AIMer: ZKP-based Digital Signature

김성광[3] 하진철[1] 손민철[1] 이병학[1] 문덕재[3] 이주희[2]
이상엽[3] 권지훈[3] 조지훈[3] 윤효진[3] 이주영[1]

[1]KAIST [2]성신여대 [3]삼성 SDS

2023. 02. 24.

Table of Contents

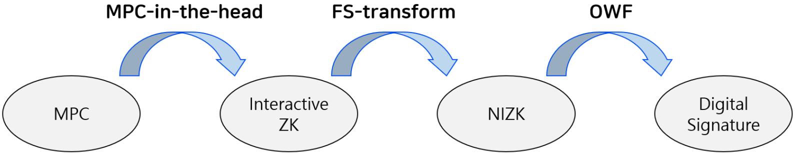
ZKP-based Digital Signature

- ZKP-based digital signature is based on a zero-knowledge proof of knowledge of a solution to a certain hard problem
  - For example, finding a preimage of a one-way function
- Efficiency of the ZKP-based signature is determined by choice of **one-way function** and **zero-knowledge proof system**
- Characteristics of the ZKP-based digital signature is:
  - ✓ Minimal assumption : Security of ZKP-based digital signature only relies on the one-wayness of one-way function
  - ✓ Trade-off between time & size
  - ✓ Small public key and secret key
  - ✓ Relatively large signature size and sign/verify time

AIMer Signature

- In AIMer digital signature, AIM one-way function and BN++ proof system is used
- Compare to the other ZKP-based digital signature, AIMer has two advantages:
  - ✓ Fully exploit repeated multiplier technique to reduce a signature size
  - ✓ More secure against algebraic attacks

ZKP from MPC-in-the-Head



MPC-in-the-Head

Variable	Share					Value
	Party 1	Party 2	Party 3	Party 4	Party 5	
x	5	6	1	3	9	2
y	10	0	6	7	5	6
z	9	4	1	2	7	1

Example of MPC-in-the-head setting for $N = 5$ parties over $\mathbb{F}_{11}$

- MPC-in-the-head is a Zero-Knowledge protocol by running the MPC protocol *in prover's head*
- In the multiparty computation setting, $x^{(i)}$ denotes the i-th party's additive share of x, $\sum_i x^{(i)} = x$
- N parties have a shares of x, y, and z which satisfies $xy = z$. They wants to prove that $xy = z$ without reveal the value
- N parties and verifier run 5 rounds interactive protocol

MPC-in-the-Head - Toy Example

Phase	Variable	Share					Value
		Party 1	Party 2	Party 3	Party 4	Party 5	
Phase 1	x	5	6	1	3	9	2
	y	10	0	6	7	5	6
	z	9	4	1	2	7	1
	a	7	2	6	2	3	9
	b	6	4	3	0	1	3
	c	4	6	3	7	7	5
	com	$h(5,10,9,7,6,4)$	$h(6,0,4,2,4,6)$	$h(1,6,1,6,3,3)$	$h(3,7,2,2,0,7)$	$h(9,5,7,3,1,7)$	-

Gray values are hidden to the verifier

Phase 1

- N parties generate the shares of the another multiplication triples (a, b, c) which satisfies $ab = c$
- Each party commits[1] to their own shares and open it

[1] Commit means that keeping the value hidden to others, with the ability to reveal the committed value later

MPC-in-the-Head - Toy Example

Phase	Variable	Share					Value
		Party 1	Party 2	Party 3	Party 4	Party 5	
Phase 1	x	5	6	1	3	9	2
	y	10	0	6	7	5	6
	z	9	4	1	2	7	1
	a	7	2	6	2	3	9
	b	6	4	3	0	1	3
	c	4	6	3	7	7	5
	com	$h(5, 10, 9, 7, 6, 4)$	$h(6, 0, 4, 2, 4, 6)$	$h(1, 6, 1, 6, 3, 3)$	$h(3, 7, 2, 2, 0, 7)$	$h(9, 5, 7, 3, 1, 7)$	-
Phase 2	Random challenge $r = 5$ from the verifier						

