Algebraic Aspects of Symmetric-key Cryptography

Carlos Cid (carlos.cid@rhul.ac.uk)

Information Security Group
Royal Holloway, University of London

Algebraic Techniques in Cryptanalysis

- Algebra is the *default* tool in the analysis of asymmetric cryptosystems (RSA, ECC, Lattice-based, HFE, etc)
- For symmetric cryptography (block and stream ciphers, hash functions), the most commonly used techniques are *statistical* in nature:
  - Block Ciphers: in linear and differential cryptanalysis (and variants), the attacker attempts to construct statistical patterns through many interactions of the cipher.
  - Stream Ciphers: linear/differential, correlation attacks, distinguishing attacks, etc.
  - Hash Functions: differential attacks, etc.

Algebraic Techniques in Cryptanalysis

- However there has been recently an increase in interest in the use of algebraic techniques in the analysis of symmetric cryptosystems:
  - The choice of Rijndael as the AES (Rijndael has a rich algebraic structure);
  - Cipher representations (dual, embeddings);
  - The proposal of algebraic attacks against stream ciphers (and block ciphers);
  - Parallel developments in asymmetric cryptography (multivariate cryptosystems: HFE, sFLASH, etc).
- In this presentation will give an overview of algebraic cryptanalysis of block and stream ciphers (background and possible future directions…)

Algebraic Attacks against Block Ciphers

- Algebraic (rather than statistical) in nature:
  - exploit the intrinsic algebraic structure of the algorithm.
- The idea: polynomial description of block ciphers.
  - In theory, most block ciphers afford a polynomial representation of the encryption.
- Early attempts to analyse ciphers with a somewhat simple algebraic structure date back to the early/mid 90s.
- Interpolation attack:
  - Proposed by Jakobsen and Knudsen in 96.

Interpolation Attacks

- Suppose a cipher can be expressed by a polynomial with total degree not too large.
  - By using n known plaintext/ciphertext pairs (x_i, y_i), one can construct an algorithm *equivalent* to the cipher (using the *Lagrange Interpolation Formula*).
  - The polynomial $f(x)$ coincides with the encryption:

$$f(x_i) = y_i$$

- Representing a cipher as a polynomial may allow encryption/decryption without knowledge of the key.

Interpolation Attacks

- However, for most ciphers the degree of such polynomial is just too high (too many unknown coefficients), perhaps approaching or exceeding the codebook.
  - Thus this should not offer any cryptanalytic benefit.
  - However it was applied against (a variant of) of SHARK ($\mathrm{SBox}(x) = x^{-1}$).
- Yet, the proposal of Interpolation Attacks ultimately shows some of the dangers of using operations with a very simple algebraic structure as component of an iterative cipher
  - even if these components were extremely good against conventional cryptanalysis, e.g. differential and linear cryptanalysis.

Algebraic Attacks against Block Ciphers

- An alternative, perhaps more promising approach, is to express the encryption operation as a *system* of polynomial equations.
  - While in theory most modern block ciphers can be fully described by a system of multivariate polynomials over a finite field, for the majority of the cases such systems prove to be just too complex for any practical purpose.
  - Yet there are a number of ciphers that present a highly algebraic structure, and could therefore be more vulnerable to algebraic attacks (e.g. the AES).

Algebraic Attacks against Block Ciphers

- "Algebraic Attack": typically refers to the technique of expressing the whole cryptosystem as a large system of multivariate polynomial equations.
- In principle applicable to both block ciphers and stream ciphers.
- Two steps:
  - Obtain a representation of the cipher as a system of equations.
  - Consider methods for solving the system.

Polynomial System from Block Ciphers

- Polynomial System from Block Ciphers:
  - Linear Equations from the diffusion layer and key addition.
  - Non-linear equations from the substitution layer.
  - Key Schedule Equations.
  - Field Equations.

Polynomial System from Block Ciphers

- For the non-linear equations, we distinct two cases:
  - *Explicit equations*: equations of the form $y_i = f_i(x_0, x_1, \ldots, x_{n-1})$.
  - *Implicit equations*: equations of the form $g(x_0,\ldots,x_{n-1}; y_0,\ldots,y_{m-1}) = 0$.
- One may consider algebraic attacks when these equations have small degree.

Polynomial System from Block Ciphers

- When mounting an algebraic attack, for each non-linear component of the cipher, one attempts to obtain as many low-degree, linearly independent equations as possible.
- The more relations, the best.
  - it is well-known that *overdefined* systems are generally easier to solve .

