's algorithm

Wu's algorithm itself simplifies as well, hence Wu's complete algorithm is not needed. One only needs one iteration of the characteristic set algorithm to obtain the roots of a binary polynomial set that has a unique solution. The reason for this is explained below.

When the characteristic set algorithm is applied to a binary polynomial set one gets either one of three possible outcomes.

1) A characteristic set of {1} is returned, showing that the system has no solution.
2) A characteristic set of the following form is returned

$$\begin{Bmatrix} x_1 \oplus a_1, \\ x_2 \oplus a_2, \\ \vdots \\ x_n \oplus a_n \end{Bmatrix} \tag{8}$$

showing that the system has a unique solution. Equation (8) has a size that equals the size of the original set. Remember a_i can be either a '1' or a '0'.

3) A characteristic set that has a smaller size than (8) is returned, showing that the system has more than one solution.

From Ritt's theorem it is clear that when a system has no solutions it will generate a characteristic set of {1}.

The reason for (8) is as follows, if a binary polynomial set has only one unique solution then the polynomial ideal that is generated by such a set will be a zero dimensional ideal. This is so, because there is no interdependence between the variables in such a system. The linear combinations that form the polynomial ideal can eliminate all variables except one, without changing the roots of the original system when the system has a unique solution. As stated before, the size of a characteristic set can be calculated using (5). Thus because the set generates a zero dimensional ideal, (5) becomes $|C| = n$.

Now because, the characteristic set is also a binary ascending set and must have a size that is equal to the original set it will be of the form stated in (8). In the case of (8) the unique solution can be determined directly and is equal to $\{a_1, a_2, \cdots, a_n\}$. In this case one characteristic set iteration is required to compute the unique solution.

When a system has more than one unique solution it does not generate a zero dimensional ideal, and therefore generates a characteristic set that is smaller than the original set. It does not do this because the linear combinations that form the polynomial ideal can not eliminate all the variables except one without changing the roots of the original system. This is impossible due to the way a polynomial ideal is constructed; it preserves the

original roots (linear combinations). Thus Ritt's theorem can be restated for the binary case as follow.

Ritt's theorem in the binary case

There is an algorithm which permits us to get, after a mechanically finite number of steps, either a binary polynomial set C = {1}, or a non-contradictory binary ascending set $C = \{c_1, \cdots, c_v\}$ *with initials that are all equal to 1 such that any zero of F is also a zero of C, and any zero of C is also a zero of F, or a non-contradictory ascending set* $C = \{c_1, \cdots, c_v\}$ *with initials* $I_1, \cdots, I_v$ *such that any zero of F is also a zero of C, and any zero of C which is not a zero of any of the initials* I_i*, will also be a zero of F. The set C obtained in this manner is known as the binary characteristic set of F.*

In the remainder of the paper we will concentrate on the case when *F* has a unique solution, because this has direct applicability to cryptography.

Example

It is time to illustrate the use of Wu as an attack. Let us investigate the following set.

$$F = \left\{ \begin{array}{l} x_3 \oplus x_1 \oplus x_1x_3 \oplus x_2x_3 \oplus x_2x_1 = y_1, \\ x_3 \oplus x_1x_2 = y_2, \\ x_3 \oplus x_2 \oplus x_2x_3 \oplus x_1 = y_3 \end{array} \right\} \quad (9)$$

Assuming that the output vector (y_1, y_2, y_3) and the input vector (x_1, x_2, x_3) of an s-box is related by (9). The concept is thus represented graphically as in Figure 1.

In Figure 1 the variables $y_1, y_2, y_3, x_1, x_2, x_3$ represent the bits of a binary number where the lowest subscript is the LSB, while the highest subscript represents the MSB.

Assuming the output vector (y_1, y_2, y_3) is known as well as the transformation function *F*. The following procedure can be used to calculate the input vector. If it is known that the output vector is equal to $(1,0,0)$ then (9) transforms to

$$\begin{aligned} R_{1_1} &= (x_2 \oplus x_1 \oplus 1) \cdot (x_3 \oplus x_1x_2) \oplus (1) \cdot f_1 \\ &= x_1 \oplus 1 \end{aligned} \quad (10)$$

where each polynomial is equal to naught. Set (10) was constructed by substituting the output vector into (9) and equating each polynomial to naught. Now the input vector is actually the root of (10). We know from "Wu's algorithm in the binary case" that we only require one iteration of the characteristic set algorithm to calculate the root of (10), because (10) has a unique solution. The first thing one needs to calculate is the basic set of F_1. The basic set is calculated as follow. First retrieve the first polynomial of lowest rank from F_1. In this case it will be

$$f_{1_1} = x_3 \oplus x_1 \oplus x_1x_3 \oplus x_2x_3 \oplus x_2x_1 \oplus 1.$$

Now make a separate polynomial set containing all the polynomials in F_1 that are already reduced with respect to f_{1_1}. Because in this case this new set is empty; f_{1_1} is the basic set of F_1. Thus

$$B_1 = \{x_3 \oplus x_1 \oplus x_1x_3 \oplus x_2x_3 \oplus x_2x_1 \oplus 1\} \quad (11)$$

Now remove the basic set from F_1 to form T_{1_1}.

$$T_{1_1} = \left\{ \begin{array}{l} x_3 \oplus x_1x_2, \\ x_3 \oplus x_2 \oplus x_2x_3 \oplus x_1 \end{array} \right\} \quad (12)$$

Now calculate $R_{1_{11}} = bprem(t_{1_1}; B_1)$ using (7) and the section "Reducing a polynomial with respect to an ascending set" under "Polynomial sets".

$$\begin{aligned} R_{1_{11}} &= (x_2 \oplus x_1 \oplus 1) \cdot (x_3 \oplus x_1x_2) \oplus (1) \cdot f_{1_1} \\ &= x_1 \oplus 1 \end{aligned} \quad (13)$$