Phase 2

- Verifier sends random challenge r to parties

MPC-in-the-Head - Toy Example

Phase	Variable	Share					Value
		Party 1	Party 2	Party 3	Party 4	Party 5	
Phase 1	x	5	6	1	3	9	2
	y	10	0	6	7	5	6
	z	9	4	1	2	7	1
	a	7	2	6	2	3	9
	b	6	4	3	0	1	3
	c	4	6	3	7	7	5
	com	$h(5,10,9,7,6,4)$	$h(6,0,4,2,4,6)$	$h(1,6,1,6,3,3)$	$h(3,7,2,2,0,7)$	$h(9,5,7,3,1,7)$	-
Phase 2	Random challenge $r=5$ from the verifier						
Phase 3	α	10	10	0	6	4	8
	β	5	4	9	7	6	9
	v	3	9	3	10	8	0

Phase 3

- The parties locally set $\alpha^{(i)} = r \cdot x^{(i)} + a^{(i)}, \beta^{(i)} = y^{(i)} + b^{(i)}$ and broadcast them
- The parties locally set

$$v^{(i)} = \begin{cases} r \cdot z^{(i)} - c^{(i)} + \alpha \cdot b^{(i)} + \beta \cdot a^{(i)} - \alpha \cdot \beta & \text{if } i = 1 \\ r \cdot z^{(i)} - c^{(i)} + \alpha \cdot b^{(i)} + \beta \cdot a^{(i)} & \text{otherwise} \end{cases}$$

MPC-in-the-Head - Toy Example

Phase	Variable	Share					Value
		Party 1	Party 2	Party 3	Party 4	Party 5	
Phase 1	x	5	6	1	3	9	2
	y	10	0	6	7	5	6
	z	9	4	1	2	7	1
	a	7	2	6	2	3	9
	b	6	4	3	0	1	3
	c	4	6	3	7	7	5
	com	$h(5,10,9,7,6,4)$	$h(6,0,4,2,4,6)$	$h(1,6,1,6,3,3)$	$h(3,7,2,2,0,7)$	$h(9,5,7,3,1,7)$	-
Phase 2	Random challenge $r=5$ from the verifier						
Phase 3	α	10	10	0	6	4	8
	β	5	4	9	7	6	9
	v	3	9	3	10	8	0

Phase 3 (Cont')

- Each party opens $v^{(i)}$ to compute v
- If $ab = c$ and $xy = z$, then $v = 0$

MPC-in-the-Head - Toy Example

Phase	Variable	Share: Party 1	Party 2	Party 3	Party 4	Party 5	Value
Phase 1	x	5	6	1	3	9	2
	y	10	0	6	7	5	6
	z	9	4	1	2	7	1
	a	7	2	6	2	3	9
	b	6	4	3	0	1	3
	c	4	6	3	7	7	5
	com	$h(5,10,9,7,6,4)$	$h(6,0,4,2,4,6)$	$h(1,6,1,6,3,3)$	$h(3,7,2,2,0,7)$	$h(9,5,7,3,1,7)$	-
Phase 2	Random challenge $r = 5$ from the verifier						
Phase 3	α	10	10	0	6	4	8
	β	5	4	9	7	6	9
	v	3	9	3	10	8	0
Phase 4	Random challenge $\bar{i} = 4$ from the verifier						

Phase 4

- Verifier sends a hidden party index $\bar{i}$ to parties

MPC-in-the-Head - Toy Example

Phase	Variable	Share: Party 1	Party 2	Party 3	Party 4	Party 5	Value
Phase 1	x	5	6	1	3	9	2
	y	10	0	6	7	5	6
	z	9	4	1	2	7	1
	a	7	2	6	2	3	9
	b	6	4	3	0	1	3
	c	4	6	3	7	7	5
	com	$h(5,10,9,7,6,4)$	$h(6,0,4,2,4,6)$	$h(1,6,1,6,3,3)$	$h(3,7,2,2,0,7)$	$h(9,5,7,3,1,7)$	-
Phase 2	Random challenge $r = 5$ from the verifier						
Phase 3	α	10	10	0	6	4	8
	β	5	4	9	7	6	9
	v	3	9	3	10	8	0
Phase 4	Random challenge $\bar{i} = 4$ from the verifier						
Phase 5	Open all parties except $\bar{i}$-th party and check consistency						

Phase 5

- Each party $i \in [N]\backslash\{\bar{i}\}$ sends $x^{(i)}, y^{(i)}, z^{(i)}, a^{(i)}, b^{(i)}$, and $c^{(i)}$ to verifier
- Verifier checks the consistency of the received shares