Polynomial System from Block Ciphers

- Field Equations:
  - We are only interested in the solutions in the ground field (e.g. GF(2) or GF(2^8)).
  - However the method of solution may yield solutions in the algebraic closure.
  - So we also add to the system the so-called field equations
    $$x^q - x = 0$$
    for all variables in the system (over GF(q)).
  - This ensures that all solutions found are in GF(q).
  - Also in computations of the solution (say of its GB), all monomials are reduced by $x_i^q - x_i$.

Algebraic Attacks against Block Ciphers

- In its general form, an algebraic attack is mounted by expressing the full cipher operation as a system of low-degree multivariate equations:
  - involving the (known) plaintext and ciphertext values, the secret key and a large number of intermediate variables arising in the cipher operation.
  - The field equations are often also included.
  - Results on very, very large systems (typically over GF(2)).
- Attack usually requires only one single plaintext/ciphertext pair.
  - Solution = key recovery!!
- Efficient algorithms for solving algebraic systems:
  - the essential ingredients of algebraic attacks and have recently started receiving special attention from the cryptographic community.

Methods of Solution of Polynomial Systems

- Solving multivariate polynomial systems is a typical problem studied in Algebraic Geometry and Computational Algebra.
- Computer Algebra has recently become an important tool in cryptography.
- Methods (used in cryptology):
  - Linearisation principle;
  - XL and variants;
  - Groebner Basis algorithms (Buchberger, F_4 , F_5).

Solution of Polynomial Systems – The Problem

- Let k be a field and $f_1,\ldots, f_m$ be polynomials in n variables with coefficients in k, and K an algebraic extension of k.
- The problem is:
  - find $(x_1,\ldots,x_n) \in K^n$ such that $f_i(x_1,\ldots,x_n) = 0$.
- This problem is often studied in the context of abstract algebra:
  - let $I \subseteq k[X_1,\ldots,X_n]$ be the *ideal* generated by $f_1,\ldots, f_m$ and $V(I) = \{(x_1, \ldots, x_n) \in K^n;\ f_i(x_1, \ldots, x_n) = 0\}$ the *variety* over K associated to I. The problem is then to find $V(I)$.

Linearisation

- The method of *linearisation* is a well-known technique for solving large systems of multivariate polynomial equations:
  - Consider all monomials in the system as independent variables and solve the system using linear algebra techniques (i.e. Gaussian reduction).

$$M = \begin{array}{c} \\ f_1 \\ \vdots \\ f_m \end{array} \begin{array}{c} \begin{array}{ccc} \dots & X^\alpha & \dots \end{array} \\ \begin{pmatrix} \dots & c^1_\alpha & \dots \\ & & \\ \dots & c^m_\alpha & \dots \end{pmatrix} \end{array}$$

Linearisation

- The effectiveness of the method clearly depends of the number of linearly independent polynomials in the system.
- In the case of Boolean functions, the total number of monomials of degree $\leq d$ is:

$$R = \sum_{i=0}^{d} \binom{n}{i} \approx n^d$$

- Complexity: $O(N^3)$, where N is the size of M (i.e. $O(n^{3d})$). In fact we may theoretically write $O(N^\omega)$, where $\omega \approx 2 + \varepsilon$, if the matrix is sparse.

Linearisation

- Linearisation has been considered in the cryptanalysis of some LFSR-based stream ciphers.
  - Each new bit of the key stream gives rise to a new equation on the key bits, and by using a large number of bits from the key stream, one should have in theory enough equations to directly apply linearization.
- Note however that the problem of estimating the rank of the linearised system is very difficult.

Linearisation

- In order to apply the linearization method, the number of LI equations in the system needs to be approximately the same as the number of monomials in the system.
- When this is not the case, a number of techniques have been proposed that attempt to generate enough LI equations.
- The most prominent is the XL algorithm.

XL (*eXtended Linearisation*)

Courtois, Klimov, Patarin, Shamir, 2000

- The XL algorithm aims at introducing new rows to the matrix M by multiplication of the original equations by monomials of prescribed degree (i.e. $\deg(X^\beta f_j) \leq D$, where D is the parameter of the algorithm).

$$M = \begin{array}{c} \\ \vdots \\ X^{\beta} f_1 \\ \vdots \\ X^{\beta'} f_m \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} \dots & X^{\alpha} & \dots \end{array} \\ \left(\begin{array}{ccc} & \vdots & \\ \dots & c_{\alpha}^{1,\beta} & \dots \\ & \vdots & \\ \dots & c_{\alpha}^{j,\beta'} & \dots \\ & \vdots & \end{array} \right) \end{array}$$

XL Algorithm

- A is system of m quadratic equations in *n* variables over a field k, and D $\geq$ 2 :
  - Multiply equations by monomials of degree up to D-2;
  - Linear Algebra step
  - Solve univariate equation and substitute
  - Repeat
- The hope is that after few iterations, one can find a solution of A.