Because B_1 consists of only one polynomial there is no recursion. Now because $R_{1_{11}}$ is not equal to naught it gets added to T_{1_2}. Now we need to calculate

$$R_{1_{21}} = bprem(t_{1_2}; B_1) = x_1x_2 \oplus x_2 \oplus x_1 \oplus 1 \quad (14)$$

which is also not equal to naught and is thus added to T_{1_2}. Now T_{1_2} is equal to

$$T_{1_2} = \left\{ \begin{array}{l} x_1 \oplus 1, \\ x_1x_2 \oplus x_2 \oplus x_1 \oplus 1 \end{array} \right\} \quad (15)$$

Because T_{1_2} is not empty it gets added to F_1 to form F_2 which is now equal to

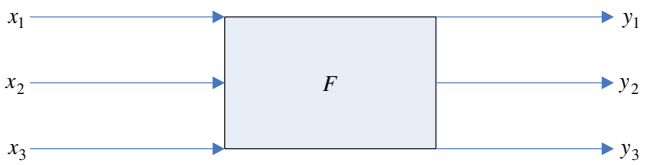

Figure 1. Graphical representation of (9).

$$F_2 = \begin{Bmatrix} x_1 \oplus 1, \\ x_1x_2 \oplus x_2 \oplus x_1 \oplus 1, \\ x_3 \oplus x_1 \oplus x_1x_3 \oplus x_2x_3 \oplus x_2x_1 \oplus 1, \\ x_3 \oplus x_1x_2, \\ x_3 \oplus x_2 \oplus x_2x_3 \oplus x_1 \end{Bmatrix} \quad (16)$$

Next we need to recalculate the basic set of F_2. In this case it is equal to $B_2 = \{x_1 \oplus 1\}$. Also T_{2_1} is recalculated by removing B_2 from F_2. Applying the same procedure as above one can calculate T_{2_2}. For convenience sake let's define a new syntax.

$$bprem\{T_{n_1}; B_n\} = \begin{Bmatrix} bprem(t_{n_1}; B_n), \\ bprem(t_{n_2}; B_n), \\ \vdots \\ bprem(t_{n_r}; B_n) \end{Bmatrix} \quad (17)$$

Please note that if $bprem(t_{n_i}; B_n)$ is equal to naught it does not get added to T_{n_2}.

Using (17) we can calculate T_{2_2}

$$T_{2_2} = non\_zero\left(bprem(T_{2_1}; B_2)\right) = \begin{Bmatrix} x_2x_3 \oplus x_2, \\ x_3 \oplus x_2, \\ x_2x_3 \oplus x_3 \oplus x_2 \oplus 1 \end{Bmatrix} \quad (18)$$

Now F_3 is equal to $F_3 = F_2 \bigcup T_{2_2}$.

Continuing as before one calculates T_{3_2}. Now $F_4 = F_3 \bigcup T_{3_2}$. $B_5 = \begin{Bmatrix} x_1 \oplus 1 \\ x_2 \oplus 1 \\ x_3 \oplus 1 \end{Bmatrix}$. We calculate T_{5_1} with $T_{5_1} = F_5 - B_5$ and then we calculate the set T_{5_2} wit $T_{5_2} = non\_zero\left(bprem(T_{5_1}; B_5)\right)$. This leads to $T_{5_2} = \{\}$. Now because T_{5_2} is an empty set we know B_5 is the characteristic set *F*. The input vector can now be determined directly and is equal to (1,1,1).

CONCLUSION

What is clear from this article is the fact that Wu's original algorithm does work and can be used as a cryptanalysis method.

REFERENCES

Bayram M, Çelik E (2002). Simultaneous solutions of polynomial equations. Appl. Math. Comput., 113: 533-538.
Buchberger B (1985). Gröbner basis: An Algorithmic method in Polynomial Ideal Theory. Recent Trends in Multidimensional System Theory. Reidel.
Chou SC (1985). Proving and Discovering Theorems in Elementary Geometries Using Wu's Method. Phd dissertation, Department of Mathematics , University of Texas at Austin.
Chou SC, Gao X (2008). Ritt-Wu's Decomposition Algorithm, Department of Computer Sciences. The University of Texas at Austin, Texas 78712 USA, pp. 1-15.
Gallo G, Mishra B (1990a). Efficient Algorithms and Bounds for Wu-Ritt Characteristic Sets. In Proceedings of MEGA 90: Meeting on Effective Methods in Algebraic Geometry, Castiglioncello, Livorno, Italy.
Gallo G, Mishra B (1990b). Wu-Ritt characteristic sets and their complexity, Courant Institute, New York University, pp. 1-25.
Kapur D (1986). Geometry Theorem Proving for Gröbner Bases. J. Symb. Comput., 2: 399-412.
Kapur D, Mundy JL (1988). Wu's method and it's application to perspective viewing. Artif. Intell., 37: 15-36.
Möller HM, Mora F (1984). Upper and Lower Bounds for the Degree of Gröbner basis. Lecture Notes in Computer Sciences. Springer Verlag, pp. 172-183.
Ritt JF (1950). Differential Algebra, American Mathematical society, New York.
Wen-Tsün W (1978). On the decision problem and the mechanization of theorem proving in elementary geometry. Sci. Sinica, 21: 157-179.
Wen-Tsün W (1986). Basic principles of mechanical theorem proving in geometries. J. Autom. Reasoning, *2:* 221-252.
Wen-Tsün W (1984). Some Recent Advances in Mechanical Theorem-Proving of Geometries, of Automated Theorem Proving: After 25 Years. Contemp. Math. Am. Math. Soc., 29: 235-242.