Cryptanalysis

Lecture 9: Introduction to Algebraic Attacks

John Manferdelli
jmanfer@microsoft.com
JohnManferdelli@hotmail.com

Algebraic Attacks

- As we've seen, ciphertext can be expressed as algebraic function of keys and plaintext (Lagrange Interpolation Theorem). Key bits may be expressible as functions of plain and cipher texts.
  - These are easy to solve if the equations are linear even for very large key spaces.
  - These are very hard to solve if the equations are even quadratic (NP-hard in fact, see "General System of Quadratic Equations" slide).
- General problem is "Find one solution of a system of *m* equations in *n* variables of bounded degree, D, over K (usually finite)."

$$\boldsymbol{\Sigma}_{\mathbf{b}} \; a_b \, \boldsymbol{x}^{\boldsymbol{b}} + c_i = 0, \quad \boldsymbol{x}^{\boldsymbol{b}} = x_1^{b1} x_2^{b2} \ldots x_n^{bn}, \quad \Sigma_i \, b_i \square \, D.$$

- We refer to this problem as SolveAlgebraic(K,D,m,n) and often abbreviate equations as $EQ_j(\boldsymbol{x}) = 0$.

General System of Quadratic Equations

- MQ: solve general system of *m* quadratic equations in *n* variables over K:

$$\Sigma_{1\leq j \leq k \leq n} a_{ijk} x_j x_k + \Sigma_{1\leq j \leq n} b_{ij} x_j + c_i = 0$$

  denoted by l_i for $1 \leq i \leq m$.
- MQ is an *NP-hard* even over a small finite field such as K=*GF*(2).

  Proof over GF(2); map 3-SAT → cubics →quadratics.

3 SAT	Cubic/GF(2)
0= x□y□z	0=xyz+xy+yz+xz+x+y+z
1=¬t	1=1+t

Finally, add equations $y_{ij}=x_i x_j$, $0=y_{ij}-x_i x_j$. This establishes correspondence.

Techniques for solving equations

1. Linear equations: Gaussian elimination, LU.
2. Berlekamp's Algorithm (single variable)
3. Linearization
4. Resultants and elimination
5. Grobner basis and elimination
6. Transforming to satisfiability instance and use SAT solver.

Resultants and results involving them

- **Theorem:** If $f_v(x)= v_n x^n + \ldots + v_0$ and $g_w(x)= w_m x^m + \ldots + w_0$. Then
  1. □□$□_{v,w}(x)$□□$□_{v,w}(x)$: $□_{v,w}(x)\, f_v(x) + □_{v,w}(x)\, g_w(x)= R(v,w)$.
  2. $R(v,w)=v_n{}^m w_m{}^n □_{I<j} (t_i - u_j)$, where t_i, u_j are roots of $f_v(x)$, $g_w(x)$ respectively.
  3. R(v, w) is 0 if and only if equations have common solution.
  4. $Res(f_1 f_2, g)=Res(f_1, g)\, Res(f_2, g)$

- **Theorem:** If $f_1, \ldots, f_r \in F[x_1 \ldots, x_n]$ has no common zeros, □□$A_1, \ldots, A_r$ such that □□$A_i f_i=1$. [This kind of thing should ring a bell.]

- **Nullstellensatz:** If $f(x_1 \ldots, x_n) \in F$ vanishes at all the common zeros of $f_1(x_1, \ldots, x_n), \ldots, f_r(x_1, \ldots, x_n)$ in every extension of F, then $f_k(x_1, \ldots, x_n) \in (f_1(x_1, \ldots, x_n), \ldots, f_r(x_1, \ldots, x_n))$ for some k.

Proof of basic formula

$$
\begin{pmatrix} x^{m-1}f_v(x) \\ x^{m-2}f_v(x) \\ \ldots \\ f_v(x) \\ x^{n-1}g_w(x) \\ x^{n-2}g_w(x) \\ \ldots \\ g_w(x) \end{pmatrix}
=
\begin{pmatrix}
v_n & v_{n-1} & \ldots & v_0 & 0 & \ldots & 0 & 0 \\
0 & v_n & v_{n-1} & \ldots & v_0 & 0 & \ldots & 0 \\
0 & \ldots & & \ldots & & \ldots & & \ldots \\
\ldots & 0 & 0 & 0 & v_n & v_{n-1} & \ldots & v_0 \\
w_m & w_{m-1} & \ldots & w_0 & 0 & 0 & \ldots & 0 \\
0 & w_m & w_{m-1} & \ldots & w_0 & 0 & \ldots & 0 \\
\ldots & 0 & \ldots & \ldots & \ldots & & & \\
0 & 0 & \ldots & 0 & w_m & w_{m-1} & \ldots & w_0
\end{pmatrix}
\begin{pmatrix} x^{m+n-1} \\ x^{n+m-2} \\ \ldots \\ \ldots \\ \ldots \\ x^2 \\ x \\ 1 \end{pmatrix}
$$

- Let the column vectors be $C_{m+n-1} \ldots C_0$, $C= (x^{m-1}f_v(x), ..., g_w (x))^T$.
- $C= C_{m+n-1}x^{m+n-1} + \ldots + C_0$. Now solve for 1.
- $1= \det(C_{m+n-1} \ldots C_1, C) \det(C_{m+n-1} \ldots C_1, C_0)$.
- So $\square_{v,w}(x) f_v(x) + \square_{v,w}(x) g_w(x)= R(v,w)$. Note: $R(v,w)$ does not contain x so, considering the function field adjoining the u and c, we get the Bezout form.

Example

- $f(x,y)= xy-1=0$
- $g(x,y)= x^2+y^2-4=0$

- Res(f,g,x)= det(

$$\begin{matrix} y, & 0, & 1, \\ -1, & y, & 0, \\ 0, & -1, & y^2-4 \end{matrix})$$

Multipolynomial resultants

Division Algorithm for many variables

- Division Algorithm analogous to a(x)=b(x)q(x)+r(x) in univariate case but degree is inadequate.
- Fix a monomial order $\leq$ for terms in $x_1, x_2, \ldots, x_n$. Example: Lex order $\square=(\square_1,\ldots,\square_n)$, $\square=(\square_1,\ldots,\square_n)$, $\mathbf{x}^{\square}\geq\mathbf{x}^{\square}$ iff leading term of $\square-\square$ is positive.

- Order relation must have the following two properties:
  1. If $\mathbf{x}^{\square}\geq\mathbf{x}^{\square}$ then $\mathbf{x}^{\square}\mathbf{x}^{\square}\geq\mathbf{x}^{\square}\mathbf{x}^{\square}$.
  2. The set of orders has a minimal element.

Division Algorithm for many variables

- Denote leading term of f under this order as $in_{\leq}(f)$. The division algorithm for f with respect to the monomial order produces $f(x) = a_1(\mathbf{x}) f_1(\mathbf{x}) + \ldots + a_m(\mathbf{x}) f_m(\mathbf{x}) + r(\mathbf{x})$ where r=0 or r is a linear combination of monomials none of which are divisible by $in_{\leq}(f_i)$. This is written as $r = \{f^F\}$. In general, the result depends on the ordering of the $f_i(x)$.

- LT(f) means "leading term." LM(f) is "leading monomial." If $f(x,y) = 2x^3y^4+3xy$, $LT(f) = 2x^3y^4$, and $LM(f)=x^3y^4$.