XL Algorithm

- The behaviour of the XL algorithm (termination, complexity) has been the focus of study in recent years.
    - In particular, its relationship to GB algorithms (F_4).
- Since the introduction of the XL method, a number of variants have been proposed attempting to exploit some specific properties of the polynomial system.
    - Of particular relevance for the analysis of the block ciphers is the method proposed in 2002, called XSL.

XSL (*eXtended Sparse Linearisation*) Algorithm

- The XSL algorithm was introduced in 2002 by Courtois and Pieprzyk, and is derived from the XL algorithm.
- It is however a method which attempts to exploit the sparsity and specific structure of the equations.
- XSL attracted a lot of attention of the cryptographic community and was the source of much speculation.

XSL Algorithm

- The claim was that with XSL one could:
  - mount a (at least theoretical) successful attack against the AES with 256-bit keys (using the system over GF(2));
  - mount a (at least theoretical) successful attack against the AES with 128-bit keys (using the system over GF(2^8)).
- However recent results (Asiacrypt'05 and FSE'07) have shown that the algorithm does not work as expected (in particular, is not an efficient method to solve the system arising from the AES).

Gröbner Basis Algorithms

- Groebner Basis algorithms are perhaps the best known technique for solving polynomial systems.
- These algorithms return a basis for the ideal derived from the set of equations, which can then be used to obtain the solutions of the system.

Gröbner Basis Algorithms

- Classical algorithm: Buchberger algorithm
- More recent algorithms: Faugère's F_4 and F_5.
  - Use of Linear Algebra;
  - Found to be related to XL (expected to be more efficient);
- It has found recent use in cryptography:
  - Joux and Faugère (CRYPTO'03) - HFE Challenge I (80 variables and 80 equations over GF(2))

Algebraic Attack – the AES

- The only non-linear component of the AES (the S-Box) is *based* on the inverse map on a finite field.
  - The function $y = \mathrm{Inv}(x)$ has high algebraic degree: $y = x^{254}$
  - However the relations

    $$y \,.\, x = 1 \;,\; y^2 \,.\, x = y \;,\; y \,.\, x^2 = x$$

    give rise to 24 *quadratic* relations over GF(2) (23 always valid, 1 not valid if $x = 0$).
- Bits $w=(w_0,\ldots,w_7)$ and $x=(x_0,\ldots,x_7)$, relations such as:

$$\begin{aligned}0 = {} & x_0 + x_6 + w_0x_2 + w_0x_5 + w_0x_6 + x_0w_7 + x_0w_5 + x_0w_2 + x_2w_5 + x_2w_3 \\ & + x_3w_7 + x_3w_4 + x_3w_2 + x_4w_6 + x_4w_3 + x_4w_1 + x_5w_6 + x_5w_5 + x_5w_4 \\ & + x_5w_2 + x_5w_1 + x_6w_6 + x_6w_7 + x_6w_5 + x_6w_3 + x_7w_6 + x_7w_7 + x_7w_5 \\ & + x_7w_4 + x_7w_2 + x_1w_6 + x_1w_4 + x_1w_1 + 1\end{aligned}$$

Algebraic Attack against the AES

- By combining all equations throughout the cipher, one can express the full AES encryption transformation as a large, sparse and overdefined system of multivariate quadratic equations over GF(2).
  (Courtois and Pieprzyk, 2002).
- Encryptions for different plaintext give rise to different systems (different intermediate variables).
- By performing substitutions we can construct system:
  - 8000 quadratic equations with 1600 variables for the AES-128.
  - 9600 equations if we include the field relations

Algebraic Attack against the AES

- By representing the AES in an alternative way (using the BES cipher), we can obtain a similar system over GF(2^8):
  - Quadratic equations are however simpler (xw=1).
- It is currently not known which of the two systems of equations would be more suitable for mounting an algebraic attack against the AES.
- The question: would we be able to (*theoretically*) solve such systems faster than exhaustive key search (i.e. the order of < 2^{128} operations)??
  - The hope is that *all* we have to do is to compute the Groebner basis for the AES to recover the secret key.

Groebner basis for the AES
(Buchmann, Pychkin, Weinmann - 2006)

- Actually, we already have a GB for the AES!!
- It has been shown that we can construct in straightforward manner a GB for the AES (and other ciphers) wrt degree lexicographic ordering
  - 336 variables and equations:
    - 176 polynomial equations arising from the encryption operation and 160 from the key schedule.
    - 200 have total degree 254 while the remaining 136 are linear.