MPC-in-the-Head

- Some agreed-upon circuit $C : \mathbb{F}^n \to \mathbb{F}^m$ and some output $\mathbf{y}$, prover wants to prove knowledge of input $\mathbf{x} = (x_1, \ldots, x_n)$ such that $C(\mathbf{x}) = \mathbf{y}$ **without revealing** $\mathbf{x}$
- The single prover simulates N parties in prover's head. Prover first divides the input $x_1, \ldots, x_n$ into shares $x_1^{(i)}, \ldots, x_n^{(i)}$
- For each addition $c = a + b$, $c^{(i)} = a^{(i)} + b^{(i)}$
- For each multiplication $c = ab$, prover divides c into shares $c^{(i)} = c$ then run multiplication check protocol

MPC-in-the-Head - Toy Example

$$C(x_1, x_2, x_3) = (x_1 + x_2 \cdot x_3) \cdot x_2 = 10$$

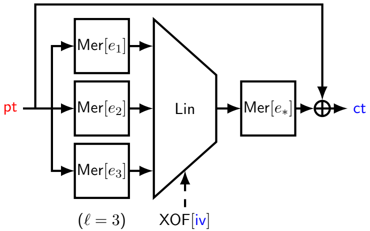
Variable	Share					Value
	Party 1	Party 2	Party 3	Party 4	Party 5	
x_1	7	2	1	3	0	2
x_2	3	5	10	5	5	6
x_3	9	5	9	3	10	3
$x_2 \cdot x_3$	2	4	3	5	4	7
$x_1 + x_2 \cdot x_3$	9	6	4	8	4	9
$(x_1 + x_2 \cdot x_3) \cdot x_2$	8	3	0	4	6	10

- Addition is almost *free*, so that efficiency is highly depend on the number of the multiplications
- Soundness error is proportional to $1/N$ and $1/|\mathbb{F}|$

Fiat-Shamir Transform

- Prover derives r and $\bar{i}$ from hash of the data of previous round without interaction. This technique is called Fiat-Shamir Transform
- Using Fiat-Shamir transform, interactive proof can be transformed into non-interactive proof
- Non-interactive zero-knowledge proof of knowledge of x which satisfies $f(x) = y$ for some one-way function f and output y is a digital signature
  - Public key: output y
  - Private key: input x

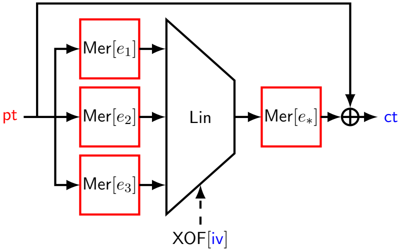
AIM - Specification



Scheme	λ	n	ℓ	e_1	e_2	e_3	e_*
AIM-I	128	128	2	3	27	-	5
AIM-III	192	192	2	5	29	-	7
AIM-V	256	256	3	3	53	7	5

- $\text{Mer}[e](x) = x^{2^e-1}$: Mersenne power function in $\mathbb{F}_{2^n}$
  - e is chosen such that $\text{Mer}[e]$ becomes a permutation
  - e_1, e_3, e_*: small values to provide smaller differential probability
  - e_2: large value to obtain full degree over $\mathbb{F}_2$ $(e_2 \cdot e_* > n)$
- $\text{Lin}(x) = Ax + b$: Multiplication by a random binary matrix A and addition by a random constant b in $\mathbb{F}_2$

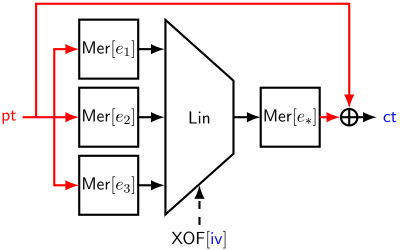
AIM - Design Rationale



Mersenne S-box

- Mer$[e](x) = x^{2^e-1}$
- Only one multiplication is required for its proof $(xy = x^{2^e})$
- More secure than Inv S-box against algebraic attacks on $\mathbb{F}_2$
- Providing moderate DC/LC resistance

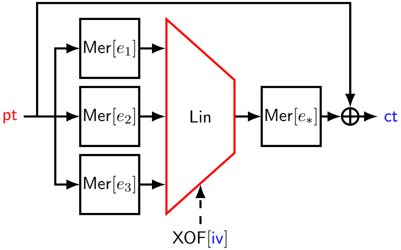
AIM - Design Rationale



Repetitive Structure

- In ZKP-based digital signature, efficiency is highly depend on the number of the multiplications
- In BN++ proof system, when multiplication triples use an identical multiplier in common, the proof can be done in a batched way, reducing the signature size
- AIM allows us to take full advantage of this technique