- Unlike the univariate case, the division algorithm over an arbitrary basis $<f_1, \ldots, f_n>$ may yield non-zero r(x) even if there are $a_i(\mathbf{x})$: $a_1(\mathbf{x}) f_1(\mathbf{x}) + a_2(\mathbf{x}) f_2(\mathbf{x}) = f(\mathbf{x})$, because no $LM(f_i)$ divides any monomial of $r(\mathbf{x})$. An example is $f(x)=1$, $f_1(x)=x+1$, $f_2(x)=x$. Grobner basis have the important property that if $<g_1(x), \ldots, g_r(x)>$ is such a basis, $<LT(g_i)> = <LT(I)>$.

Division Algorithm

- Hilbert Basis Theorem: Every $I \subset k[x_1,\ldots,x_n]$ has a finite generating set.

- Grobner condition: $<LT(g_1), \ldots, LT(g_s)> = <LT(I)>$.
- If $G=<g_1, \ldots, g_s>$ is a Grobner Basis and $f(x)= a_1(x)\ g_1(x) + \ldots + a_m(x)\ g_m(x) + r(x)$ then every term of $r(x)$ is divisible by none of $LT(g_s)$.
- $\mathbf{x}^{\square}$= LCM(LM(f), LM(g)). S(f, g)= $\mathbf{x}^{\square}/LT(f)+ \mathbf{x}^{\square}/LT(g)$. Used in constructing Grobner basis.
- Let I be a polynomial ideal. □□$G=<g_1, \ldots, g_s>$ a Grobner Basis for I iff for all $x \neq y$, $REM(S(g_i,g_j))=0$. $f \in I$ iff $f^G=0$.
- Example:
  - $f=x^3y^2-x^2y^3+x$, $g= 3x^4y+y^2$ in $R[x,y]$.
  - $\mathbf{x}^{\square}=x^4y^2$.
  - $S(f,g)= -x^3y^3+x^2-(1/3)y^3$.

Grobner

- Grobner Basis: A finite subset $G = \{g_1, g_2, ..., g_s\}$ is a Grobner basis for an ideal I with respect to the monomial order $\leq$ if $<in_{\leq}(g_1), in_{\leq}(g_2), ..., in_{\leq}(g_s)> = < in_{\leq}(I)>$. Equivalently, if $f \in I$, $in_{\leq}(g_i) | in_{\leq}(f)$ for some i.

- Theorem: If G is a Grobner basis f^G is independent of the order of the $f_i(x)$. If G is a Grobner basis and $I=<G>$, $f \in I$ iff $f^G = 0$.

- Consequence: Every ideal has a Grobner basis.
- There is a computationally efficient way to find these bases!
- Note connection between Grobner and Hilbert's original proof of the Hilbert Basis Theorem.

Buchberger

Input: $F=<f_1, f_2, ..., f_m>$. Output: Grobner Basis $G= \{g_1, g_2, ..., g_s\}$.

// see definition of S(p,q) in earlier slide.

$G \leftarrow F$;

Do {

    $G' \leftarrow G$;

    for($p,q \in G'$, $p\neq q$) {

        Compute S(p,q);

        $r \leftarrow REM(S(p,q), G')$;

        if($r \neq 0$)

            $G \leftarrow G' \cup \{r\}$;

        }

} while(G!=G')

- Theorem: Foregoing algorithm yields Grobner Basis.

Polynomial Problems

- **Ideal Membership:** Does every ideal $I \subset k[x_1,\ldots,x_n]$ have a finite generating set.
- **Ideal Description:** Given $f \in k[x_1,\ldots,x_n]$ and an ideal $I=<f_1, \ldots, f_s>$ determine if $f \in I$.
- **Implicitization:** Let V be a subset of k^n given parametrically as:

$$x_1 = g_1(t_1,\ldots,t_m)$$
$$x_2 = g_2(t_1,\ldots,t_m)$$
$$x_n = g_n(t_1,\ldots,t_m)$$

  Find the generating polynomials and conversely.
- Note the cryptographic application of this last problem.
- All these are “solved” by the Grobner basis.

Elimination Ideals

- $I_s = I \cap k[x_{s+1}, \ldots, x_n]$
- If G is a Grobner basis for I with respect to lex then $G_s = G \cap k[x_{s+1}, \ldots, x_n]$ is a Grobner basis for the sth elimination ideal.
- If k is algebraically closed, then a partial solution, $(a_{l+1}, a_{l+2}, \ldots, a_n)$ is $V(I_{l-1})$.
- Successively looking at the elimination ideals $I_1, \ldots, I_n$ reduces each set of variables one at a time. When we have one variable left, we can solve in the usual way.

Example Grobner and elimination ideal

- $x^2+y^2+z^2=4$, $x^2+2y^2=5$, $xz=1$
- $G=\{2z^3-3z+x, -1+y^2-z^2, 1+2z^2-3z^4\}$
- $z= \pm 1, \pm 1/\sqrt{2}$

- $f(x)=x^3+x-1$, $g(x)=2x^2+3x+7$

Grobner Examples

- Example 1
  - $I=<x^2+y^2+z^2=1, x^2+z^2=y, x=y>$
  - G: $g_1= x-z$, $g_2= -y+2z^2$, $g_3=z^4+(1/2)z^2-(1/4)$.
  - $z= \pm(1/2) \sqrt{(\pm \sqrt{5}-1)}$

- Example 2
  - $x^2+y+z=1, x+y^2+z=1, x+y+z^2=1.$
  - $I=< x^2+y+z-1, x+y^2+z-1, x+y+z^2-1>$
  - $g_1= x+y+z^2-1$
  - $g_2= y^2-y+z^2+z$
  - $g_3= 2yz^2+z^4-z^2$
  - $g_4= z^6-4z^4+4z^3-z^2$

Solving SolveAlgebraic(K,D,m,n)

- A general solving technique involves the Grobner Basis and found by Buchberger's Algorithm which is doubly exponential time in the worst case since the monomial grow very rapidly and singly exponential time on average.
- This is not practical for n>15.
- To solve larger systems we must take advantage of special properties of the system like sparseness by using "nice" mappings to SAT or "linearized" equations. We can do this with an overdefined set of equations (m>n).
- Note first that if we pick m random equations m>n they will likely be inconsistent.

SAT and equation solving

- $x_1 + x_2 \rightarrow (x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)$
- $x_1 x_2 \rightarrow x_1 \wedge x_2$
- $x_1+x_2+x_3+x_4=0$. Must add variables to avoid the exponential explosion in terms. $x_1+x_2+x_3+x_4=0 \rightarrow$
  1. $y_1+x_1+x_2= y_2$
  2. $y_2+x_3+x_4= 0$.

SAT equation solving example

1. $x_1+x_2x_3=0 \rightarrow \neg((x_1\wedge \neg(x_2\wedge x_3)) \vee(\neg x_1\wedge (x_2\wedge x_{3)})))$
2. $x_1x_3=1. \rightarrow x_1\wedge x_3$

- 1 simplifies to $\neg(x_1\wedge\neg(x_2\wedge x_3)) \wedge\neg (\neg x_1\wedge (x_2\wedge x_3)))) \rightarrow$ □$\neg x_{1)}\vee(x_2\wedge x_3) \wedge (x_1\vee\neg x_2\vee\neg x_3)$ which is satisfied by x_1= T, x_2=T□x_3=T. This translates into x_1= 1, x_2=1□x_3=1 and indeed $1+ 1\cdot1=0$ and $1\cdot1=1$.