Groebner basis for the AES

- As a result, we have that the AES ideal is 0-dimensional.
  - R/I has dimension $254^{200} \approx 2^{1598}$
- So we have many solutions in the algebraic closure.
- We have already a GB for AES, but with wrong ordering!!
  - New problem: changing ordering.
  - Infeasible with current known methods.
- The natural obvious approaches do not seem to provide a direct solution to the key recovery problem.
  - Yet it is quite surprising that a Groebner basis for the AES can be obtained in such straightforward manner.

Algebraic Attacks against Block Ciphers – is there hope?

- Algebraic attacks have received a lot of attention of the cryptographic community in recent years.
  - Many strong early claims.
  - However there has not been too much progress in assessing whether they can be effective against block ciphers in general.
- Experiments with small ciphers (small versions of the AES - FSE'05, and Flurry and Curry – RSA-CT 2006) have indicated that modern block cipher features (strong diffusion, etc) make algebraic attacks quite hard.

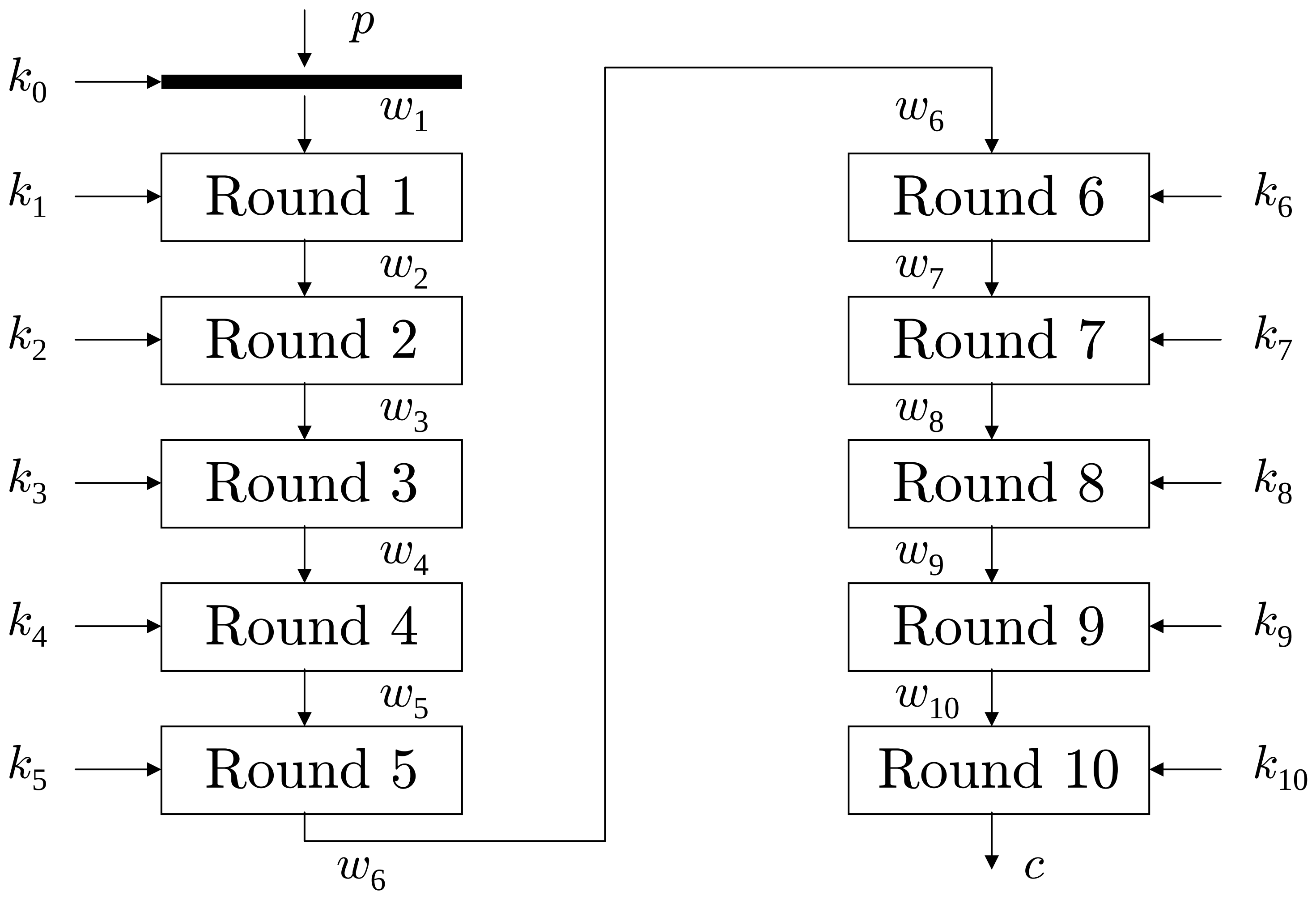
Algebraic Attacks against Block Ciphers – Future Directions