AIM - Design Rationale



Random Affine Layer

- Random affine layer incereases the algebraic degree of equations over $\mathbb{F}_{2^n}$
- In order to mitigate multi-target attacks, the affine map is uniquely generated for each user's iv

AIMer - Performance

Type	Scheme	$\lvert pk \rvert$ (B)	$\lvert sig \rvert$ (B)	Sign (ms)	Verify (ms)
Lattice-based	Dilithium2	1312	2420	0.10	0.03
	Falcon-512	897	690	0.27	0.04
Hash-based	SPHINCS$^+$-128s*	32	7856	315.74	0.35
	SPHINCS$^+$-128f*	32	17088	16.32	0.97
ZKP-based	Picnic3-L1	32	12463	5.83	4.24
	Banquet	32	19776	7.09	5.24
	Rainier$_3$	32	8544	0.97	0.89
	Rainier$_4$	32	9600	1.15	1.05
	BN++Rain$_3$	32	6432	0.83	0.77
	BN++Rain$_4$	32	7488	0.93	0.86
	AIMer-I	32	5904	0.82	0.78

*: -SHAKE-simple

- Experiments are measured in Intel Xeon E5-1650 v3 @ 3.50GHz with 128 GB memory, AVX2 enabled
- Among the ZKP-based and hash-based digital signatures, AIMer is the most efficient one

Algebraic Attacks

- Basically, an algebraic attack is to model a symmetric key primitive as a system of (multivariate) polynomial equations and to solve it using algebraic technique.
- In this work, we mainly consider the following two attacks since they are possible using only a single evaluation data.
    - The Gröbner basis attack
    - The eXtended Linearization attack
- The condition giving only one evaluation data considers the ZKP-based digital signature based on symmetric key primitives.

Gröbner Basis Attack[2]

Definition (informal)

Given a field $\mathbb{F}$ and its polynomial ring $\mathbb{F}[\mathbf{x}]$, a Gröbner basis G for a system $I \subseteq \mathbb{F}[\mathbf{x}]$ is a set of polynomials such that

- for all $f \in \mathbb{F}[\mathbf{x}]$ the remainder of f divided by G is unique, and
- for all $f \in I$ the remainder of f divided by G is 0.

(Counter-example) Consider $\mathbb{R}[x, y, z]$ with lexicographic order. For $G = \{x^2y - 2yz, y^2 - z^2, xz^2\}$ and $f = x^2y^2 + y^2z^2 - 2y^2z$,

- $f = y \cdot (x^2y - 2yz) + z^2 \cdot (y^2 - z^2) + 0 \cdot xz^2 + z^4$
- $f = (x^2 + z^2 - 2z) \cdot (y^2 - z^2) + x \cdot xz^2 + 0 \cdot (x^2y - 2yz) + (z^4 - 2z^3)$

[2] Examples in this presentation are from J. F. Sauer and A. Szepieniec. *SoK: Gröbner Basis Algorithms for Arithmetization Oriented Ciphers*.

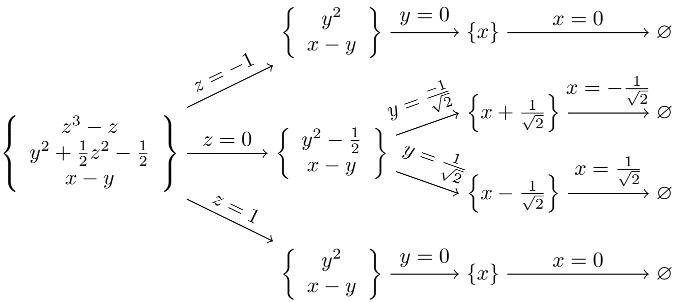
Gröbner Basis Attack (Example)

In $\mathbb{R}[x, y, z]$, a system

$$\{x - y, xyz, x^2 + y^2 + z^2 - 1\}$$

has a Gröbner basis in lex order as follows.

$$\{x - y, y^2 - 0.5z^2 - 0.5, z^3 - z\}.$$



Gröbner Basis Attack

- The Gröbner basis attack: solve a system by computing its Gröbner basis
  1. Compute a Gröbner basis in the grevlex[3] order
  2. Change the order of terms to obtain a Gröbner basis in the lex[4] order
  3. Find a univariate polynomial in this basis and solve it
  4. Substitute the solution into the basis and repeat Step 3
- Existence of a univariate polynomial in Step 3 is guaranteed the system has only finitely many solutions in the algebraic closure of the domain.
  - This is the reason we need to add field equations of the form $x^q = x$ for all variables in the system over $\mathbb{F}_q$.
- The attack complexity is usually lower bounded by Step 1, computing a Gröbner basis (in the grevlex order).