- There are standard SAT packages that work very well when the number of clauses compared to variables is small or very large (MiniSAT).

Review: Solving Linear Equations

Solve the following over GF(7)

$3x + y + 4z + 1 = 0$ … [1]

$6x + 5y + 3z + 6 = 0$ … [2]

$x + 4y + 2z + 5 = 0$ … [3]

Gaussian Elimination

$x + 4y + 2z + 5 = 0$ … [3]

2y + 5z + 4 = 0 … [3]+[2]

-4y + 5z + 0 = 0 … [1]-3 x [3]

-y + 4 = 0 → y=4, z= -1, x=2

Idea: Linearization of Quadratics

Solve

$$x^2 + 4y^2 + z^2 + 5xy + 2xz + 6yz + 5x + 3y + 5z + 1 = 0$$
$$3x^2 + 2y^2 + 3z^2 + 4xy + 6xz + 2yz + 6x + 4y + 3z + 2 = 0$$
$$2x^2 + 3y^2 + 2z^2 + 5xy + 2yz + 4x + y + z + 4 = 0$$
$$6x^2 + 3y^2 + 3z^2 + 5xz + yz + 5y + 2z + 2 = 0$$

Linearize by assigning quadratic monomial terms to new variables:

$x^2 \rightarrow A$, $y^2 \rightarrow B$, $z^2 \rightarrow C$, $xy \rightarrow D$, $xz \rightarrow E$, $yz \rightarrow F$

$$A + 4B + C + 5D + 2E + 6F + 5x + 3y + 5z + 1 = 0$$
$$3A + 2B + 3C + 4D + 6E + 2F + 6x + 4y + 3z + 2 = 0$$
$$2A + 3B + 2C + 5D + 2F + 4x + y + z + 4 = 0$$
$$6A + 3B + 3C + 5E + F + 5y + 2z + 2 = 0$$

Problem: Find more equations so system is overdetermined.

Adding Equations by Relinearization

- If # {variables} >> # {equations}, there are too many solutions to the system of linear equations.
- Consider each quadratic monomial as a new variable and linearize again with more variables:
  - $(ab)(cd) = (ac)(bd) = (ad)(bc)$
  - $(ab)(cd)(ef) = (ad)(cf)(eb) = \ldots$

- Kipnis and Shamir, *Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization*, Crypto '99.

- Toy example from [CKPS] cited later.

“Toy” Example

1. $x_1^2+\square x_1x_2=\square$
2. $x_2^2+\square x_1x_2=\square$

D=4

3. $x_1^4+\square x_1^3x_2=\square x_1^2$
4. $x_1^2x_2^2+\square x_1^3x_2=\square x_1^2$
5. $x_1^2x_2^2+\square x_1^3x_2=\square x_2^2$
6. $x_2^4+\square x_1^3x_2=\square x_2^2$
7. $x_1^3x_2+\square x_1^2x_2^2=\square x_1x_2$
8. $x_1x_2^3+\square x_1^2x_2^2=\square x_1x_2$

$x_1x_2=\square\ \square\square\ \square\ x_1^2\ \square\square$

$x_2^2=\square\square\square\ \square\square\square\square\ \square\square\ \square\square\square\square\ x_1^2$

$x_1^3x_2=\square\square\ \square\square\square\ x_1^2-x_1^4(1/\square\square$

$x_1^2x_2^2=\square\square\square\ \square\ \square\square\square x_1^2+\square\square\square\square x_1^4$

$x_1x_2^3\square\ \square\square\ \square\square\square\ \square\square\ \square\square\ \square^{\square}\square\square\ \square\square\square\ \square\square\square x_1^2+$
$\square\square^{\square}\square\square\ \square x_1^4$

$x_2^4\square\ \square\square^{\square}\square\ \square\square\ \square\square\square\square\ \square\square\ \square\square\square\square\square\ \square\ \square\square^{\square}\square$
$\square\ \square^{\square}\square\square\ \square x_1^2+\square\square^{\square}\square\square\ \square x_1^4$

From 5:

$\square^2+x_1^2\square\square\ \square\square\square\ \square\square^{\square}\square\ \square\square\ \square\square x_1^4(\square\square\ \square\square)=0$

Relinearization Procedure

- Use first Linearization to solve m linear equations in (n(n+1)/2 variables
- $y_{ij} = x_i\ x_j$
- Express $y_{ij} = \sum_{[k=1,\ l]} c_{ij}^{(k)}\ t_k$
- Degree 4 relinearization

n	m	l	n’	m’
6	8	13	104	105
8	12	24	324	336
10	16	39	819	825
15	30	90	4185	4200

n’: # variables in final eqn
m’: # equations in final

- Where do the extra equations come from?
  - $(x_a x_b)(x_c x_d)\ \ldots = (x_{a'} x_{b'})\ \ldots$

XL

- XL – E**X**tended **L**inearization
  - [CKPS] N. Courtois, A. Klimov, J. Patarin, and A. Shamir, *Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations*, Eurocrypt 2000.
- Extension of linearization idea.
- Appears to be polynomial when $m > \epsilon n^2$ and subexponential when $m > n+1$.

Basic XL algorithm ($l_j(X)$, quadratic)

1. Generate all $\prod_{[j=1,k]} x^{\mathbf{k}}\, l_j(X)$, $k \leq D-2$.
2. Linearize
3. Solve
4. Repeat

XL Algorithm

- Take all monomials $\boldsymbol{x}^{\boldsymbol{b}} = x_1^{b_1} x_2^{b_2} \ldots x_n^{b_n}$ with total degree k, $k \le D - 2$.
  - There are ${}_{n+1}H_{D-2} = {}_{D-2+n}C_{D-2}$ such monomials.
- Generate all equations $\boldsymbol{x}^{\boldsymbol{b}}\, l_i$.
  - There are $R = m \times {}_{D-2+n}C_{D-2}$ such equations.
  - There must be linearly dependency among them if $D \geq 4$.
  - Denote $I = \#$ {linearly independent equations}.
- Treat all monomials of total degree $\le D$ as variables.
  - There are $T = {}_{n+1}H_D = {}_{D+n}C_D$ of them.
- Perform Gaussian elimination. Keep $x_i^{\,d}$ last.
- If $T - I \leq D$, the last row represents an equation in $x_n^D, \ldots, x_n^2, x_n$, 1.
- Solve the univariate equation in x_n.
- Solve $x_{n-1}, \ldots, x_1$ recursively.