- Current Groebner basis algorithms are powerful tools.
  - they are however general-purpose algorithms, which are used to deal with a number of problems (including computing the solutions of a system).
- Systems arising from ciphers are very structured and with special properties:
  - they are usually sparse, with unique solution over a finite field, structured in blocks of similar format (rounds), etc.
  - Experiments show this can help computations (G. Ars' PhD Thesis 2005)

Algebraic Attacks – Future Directions

- Perhaps the most promising approach is the development of *dedicated* methods for specific ciphers.
    - In a way, XSL was perhaps the first (albeit unsuccessful) attempt.
- Block Cipher systems can be viewed as a set of iterated systems of equations, with similar blocks for every round.
    - Blocks are connected via the input and output, as well as key schedule.
- Dedicated methods could exploit these features. Examples:
    - meet-in-the-middle technique.
    - Groebner Surfing.

AES-128



p
k_0
w_1
k_1
Round 1
w_2
k_2
Round 2
w_3
k_3
Round 3
w_4
k_4
Round 4
w_5
k_5
Round 5
w_6
w_6
Round 6
k_6
w_7
Round 7
k_7
w_8
Round 8
k_8
w_9
Round 9
k_9
w_{10}
Round 10
k_{10}
c

Meet-in-the-Middle
(Cid et al. 2005)

- Rather than solving the full system of equations for n rounds, try to solve two subsystems with n / 2 rounds.
- Two systems:
  - Solve S1 to obtain $k_5(w_6)$;
  - Solve S2 to obtain $k_6(w_6)$;
- Solve S3 $= \{k_5(w_6), k_6(w_6), k_6(k_5)\}$
- This technique is cryptographically (and algebraically) intuitive.
  - Simulations show that it does indeed work better than solving the full system!

Groebner Surfing
(Albrecht 2007)

- Incrementally compute the Groebner Basis.

$$\texttt{GB}\left(\bigcup_{r=1}^{N_r}\mathcal{P}_r\right) = \texttt{GB}(\mathcal{P}_{N_r} \cup \texttt{GB}(\mathcal{P}_{N_{r-1}} \cup (\ldots \cup \texttt{GB}(\mathcal{P}_1)))$$

- This method seems to be more efficient than direct computation (especially if combined with MITM).

Algebraic Attacks – Future Directions

- Probabilistic Approach:
  - Combine algebraic attacks with typical probabilistic cryptanalytic methods.
  - This may simplify the equations and reduce complexity of computations.
  - AES S-Box:
    - Instead of $y = x^{254}$, use $xy = 1$.
  - Boolean monomials of very high degree equal to zero.
  - Low-Degree Approximations (linear cryptanalysis).

Algebraic Attacks – Future Directions

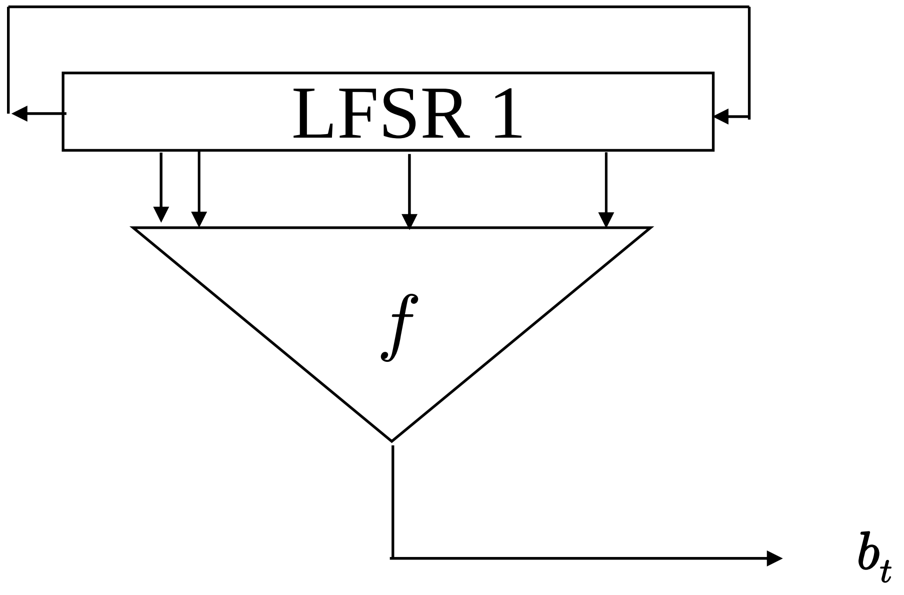
- In its current form, algebraic attack is a key recovery attack.
  - 1 plaintext/ciphertext pair, and key recovery!!
- Can we use the algebraic structure of ciphers for mounting less ambitious attacks?