[3] graded reverse lexicographic
[4] lexicographic

The eXtended Linearization (XL)

- Trivial Linearization:
  1. Replace every monomial of degrees greater than 1 with a new variable to make the system linear
  2. Solve the linearized system using linear algebra techniques
  3. Check whether the solution satisfies the substitution in Step 1

  - The number of equations should be greater than or equal to the number of monomials appearing in the system.
  - It is hard to satisfy the above condition when only a single evaluation data is given.
- The XL attack (for Boolean quadratic system):
  - Multiplying all monomials of degrees at most $D-2$ for some $D>2$
  - For large enough D, the extended system has more equations than the number of appearing monomials.
  - Apply trivial linearization to the extended system.

XL Attack (Example)

Consider the following system of equations over $\mathbb{F}_2$:

$$\begin{cases} f_1(x,y,z) = xy + x + yz + z = 0 \\ f_2(x,y,z) = xz + x + y + 1 = 0 \\ f_3(x,y,z) = xz + yz + y + z = 0 \end{cases}$$

- Trivial linearization does not work since there are 6 monomials and 3 equations.
- Choose $D = 3$ and apply the XL attack.

XL Attack (Example)

$$
\begin{cases}
xf_1 : xyz + xy + xz + x = 0 \\
yf_1 : 0 = 0 \\
zf_1 : xyz + xz + yz + z = 0 \\
\;\; f_1 : xy + x + yz + z = 0 \\
xf_2 : xz + xy = 0 \\
yf_2 : xyz + xy = 0 \\
zf_2 : yz + z = 0 \\
\;\; f_2 : xz + x + y + 1 = 0 \\
xf_3 : xyz + xy = 0 \\
yf_3 : xyz + y = 0 \\
zf_3 : xz + z = 0 \\
\;\; f_3 : xz + yz + y + z = 0
\end{cases}
$$

$$
\begin{bmatrix}
1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0\\
0&1&0&1&1&0&1&0\\
0&1&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0\\
0&0&0&0&1&0&1&0\\
0&0&1&1&0&1&0&1\\
1&1&0&0&0&0&0&0\\
1&0&0&0&0&1&0&0\\
0&0&1&0&0&0&1&0\\
0&0&1&0&1&1&1&0
\end{bmatrix}
\begin{bmatrix} xyz \\ xy \\ xz \\ x \\ yz \\ y \\ z \\ 1 \end{bmatrix} = 0
\qquad
\begin{bmatrix}
1&1&1&1&0&0&0&0\\
0&1&0&1&1&0&1&0\\
0&0&1&1&1&0&1&0\\
0&0&0&1&0&1&0&0\\
0&0&0&0&1&0&1&0\\
0&0&0&0&0&1&0&1\\
0&0&0&0&0&0&1&1\\
0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0
\end{bmatrix}
\begin{bmatrix} xyz \\ xy \\ xz \\ x \\ yz \\ y \\ z \\ 1 \end{bmatrix} = 0
$$

1. Extended system of equations
2. Macaulay matrix for the extended system
3. Performing Gaussian elimination

The Number of Quadratic Equations

To apply algebraic attacks, one has to represent a symmetric primitive as a system of equations.

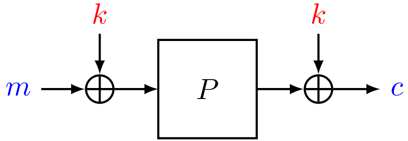
- Each Mersenne S-box in AIM can be represented as a system of Boolean quadratic equations (w.r.t. its input/output).
  - For example, there are n quadratic equations directly obtained from $xy = x^{2^e}$ for $x, y \in \mathbb{F}_{2^n}$.
  - In fact, we choose the parameter e for the Mersenne S-boxes in AIM such that Mer$[e]$ has $3n$ quadratic equations.
  - Compared to the inverse S-box having $5n$ quadratic equations, our Mersenne S-boxes have smaller numbers of quadratic equations.
- The exact number of quadratic equations induced from S-box is a critical factor to algebraic attacks.

Experiment on an Even-Mansour Cipher

Consider an Even-Mansour cipher defined as

$$E_k(m) = P(m+k) + k = c$$

where the permutation P is defined as $P = R \circ S \circ L$ for random affine mappings L and R, and an S-box S given as $S(x) = x^a$.