XL

- Consider previous system of quadratic equations:

$$l_1 : \quad x^2 + 4y^2 + z^2 + 5xy + 2xz + 6yz + 5x + 3y + 5z + 1 = 0$$

$$l_2 : \; 3x^2 + 2y^2 + 3z^2 + 4xy + 6xz + 2yz + 6x + 4y + 3z + 2 = 0$$

$$l_3 : \; 2x^2 + 3y^2 + 2z^2 + 5xy \qquad + 2yz + 4x + y + z + 4 = 0$$

$$l_4 : \; 6x^2 + 3y^2 + 3z^2 \qquad + 5xz + yz \qquad + 5y + 2z + 2 = 0$$

- Try degree $D = 3$:
  - Multiply each EQ_i by x, y, z respectively.
  - Linearize: Consider all monomials as variables.
    - How many equations now? $4 \times 4 = 16$
    - How many variables now? ${}_4H_3 = {}_6C_3 = 20$

Matrix of Coefficients

	x^2y	x^2z	xy^2	xyz	xz^2	y^2z	yz^2	xy	xz	yz	x^3	x^2	x	y^3	y^2	y	z^3	z^2	z	1	
[	0	0	0	0	0	0	0	5	2	6	0	1	5	0	4	3	0	1	5	1	]
[	0	0	0	0	0	0	0	4	6	2	0	3	6	0	2	4	0	3	3	2	]
[	0	0	0	0	0	0	0	5	0	2	0	2	4	0	3	1	0	2	1	4	]
[	0	0	0	0	0	0	0	0	5	1	0	6	0	0	3	5	0	3	2	2	]
[	5	2	4	6	1	0	0	3	5	0	1	5	1	0	0	0	0	0	0	0	]
[	1	0	5	2	0	6	1	5	0	5	0	0	0	4	3	1	0	0	0	0	]
[	0	1	0	5	2	4	6	0	5	3	0	0	0	0	0	0	1	5	1	0	]
[	4	6	2	2	3	0	0	4	3	0	3	6	2	0	0	0	0	0	0	0	]
[	3	0	4	6	0	2	3	6	0	3	0	0	0	2	4	2	0	0	0	0	]
[	0	3	0	4	6	2	2	0	6	4	0	0	0	0	0	0	3	3	2	0	]
[	5	0	3	2	2	0	0	1	1	0	2	4	4	0	0	0	0	0	0	0	]
[	2	0	5	0	0	2	2	4	0	1	0	0	0	3	1	4	0	0	0	0	]
[	0	2	0	5	0	3	2	0	4	1	0	0	0	0	0	0	2	1	4	0	]
[	0	5	3	1	3	0	0	5	2	0	6	0	2	0	0	0	0	0	0	0	]
[	6	0	0	5	0	1	3	0	0	2	0	0	0	3	5	2	0	0	0	0	]
[	0	6	0	0	5	3	1	0	0	5	0	0	0	0	0	0	3	2	2	0	]

Gaussian Elimination

$$
\begin{array}{cccccccccccccccccccc}
x^2y & x^2z & xy^2 & xyz & xz^2 & y^2z & yz^2 & xy & xz & yz & x^3 & x^2 & x & y^3 & y^2 & y & z^3 & z^2 & z & 1 \\
\end{array}
$$

$$
\begin{array}{l}
[\ 5\ 2\ 4\ 6\ 1\ 0\ 0\ 3\ 5\ 0\ 1\ 5\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ ] \\
[\ 0\ 1\ 0\ 5\ 4\ 6\ 1\ 3\ 6\ 5\ 4\ 6\ 4\ 4\ 3\ 1\ 0\ 0\ 0\ 0\ ] \\
[\ 0\ 0\ 3\ 6\ 0\ 3\ 4\ 1\ 2\ 6\ 0\ 5\ 6\ 2\ 5\ 4\ 0\ 0\ 0\ 0\ ] \\
[\ 0\ 0\ 0\ 1\ 0\ 2\ 3\ 4\ 5\ 3\ 0\ 2\ 1\ 2\ 4\ 2\ 0\ 0\ 0\ 0\ ] \\
[\ 0\ 0\ 0\ 0\ 5\ 5\ 5\ 4\ 6\ 5\ 3\ 1\ 3\ 3\ 4\ 6\ 1\ 5\ 1\ 0\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 5\ 3\ 2\ 4\ 0\ 0\ 1\ 4\ 1\ 2\ 1\ 0\ 2\ 6\ 0\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 6\ 4\ 2\ 0\ 5\ 1\ 5\ 6\ 5\ 6\ 1\ 0\ 0\ 0\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 5\ 0\ 2\ 0\ 2\ 4\ 0\ 3\ 1\ 0\ 2\ 1\ 4\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 5\ 1\ 0\ 6\ 0\ 0\ 3\ 5\ 0\ 3\ 2\ 2\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 2\ 0\ 4\ 0\ 0\ 3\ 0\ 0\ 2\ 4\ 2\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 6\ 0\ 6\ 3\ 1\ 0\ 4\ 1\ 6\ 1\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 2\ 1\ 0\ 0\ 0\ 0\ 4\ 3\ 1\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 3\ 1\ 2\ 4\ 2\ 0\ 1\ 0\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 4\ 6\ 0\ 0\ 1\ 5\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 6\ 3\ 6\ 1\ 5\ 5\ ] \\
[\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 5\ 2\ 1\ 6\ ]
\end{array}
$$

XL at Degree *D*

- XL only operates on *determined* (*n* = *m*) or *over-determined* (*n* < *m*) systems.
- Select an appropriate degree *D* before performing XL.
- Given a large system of equations, it is difficult to find optimal *D*.
- XL succeeds when m□□n²/(D(D-1)). D□□n/√m
- [CKPS] gives a rough estimate and some simulation results.
  - m=n, D~2^n
  - m=n+1, D~n
  - m=n, D~√n
  - m=□n^2, D~ 1/√ϵ

Time Complexity of XL

- Denote $E(N, M)$ the complexity of elimination on N variables and M equations.

- $C_{\mathrm{XL}} = E(T, R) = E\left(\binom{D+n}{D}, m\binom{D-2+n}{D-2}\right)$
  - T = The number of monomials, including 1.
  - R = The number of equations.

- $T^{2.8}$ was claimed for $E(T, R)$ under Strassen's blocking elimination algorithm.
  - Not really suited to XL implementation.

Algebraic description of AES

- M= CRL, is the linear map over GF(2) representing mix column, shift row and the linear equation.
- Minimal polynomials C: (x^4+1),R: (x^4+1),L: $(x+1)^3$,C: $(x+1)^{15}$.
- Single AES round is $\square_i(x)= M(x)^{-1}+(k)_i+63$
- Full AES (128) is:
  - $w_0= p + (k)_0 + 63$
  - $w_i= M(w_{i-1})^{-1}+(k)_i+63$, i=1,2,…9
  - $c=M^*(w_9)^{-1}+(k)_{10}+63$

- Rank of system is (equations)/(monomials).

Resulting algebraic description of AES

- If 8j+m component denoted by $v_{(j,m)}$.
  - $0=w_{0,(j,m)}+p_{(j,m)}+k_{0,(j,m)}$.
  - $0=x_{i,(j,m)}\ w_{i,(j,m)}+1$, i=1,2, …, 9.
  - $0=w_{i,(j,m)}+(M\ x_{i-1})_{(j,m)}+k_{i,(j,m)}$, i= 1,2, …, 9.
  - $0=c_{(j,m)}+(M^*\ x_9)_{(j,m)}+k_{10,(j,m)}$.

- This is a total of 10368 encryption equations over GF(2) involving 2560 state variables and 1728 key variables. The equations come from 6400 inversion equations, 1408 linear diffusion operations and 2560 field equations.
- We could also calculate the key schedule equations.