Algebraic Attacks – Stream Ciphers

- In contrast to block ciphers, algebraic attacks have been (in theory) quite effective in the analysis of several LFSR-based stream ciphers.
- Exploit the fact that each new bit of the keystream gives a new equation on the initial state.
- Collect a large number of bits from the keystream to construct the system of equations.
- First introduced by Courtois and Meier.
  - applies to LFSR-based ciphers, using non-linear Boolean functions as combiner or filter.

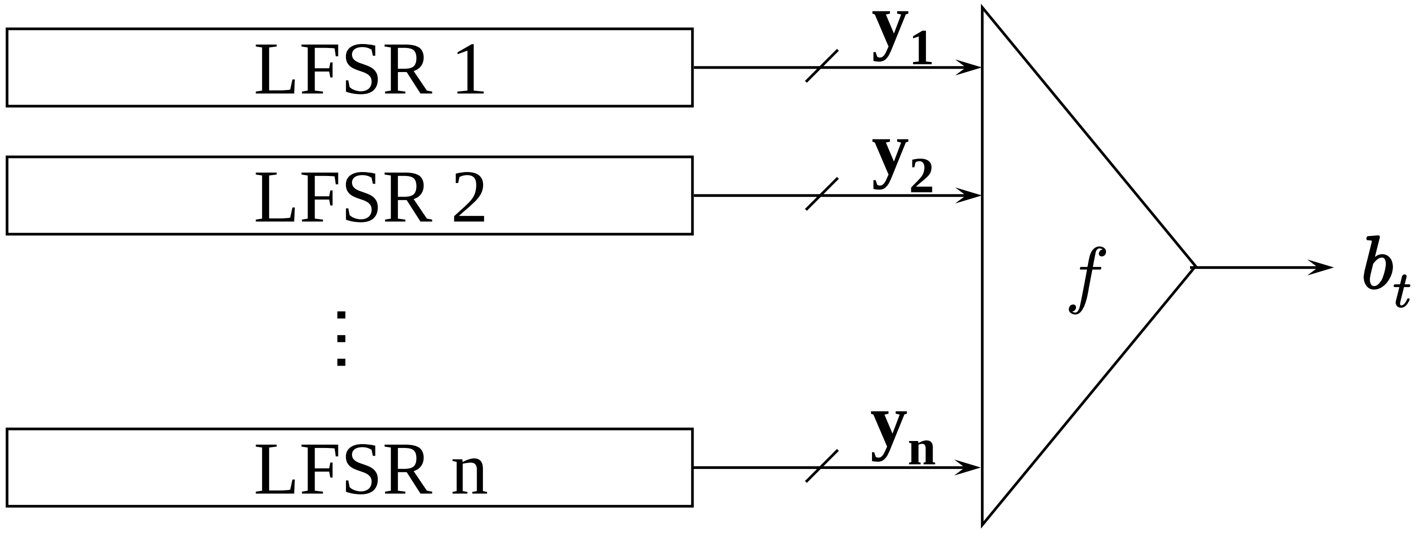
Filter Generator

- Output is computed using a non-linear function of the contents of the LFSR.



Combination Generator

- We combine the outputs from n LFSRs



Filter and Combination Generators

- Boolean Function f – known design criteria:
  - It should have a high algebraic degree;
  - It should have high non-linearity;
  - It should be balanced;
  - It should be correlation-immune of high order.
    - there are some trade-offs to consider.

The problem for the cryptanalyst

- If $(k_0, k_1, \ldots, k_{n\text{-}1})$ is the initial state, L is the linear recursion function, f is the combining function and b_i the output bits then

$$\begin{aligned} b_0 &= f(k_0, k_1, \ldots, k_{n-1}) \\ b_1 &= f(L(k_0, k_1, \ldots, k_{n-1})) \\ b_2 &= f(L^2(k_0, k_1, \ldots, k_{n-1})) \\ \ldots & \\ b_t &= f(L^t(k_0, k_1, \ldots, k_{n-1})) \end{aligned}$$

- Given $b_0, b_1, \ldots, b_t$, we want to recover $(k_0, k_1, \ldots, k_{n\text{-}1})$

Algebraic Attack - First Attempt

- Obtain enough b_t to construct a system large enough such that it has unique solution.
  - the problem is that if f has high algebraic degree, solving the system is very difficult.
  - we could just keep collecting enough bits until we have a system very overdefined for solving it by linearisation.
  - But then we would need around $R = \sum_{i=0}^{d} \binom{n}{i} \approx n^d$ keystream bits and the attack would have complexity R^3 .
    - i.e. the complexity of the attack is polynomial in the key size but exponential in the degree.