- Goal: given a pair of (m, c), find corresponding key k
- Suppose S has νn Boolean quadratic equations. How the value of ν affects the cost of algebraic attacks to recover k?

Experiment on Some S-boxes

S-box	Condition on the size n	Exponent	Implicit Boolean Quadratic Relation	ν
Inverse	$n > 4$	$2^n - 2$	$xy = 1^\dagger$	$5^\dagger$
Mersenne	$\gcd(n, e) = 1$	$2^e - 1$	$xy = x^{2^e}$	$3^{\dagger\dagger}$
NGG	$n = 2s \geq 8$	$2^{s+1} + 2^{s-1} - 1$	$xy = x^{2^{s+1}+2^{s-1}}$	2

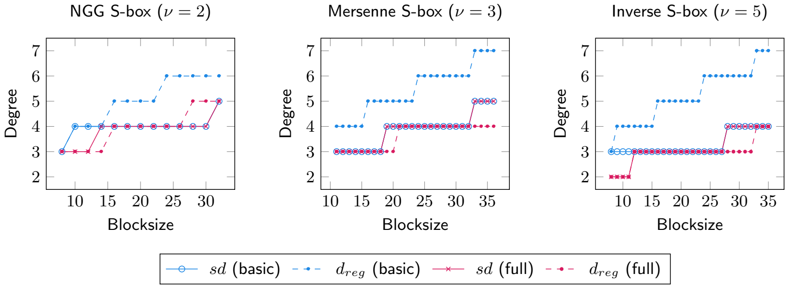
† Assuming x, y are nonzero.

†† This is not for all e, but we can choose such e.

We perform an experiment computing a Gröbner basis for two kinds of systems representing the Even-Mansour ciphers with the above S-boxes.

1. Basic system
   - n quadratic equations that directly comes from the implicit Boolean quadratic relation
   - n field equations of degrees 2 for computing Gröbner basis
2. Full system
   - all possible νn linearly independent quadratic equations induced from the S-box
   - n field equations of degrees 2 for computing Gröbner basis

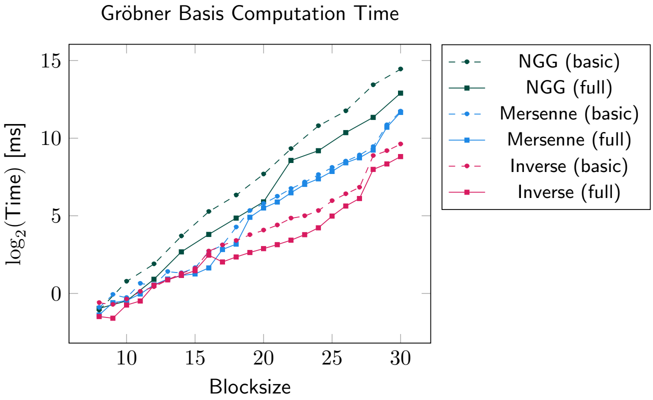
Experiment Result: Gröbner Basis Attack



The cost of computing Gröbner basis is usually represented by the highest degree reached during the computation.

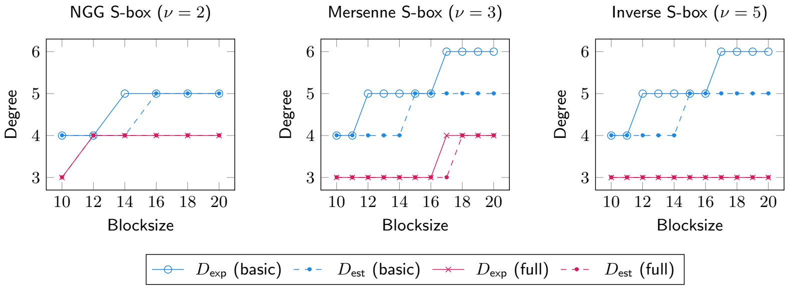
- sd: result from the experiment
- d_{reg}: theoretic estimation

Experiment Result: Gröbner Basis Attack



- Environment: AMD Ryzen 7 2700X 3.70GHz with 128 GB memory

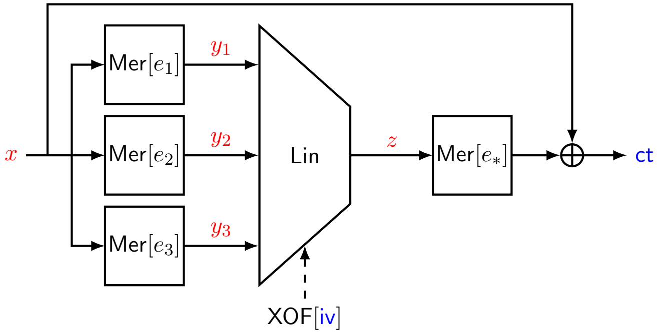
Experiment Result: XL Attack



The cost of XL attack is determined by the target degree D.