XL and AES from [CP]

- S Box is map from $GF(2^8) \rightarrow GF(2^8)$.
- Remaining operations are linear diffusion
- s=8 (size of substitution box), r=24, t=41.
- k_{ij}: key bits, i=1,2,…,N_r+1; B= 4N_b; j= 1,2,…, sB [N_r=10…14]
- z_{ij}: output bits $x_{i+1,j} = z_{ij} \oplus k_{ij}$
- Number of monomials: $t << {}_sC_d$
- S-box: 8 bilinear equations, 7 hold with p=1, one with p=255/256
- Rijndael can be solved with m= 8000 over n=1600.
- XSL
  - X: xor key
  - S: substitution
  - L: linear mixing

Typical problem of algebraic cryptanalysis

- Solve a system of black box polynomial equations over GF(2):

  $P_1(x_1 \ldots x_n v^1{}_1 \ldots v^1{}_m)=0$

  $P_2(x_1 \ldots x_n v^2{}_1 \ldots v^2{}_m)=1$

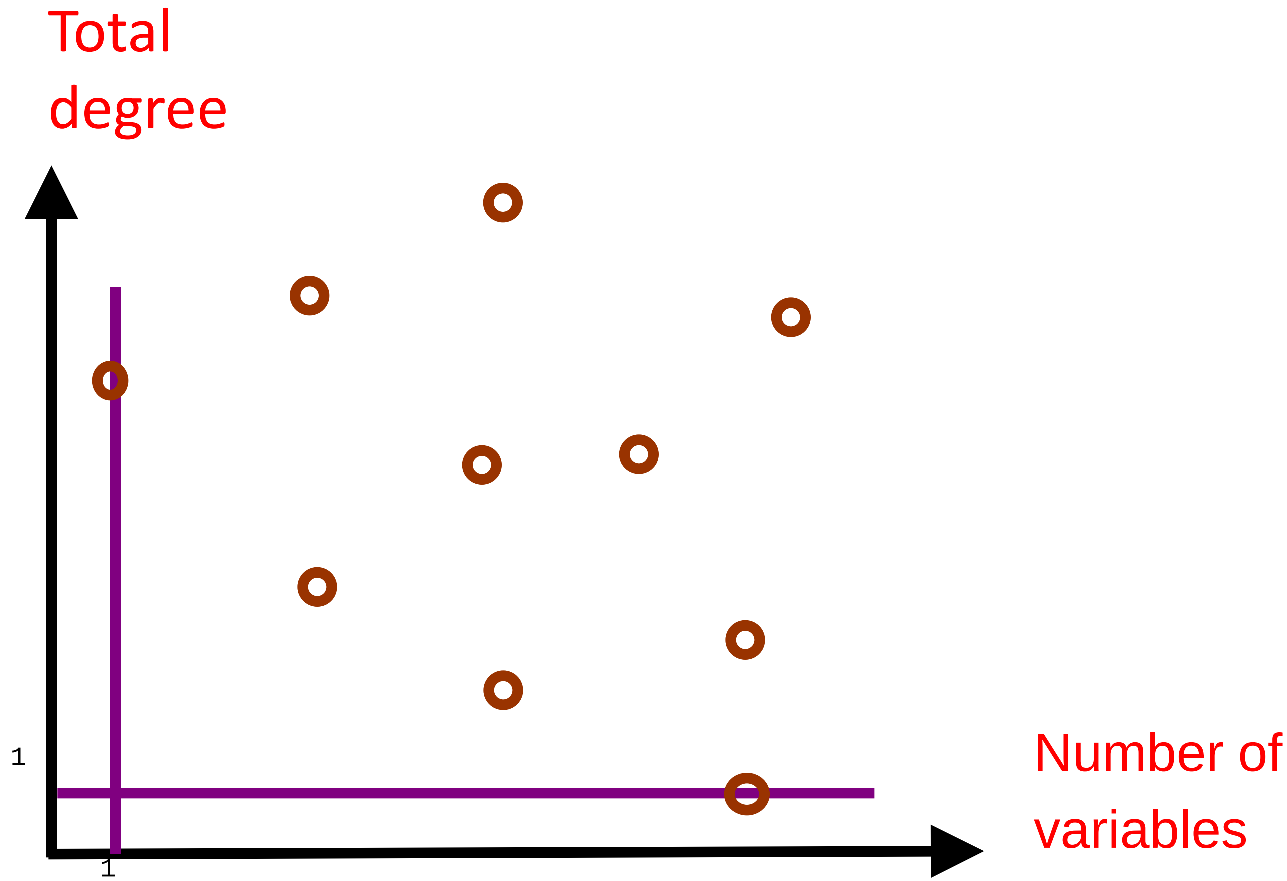
  $P_3(x_1 \ldots x_n v^3{}_1 \ldots v^3{}_m)=0$

  …

  in which the fixed key variables x_i are unknown, and the various plaintext/IV variables $v^j{}_j$ are known
- The problem is NP-hard and exceedingly difficult in practice, even with explicitly given polynomials

Source: Adi Shamir.

The only easily solvable cases of simultaneous algebraic equations



Source: Adi Shamir.

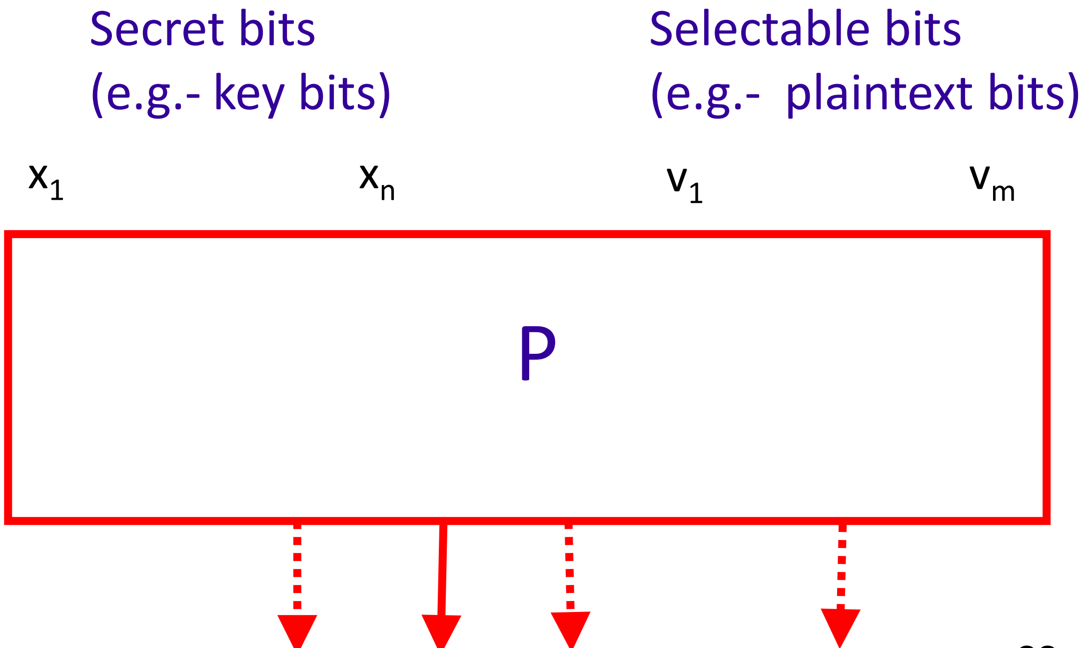
Characteristics of cryptographically defined polynomials

- Consider the case of the AES, with 128 key and 128 input bits with Multivariate polynomials in fully expanded Algebraic Normal Form
  - These polynomials are huge, and can not be explicitly defined, stored, or manipulated with a feasible complexity.
  - The data available to the attacker will typically be insufficient to interpolate their coefficients from their output values.

Source: Adi Shamir.