Algebraic Attacks

- The goal is to obtain a (hopefully overdefined) system of low degree equations.
  - Usually f has high degree;
  - A possible approach: obtain low-degree approximation of the function f with high probability.

Toyocrypt

- Toyocrypt: submission to CRYPTEC.
- Filter generator with one LFSR of length 128, and output function of degree 63.

$$f(s_0, \ldots, s_{127}) = s_{127} + \sum_{i=0}^{62} s_i s_{\alpha_i} + s_{10}s_{23}s_{32}s_{42} + \prod_{i=0}^{16} s_i + \prod_{j=0}^{62} s_j$$

- *f* satisfies all previously known design criteria.
- However, monomials of degree 17 and 63 will be *almost always* zero!!
  - We can use this fact to construct good approximations of degree 4, with probability ≈ 1 – 2^{17} !

Algebraic Attacks

- What if it is not possible to obtain good low degree approximations?
  - Try to reduce the degree of the equations!
- Suppose that *f* has high degree. We search for function g of low degree such that the relation

$$f(s) \cdot g(s) = h(s) = b_t \cdot g(s)$$

has low degree (i.e. h has low degree).

Toyocrypt

- Monomials of degree 4, 17 and 63 all have a common factor $s_{23}s_{42}$.
- Let $g_1(s) = (s_{23} + 1)$ and $g_2(s) = (s_{42} + 1)$.
  - then $h_1(s) = f(s)\ g_1(s)$ and $h_2(s) = f(s)\ g_2(s)$ have degree 3.
  - so for each output bit, we have 2 low degree equations.
  - using linearisation, we need around 2^{20} keystream bits, with attack complexity $\approx\ 2^{50}$.

Algebraic Attacks against Stream Ciphers

- In general we have:
  - (Theorem) Let f be a Boolean functions in k variables. Then there is a function $g \neq 0$, of degree at most $\lceil k/2 \rceil$ such that $f(s).g(s)$ is of degree at most $\lfloor k/2 \rfloor$.
  - So instead of direct attack (using linearisation) with complexity
    $$R^\omega = \left(\sum_{i=0}^{d} \binom{n}{i} \right)^\omega \approx n^{d\omega}$$
    we can do with (square root attack)
    $$R^\omega = \left(\sum_{i=0}^{d/2} \binom{n}{i} \right)^\omega \approx n^{\frac{d\omega}{2}}$$

Algebraic Attacks against Stream Ciphers

- The attack can be adapted to ciphers that are not regularly clocked:
  - applied to LILI-128 (NESSIE submission).
  - Uses function of degree 6 with only 10 variables; however $f\,.\,(x_9 + 1)\,.\,(x_{10} + 1)$ has degree 4.
- The attack can also be generalised for stream ciphers using combiners with memory
  - applied to Bluetooth generator E0.
- The attack can be improved (fast algebraic attacks).

Algebraic Attacks – Consequences to Design Criteria for LFSR-based Stream Ciphers

- Output function *f* should use a large subset of state bits (LILI-128 used 10 out of 89).
- Output function *f* should have many different terms.
- No multivariate equations of low degree should exist relating the key bits and one or more output bits.

Algebraic Attacks – Consequences to Design Criteria for LFSR-based Stream Ciphers

- In general, algebraic attacks are possible when there exist g , h of low degree for which either
  - $f \,.\, g = 0$.
  - $(f + 1) \,.\, h = 0$
- If $f(s) = b_t = 1$, then we use the first relation and have $g(s) = 0$.
- If $f(s) = b_t = 0$, then we use the second relation and have $h(s) = 0$.

Algebraic Attacks – Consequences to Design Criteria for LFSR-based Stream Ciphers

- Consequently, to mount such attack we need to search for low degree annihilators of *f* and (*f*+1).
- The lowest degree of such annihilators is called the *algebraic immunity* (AI) of *f*.
- For LFSR of length 2^k and AI *d*, the attack complexity would be

$$R^{\omega} = \left(\sum_{i=0}^{d} \binom{2^k}{i} \right)^{\omega} \approx 2^{kd\omega} \approx 2^{5kd/2}$$

Algebraic Attacks against Stream Ciphers

- Algebraic immunity is currently a design criterion for designing LFSR-based (combination and filter) stream ciphers.
- Can we extend it to other types of ciphers?
- Computation of complexity is made considering linearisation as method of solution.
    - can we use any information about the cipher to apply a more efficient method ?