- D_{exp}: result from the experiment
- D_{est}: theoretic estimation

Systems for AIM-V



- $y_i = \text{Mer}[e_i](x) \iff x = \text{Mer}[e_i]^{-1}(y_i) \iff xy = x^{2^e}$
- $x \oplus \text{ct} = \text{Mer}[e_*](z) \iff z = \text{Mer}[e_*]^{-1}(x \oplus \text{ct}) \iff z(x \oplus \text{ct}) = z^{2^e}$
- $y_i = \text{Mer}[e_i] \circ \text{Mer}[e_j]^{-1}(y_j) = \text{Mer}[e_i]\left(\text{Mer}[e_*](z) \oplus \text{ct}\right)$

Algebraic Analysis on AIM

Scheme	#Var	Variables	Gröbner Basis		XL	
			d_{reg}	Time	D	Time
AIM-I	n	z	51	300.8	52	244.8
	$2n$	x, y_2	22	**214.9**	14	150.4
	$3n$	x, y_1, y_2	20	222.8	12	**148.0**
AIM-III	n	z	82	474.0	84	375.3
	$2n$	x, y_2	31	**310.6**	18	203.0
	$3n$	x, y_1, y_2	27	310.8	15	**194.1**
AIM-V	n	z	100	601.1	101	489.7
	$2n$	x, y_2	40	**406.2**	26	289.5
	$3n$	x, y_2, y_3	47	510.4	20	**260.6**
	$4n$	x, y_1, y_2, y_3	45	530.3	19	266.1

Thank you for listening!

Appendix

Algebraic Degree

Suppose $f : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ is defined as $f(x) = x^a$ for some $1 \leq a < 2^n$. Then the algebraic degree of f is $\mathrm{hw}(a)$.

Suppose $\mathbb{F}_{2^n}$ is constructed as $\mathbb{F}_2(\alpha)$ where α is a root of an irreducible polynomial of degree n.

- $x \in \mathbb{F}_{2^n}$ can be represented as

$$x = x_0 + x_1\alpha + x_2\alpha^2 + \cdots + x_{n-1}\alpha^{n-1}$$

  for some $x_0, x_1, \ldots, x_{n-1} \in \mathbb{F}_2$.
- $x^2 = x_0 + x_1\alpha^2 + x_2\alpha^4 + \cdots + x_{n-1}\alpha^{2(n-1)}$
- Each coefficient of x^a is a monomial of degree $\mathrm{hw}(a)$ with respect to $x_0, x_1, \ldots, x_{n-1}$.

Monomial Orders

A monomial order $\prec$ is a total order on the set of monomials $\mathcal{M}$;

1. $\forall m \in \mathcal{M}$, $\mathbf{x}^{\mathbf{a}} \prec \mathbf{x}^{\mathbf{b}} \iff m\mathbf{x}^{\mathbf{a}} \prec m\mathbf{x}^{\mathbf{b}}$
2. The monomial $1 = \mathbf{x}^{(0,0,\ldots,0)}$ is the smallest one

- lex (lexicographical) order
  - $\mathbf{x}^{\mathbf{a}} \prec_{\text{lex}} \mathbf{x}^{\mathbf{b}}$ iff the first nonzero entry of $\mathbf{a} - \mathbf{b}$ is negative
  - In $\mathbb{F}[x, y, z]$ with lex order,

$$xy^2 \prec xy^2z \prec x^2z^2 \prec x^2yz \prec x^3$$

- grevlex (graded reverse lexicographical) order
  - $\mathbf{x}^{\mathbf{a}} \prec_{\text{grevlex}} \mathbf{x}^{\mathbf{b}}$ iff either $\sum_i a_i < \sum_i b_i$ or $\sum_i a_i = \sum_i b_i$ and $\mathbf{x}^{\mathbf{a}} \succ_{\text{invlex}} \mathbf{x}^{\mathbf{b}}$, where invlex is a lex order with inversely labeled variables.
  - In $\mathbb{F}[x, y, z]$ with grevlex order,

$$xy^2 \prec x^3 \prec xy^2z \prec x^2z^2 \prec x^2yz$$

Gröbner Basis Attack

- The complexity of computing Gröbner basis is estimated using *the degree of regularity* of the system.
- It basically estimates the highest degree reached during the Gröbner basis computation.
- For the degree d_{reg} of regularity, the complexity computing a Gröbner basis is given by

$$O\left(\binom{n_{var}+d_{reg}}{d_{reg}}^{\omega}\right)$$

where n_{var} is the number of variables in the system and $2 \leq \omega \leq 3$ is the linear algebra constant.