Cryptographic scheme as "black box"

- Each output bit is some multivariate polynomial $P(x_1,\ldots x_n,v_1,\ldots v_m)$ over GF(2) of secret variables x_i (key bits), and public variables v_j (plaintext bits in block ciphers, IV bits in stream ciphers)



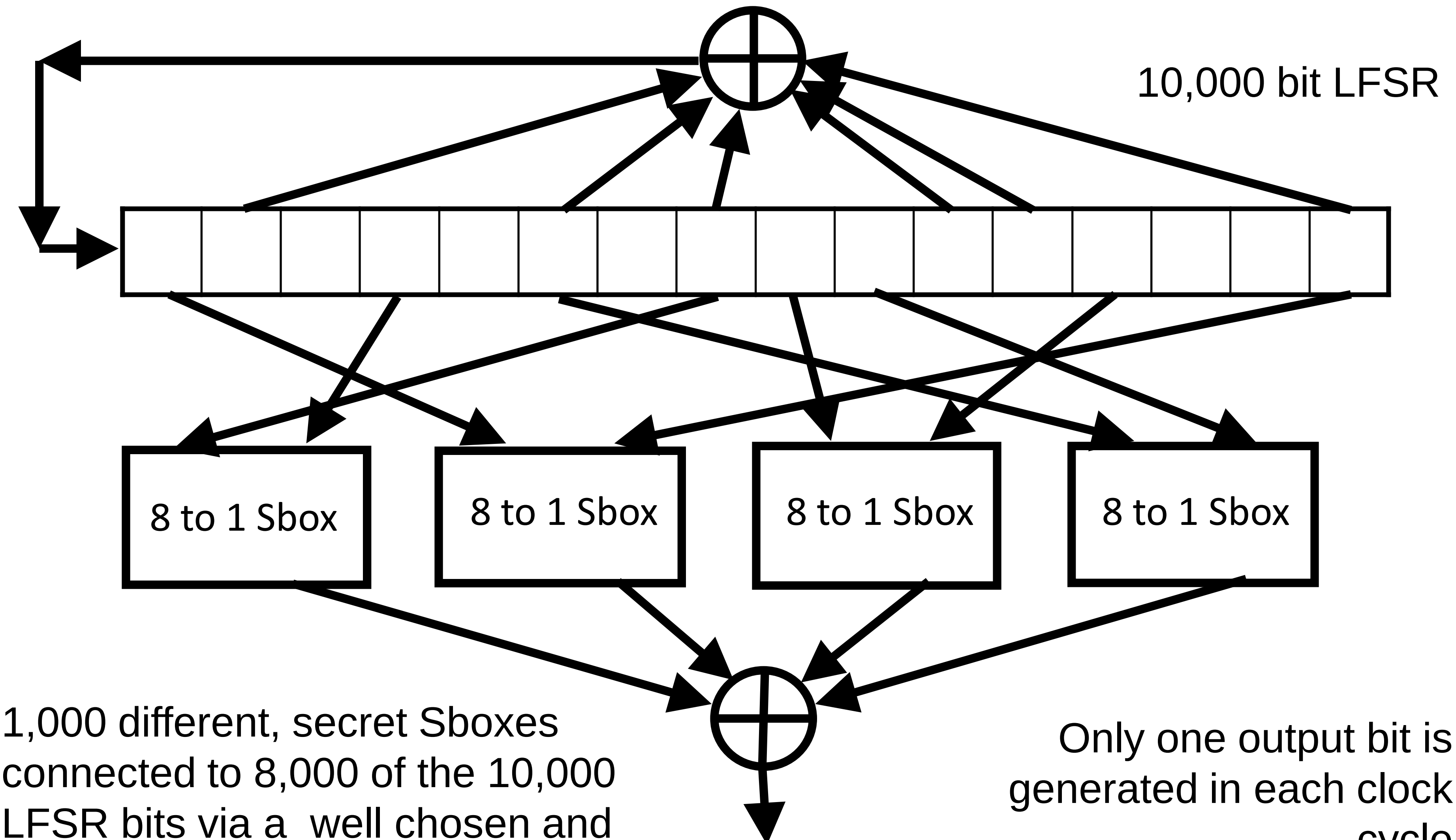
Many of the following slides are from or inspired by a talk of Adi Shamir. Adi kindly provided a copy.

The cube attack (Dinur&Shamir)

- Algebraic attack on “black box” ciphers that is much faster than general equation solving (in special cases).
- Applies when encryption equations are derived from a “low degree” sparse master polynomial.
- Attack will be demonstrated on an LFSR-based stream cipher with non-linear filter.
- Cryptanalyst knows the structure of cipher:
  - The schematic diagram
  - The size of the various components
- Cryptanalyst does not know the many details, for the “LSFR” example, cryptanalyst does not know:
  - The LFSR feedback function
  - The Sboxes
  - The LFSR/Sboxes connections
  - The quadratic key/IV mixing function

Source: Adi Shamir.