Some New Approaches for Algebraic Cryptanalysis

- New Techniques for Solving Sparse Systems of Equations (Raddum and Semaev – 2007)
  - Equations are not represented as polynomials.
  - Algorithm can be seen as message-passing on a graph.
  - Experiments with DES, small AES, with good results.

Some New Approaches for Algebraic Cryptanalysis

- Using SAT-Solvers
  - Propositional Satisfiability Problem (SAT) - determining whether the variables of a given Boolean formula can be assigned such that the formula evaluates to TRUE.
  - SAT-Solvers are algorithms used for testing satisfiability formulae.

Some New Approaches for Algebraic Cryptanalysis

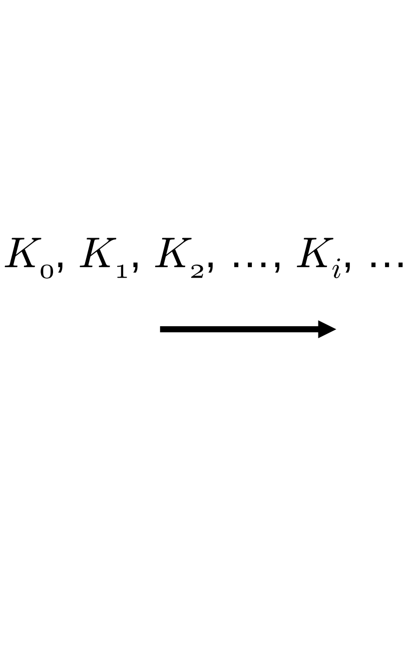
- Algebraic Attacks using SAT-Solvers (Bard, Courtois and Jefferson – 2007)
  - Boolean Equations are described in the conjunctive normal form (CNF).
  - SAT-Solver used to solve the system.
  - Applied to reduced-round DES, KeeLoq (Stream Cipher), with good results.

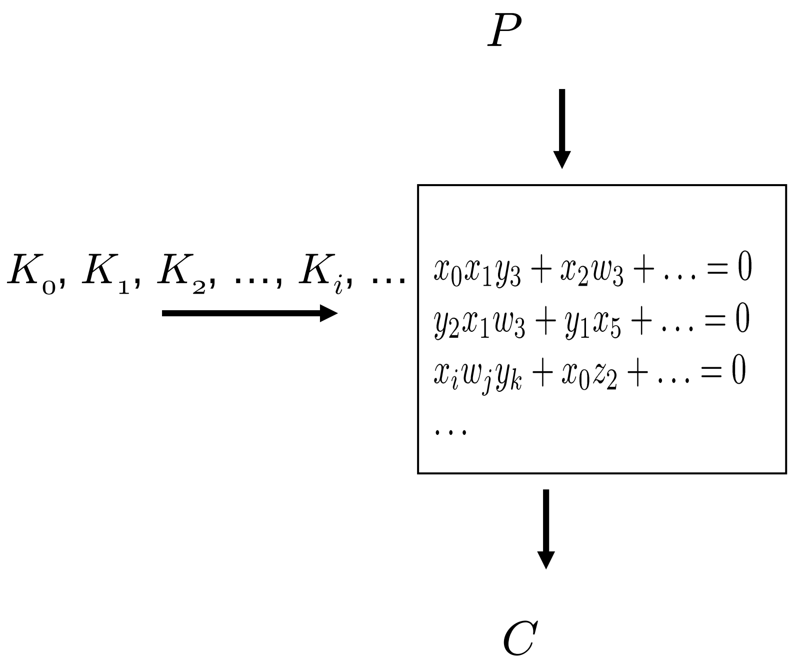
Some New Approaches for Algebraic Cryptanalysis

- Can SAT-Solver based attacks be considered "algebraic attacks"?
  - Variables are assigned values.
  - Consistency is checked.
  - If wrong, it learns why (and add the condition).

Some New Approaches for Algebraic Cryptanalysis

- Intelligent Exhaustive Key Search:




P
$K_0, K_1, K_2, \ldots, K_i, \ldots$
$x_0x_1y_3 + x_2w_3 + \ldots = 0$
$y_2x_1w_3 + y_1x_5 + \ldots = 0$
$x_iw_jy_k + x_0z_2 + \ldots = 0$
$\ldots$
C

Conclusions

- Many interesting problems in this area.
- Algebraic Attacks have been receiving a lot of attention.
  - For stream ciphers, it has already been incorporated into design criteria.
  - For block cipher, it still not very well understood (we are not sure of its merits and limitations).
  - New methods arising (with more success).
    - Possible direction to go: combination of different methods.
  - Hash Functions?