LFSR scheme

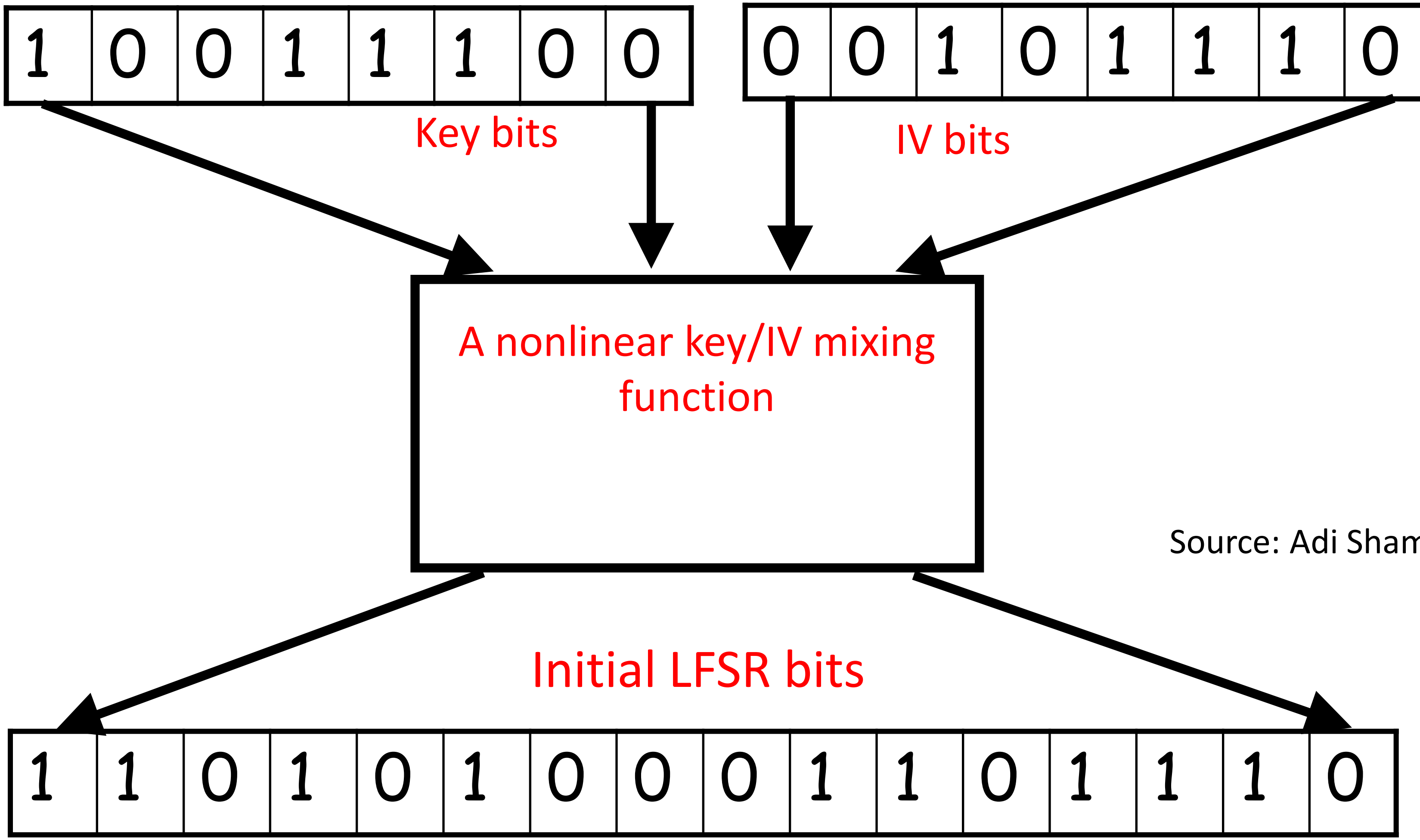


1,000 different, secret Sboxes connected to 8,000 of the 10,000 LFSR bits via a well chosen and secret bit permutation

Only one output bit is generated in each clock cycle

Source: Adi Shamir.

The initial loading of the LFSR



Source: Adi Shamir.

Source: Adi Shamir.

Further description

- We used a *random* dense secret *quadratic* mixing function on all the 10,000 key and IV bits for initial LFSR state.
  $x_i x_j + \ldots + x_k v_l + \ldots + v_m v_n + \ldots + x_p + \ldots + v_q + \ldots$
- We added a large and secret number of dummy initial LFSR steps which produce no output.
- We assume that each key can be used with at most 2^{20} IV's.
- We assume that for each IV only 1 output bit is known.
- The known output bit of the stream cipher is a multivariate polynomial P over GF(2) of the n=10,000 key variables x_i and IV variables v_j
- What is the degree d of this polynomial?
- The key/IV mixing function was chosen as a random dense quadratic mapping, the dummy and real LFSR steps re-randomize these polynomials but their degree remains 2.

Source: Adi Shamir.

Key exploited algebraic feature: low degree representations

- Each 8-bit to 1-bit Sbox is a dense polynomial of degree at most 8 over GF(2) in its input bits $z_1 \ldots z_{8.}$ (Ex: $z_1z_2z_3z_4z_5z_6z_7z_8 + z_2z_3z_6z_8 + \ldots$)
- Substituting the random quadratic polynomials and expanding , we can describe the output bit of each Sbox as the sum of terms of degrees at most 16 $z_t = x_ix_j + \ldots + x_kv_l + \ldots + v_mv_n + \ldots + x_p + \ldots + v_q + \ldots$.
- Each output bit is the sum of 1,000 such polynomials, and can be described as a random looking dense polynomial of degree at most 16 in the 10,000 input variables.
- This low degree representation will be the only weakness used by the new cube attack to extract the key.

Source: Adi Shamir.

Two stage attack

- A preprocessing phase (uses black box simulation):
  - The stream cipher is given as a black box. Attacker can obtain one bit of output for any chosen key and IV.
- An online phase (uses data from eavesdropping):
  - The stream cipher is given as a black box, with the key set to a secret fixed value. The attacker can obtain one bit of output for any chosen IV value.
- For an black box scheme represented by random polynomials of degree d in n input variables over GF(2):
  - The online stage takes $O(n2^{d-1}+n^2)$ bit operations.
  - The preprocessing stage is n times larger.
- In LFSR example (to follow), d=16 and $n=2^{13}$, so the running time of the attack is $2^{13}2^{15}+2^{26}$, which is about 2^{28}.

Source: Adi Shamir.

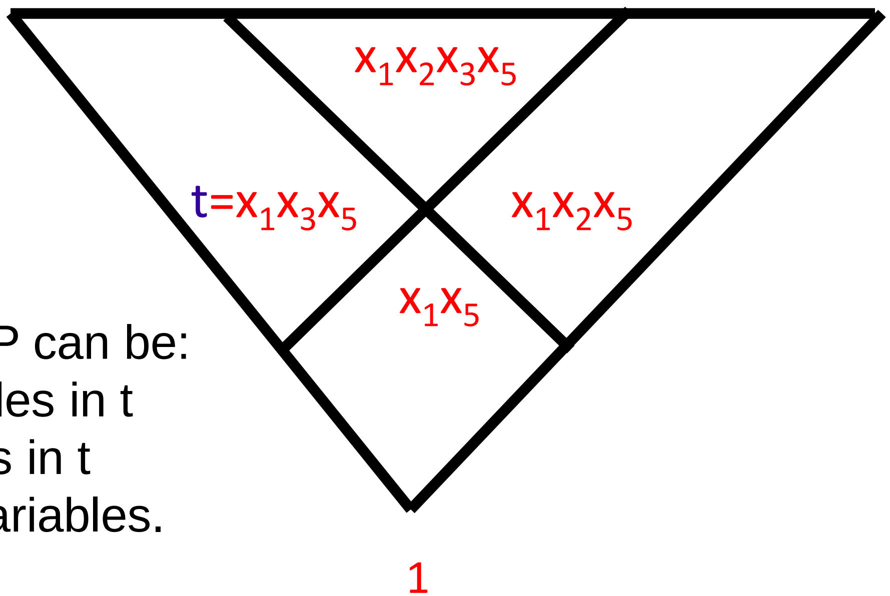
Small example of cube attack

- Suppose we have a dense master polynomial of degree d=3 over three secret variables x_1,x_2,x_3 and three public variables v_1,v_2,v_3:
  - $P(v_1,v_2,v_3,x_1,x_2,x_3)= v_1v_2v_3+v_1v_2x_1+v_1v_3x_1+v_2v_3x_1+v_1v_2x_3+v_1v_3x_2+ v_2v_3x_2+ v_1v_3x_3+v_1x_1x_3+v_3x_2x_3+x_1x_2x_3+v_1v_2+ v_1x_3+ v_3x_1+x_1x_2+x_2x_3+x_2+v_1+v_3+1$
- Summing the derived polynomials (over v_2 , v_3) with v_1=0, we get x_1+x_2. Similarly, summing with v_2=0, we get $x_1+x_2+x_3$ and summing with v_3=0 we get x_1+x_3.
- This give rise to three linear equations in the three secret variables x_i, which can be easily solved.

Source: Adi Shamir.

Why did the nonlinear terms in the sum?

- All the terms are the products of at most 3 of the 6 x_i and v_j variables. We sum over all the values of two v_j's
- Any term in the master polynomial P such as $x_1x_2v_1$ which contains the nonlinear product of two or more x_i in it, is missing at least one of the v_j that we sum over, and is thus added an even number of times modulo 2 to the sum.



Given P and t terms in P can be:
Supersets of the variables in t
Subsets of the variables in t
Incomparable sets of variables.

Source: Adi Shamir.

The superpoly of a term t in P

- For any polynomial P and term t, write $P=tP_t+Q$ where:
  - The variables in P_t are disjoint from those in t.
  - Each term in Q misses at least one variable from t.
- P_t is called the superpoly of t in P.
- A maxterm of P is any product t of v_j variables whose superpoly has degree 1 (i.e., is a linear or affine function which is not a constant).
- Example:
  - $P(x_1,x_2,x_3,x_4,x_5) = x_1x_2x_3 + x_1x_2x_4 + x_2x_4x_5 + x_1x_2 + x_2 + x_3x_5 + x_5+1$
  - Let $t=x_1x_2$, $P(x_1,x_2,x_3,x_4,x_5)= x_1x_2(x_3+x_4+1)+(x_2x_4x_5+x_3x_5+x_2+x_5+1)$
  - The superpoly of x_1x_2 in P is (x_3+x_4+1)

Source: Adi Shamir.

Main observation on the superpoly

- Theorem: The symbolic sum over GF(2) of all the derived polynomials obtained from a master polynomial P by assigning all the possible 0/1 values to the subset of variables in the term t is exactly the superpoly of t in P.
  - Proof: Let $P=tP_t+Q$. Any term t’ in Q misses at least one variable from t, and is thus added an even number of times. This cancels its sum modulo 2. Any term tt’ which contains a superset of the variables in t is zero for all the assignments of values to the variables in t, except when all of them are 1. In this case we add t’ once to the sum. The sum thus contains exactly the terms t’ in the superpoly of t in P.

Source: Adi Shamir.

Applying this to low degree "black box"

- Random polynomials of degree d are expected to have only maxterms of degree d-1. However, some polynomials have no maxterms and some maxterms can have considerably lower degrees.
- Even when P is huge, the linear superpoly of any maxterm t can be compactly represented.
- The master polynomial has about 2^{200} terms of degree at most 16 in 10,000 key and IV variables.
- Since P is sufficiently random, almost all the products of 15 IV variables are maxterms whose superpolys are linear combinations of the other variables.

Source: Adi Shamir.

Applying the cube attack to our full stream cipher example

- The master polynomial has about 2^{200} terms of degree at most 16 in 10,000 key and IV variables
- Since P is sufficiently random, almost all the products of 15 IV variables are maxterms whose superpolys are linear combinations of the other variables
- How much data
  - Consider the 20 dimensional boolean cube in which the cryptanalyst sets the 20 least significant IV bits to all their possible values, leaving all the other x_i and v_j variables fixed
  - There are 15,504 possible terms of degree 15 defined by these 20 variables, and more than 10,000 of them are maxterms which yield linear equations in the 10,000 key variables

Source: Adi Shamir.

Preprocessing Stage

- The derived polynomials cannot be explicitly generated or symbolically summed from the master polynomial with feasible complexity.
- The preprocessing phase (executed only once for each cryptosystem) finds the maxterms and their superpolys. Note that during preprocessing, the attacker is allowed to choose both the key and IV variables.
- For each candidate maxterm t, the attacker chooses pairs of values for all the other variables X' and X", and verifies that the numerical values of the subcube sums satisfy the linearity test: $P_t(X') + P_t(X'') = P_t(X'+X'') + P_t(0)$.
- If the test succeeds multiple times, the attacker finds the actual linear superpoly by checking the numeric effect of flipping each key bit x_i:

Source: Adi Shamir.

When does attack succeed?

- The attack is provably successful against sufficiently random multivariate polynomials in which:
  - Each term occurs with probability 0.5
  - Each term of maximum degree d occurs with probability 0.5
  - Each term containing one x_i variable and d-1 v_j variables occurs with probability 0.5
- Polynomials representing cryptographic schemes are typically sufficiently random.

Source: Adi Shamir.

Polynomials, P, which are nonrandom, or of unknown degree

- Choose an arbitrary degree d and a random term t which is the product of d-1 v_j variables.
- Find multiple numeric values of the superpoly of t by computing several random subcube sums.
- If the result is always the same value, d is too high: eliminate one v_j from t, and repeat.
- If the result is a nonlinear function, d is too low: add a random v_j to the term t, and repeat.
- Advantage: partial sums can be reused.

Source: Adi Shamir.

End

Extra

- $P_{n,d} \subset k[x_1, \ldots, x_n]$ is the vector subspace of polynomials of degree $\leq d$. $\dim(P_{n,d}) = {}_{n+d}C_n$.
- Given N points, for what d, does the ideal, $I_d \subset P_{n,d}$, such that I vanishes on all N points.
- $C_d(S) = \dim(P_{n,d}) - \dim(I_d)$.
- If □ is an invertible affine transformation, then $C_d(S) = C_d(□(S))$.
- Rational maps induce k-algebra homomorphisms between function fields.
- Res(f,g) is irreducible.
- If $I = \langle f_v, g_w \rangle$, then $I \cap k[v, w] = \mathrm{Res}(f_v, g_w)$.
- Pullback: □* is a pullback of f if □*f= f□
- $\deg(F_i) = m_i$. $F_1 = F_2 = F_3 = 0$ have a common solution.
- $□_0(d)$: $S_{d-m1} \oplus S_{d-m2} \oplus S_{d-m3} \rightarrow S_d$ by $(A_1, A_2, A_3) \rightarrow (A_1F_1 + A_2F_2 + A_3F_3)$.
- Over 2 affine variable we get $\dim(S_d) = {}_{d+1}C_d = (d+2)(d+1)/2$.

BES

- $AES_k(P)=C \leftarrow BES_{\square(k)}(\square(P))= \square(C)$.
- Let $\square(i)= 2^i$, $\square(a)= (a^{\square(0)}, a^{\square(1)}, a^{\square(2)}, a^{\square(3)}, a^{\square(4)}, a^{\square(5)}, a^{\square(6)}, a^{\square(7)})$.
- BES: $b \rightarrow M_B\, b^{-1} + k_B$.
  - $w_0=p+k_0$.
  - $x_i=w_i^{-1}$, $w_i=M_B\, x_{i-1} + k_i$.
  - $c=M_B * x_9 + k_{10}$.
- Circulant as linearized polynomial: $x \rightarrow$ 0x05$x^{\square(0)}$+ 0x09$x^{\square(1)}$+ 0xf9$x^{\square(2)}$+ 0x25$x^{\square(3)}$+ 0xf4$x^{\square(4)}$+ 0x01$x^{\square(5)}$+ 0xb5$x^{\square(6)}$+ 0x8f$x^{\square(7)}$.
- S: $w \rightarrow \square_{i=0}^{7} \square_i\, w^{255-\square(i)}$ +0x63, modified: S: $w \rightarrow \square_{i=0}^{7} \square_i\, w^{-\square(i)}$.

AES algebraic expansion

- For each round, $(0 \le i \le 9)$ and each S-box $(0 \le j \le 15)$, we get r= 8 x 3 =24 quadratics.
- S: Total S-boxes. B: S-boxes/round.
- P-1: passive S-Boxes, Highest degree: 2P. R: Equations.
- L_k: independent key variables, S_k: key variables.
- $|R| = {}_S C_P \, (t^P - (t-r)^P)$, $|R'| = {}_S C_{P-1} \, SB \, (N_r + 1) \, (t-r)^{P-1}$.
- $|R''| = {}_S C_{P-1} \, (S_k - L_k) \, (N_r + 1) \, (t-r)^{P-1}$.
- Total terms: $T = {}_S C_P \, t^P$.
- For P=2, (R+R'+R'')= 33,665,888, T= 33,788,100.
- For P=3, (R+R'+R'')= 95.18×10^9, T= $91.9 \times x \times 10^9$.