Gröbner Basis Attack

- d_{reg} for an over-defined system is computed as follows.
  - Consider a system $\{f_i\}_{i=1}^m$ of m equations in n variables where $m > n$ and $d_i = \deg f_i$.
  - Then d_{reg} is the smallest of the degrees of the terms with non-positive coefficients for the following Hilbert series under the semi-regularity assumption.

$$\mathrm{HS}(z) = \frac{1}{(1-z)^n} \prod_{i=1}^{m} (1 - z^{d_i}).$$

- For an application to a symmetric key primitive,
  - The system modeling the primitive is always over-defined due to the field equation of the form $x^{p^e} - x = 0$ over $\mathbb{F}_{p^e}$.
  - In most cases, compute d_{reg} assuming the semi-regularity.

Example

Consider an Even-Mansour cipher defined as

$$E_k(m) = P(m+k) + k = c$$

where the permutation P is defined as $P = R \circ S \circ L$ for random affine mappings L and R, and an S-box S given as $S(x) = x^a$.

- Goal: given a pair of (m, c), find corresponding key k
  1. Build a system over $\mathbb{F}_{2^n}$ in one variable k:
     - This kind of system is mainly considered in recent papers.
  2. Build a system over $\mathbb{F}_2$ in n variables representing bits of k:
     - νn implicit quadratic equations for some $\nu > 0$, and n field equations of degree 2
     - $\mathrm{HS}(z) = \frac{1}{(1-z)^n}(1-z^2)^{\nu n}(1-z^2)^n = (1+z)^n(1-z^2)^{\nu n}$

Example

$$\mathrm{HS}(z) = (1+z)^n (1-z^2)^{\nu n}$$

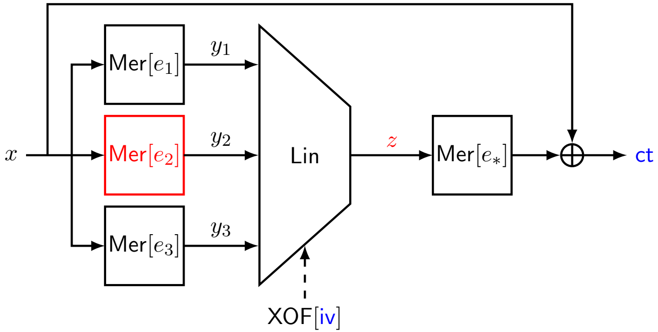
n	ν	d_{reg}	Time [bits]
8	1	3	14.73
	2	3	14.73
	3	3	14.73
	4	2	10.98
	5	2	10.98
9	1	4	18.96
	2	3	15.56
	3	3	15.56
	4	2	11.56
	5	2	11.56
10	1	4	19.93
	2	3	16.32
	3	3	16.32
	4	3	16.32
	5	2	12.09

n	ν	d_{reg}	Time [bits]
128	1	17	144.63
	2	11	104.94
	3	9	90.05
	4	8	82.20
	5	7	74.02
192	1	23	203.99
	2	15	148.81
	3	12	125.52
	4	10	108.93
	5	9	100.26
256	1	29	263.12
	2	19	192.58
	3	14	152.48
	4	12	135.19
	5	10	117.03

XL Attack

- How large D should be to solve the given system?
  - There is no method to find such D without experimentally running the XL algorithm.
  - We can give a loose bound for D, assuming the extended equations during the XL algorithm are linearly independent.
- Given a system of m Boolean quadratic equations in n variables:
  - The XL algorithm with the target degree D multiplies $\sum_{i=1}^{D-2} \binom{n}{i}$ monomials, obtaining $m \cdot \sum_{i=1}^{D-2} \binom{n}{i}$ equations.
  - Let T_D be the number of monomials appearing in the extended system. When the extended system is dense, i.e., all monomials appear, we have $T_D = \sum_{i=1}^{D} \binom{n}{i}$.
  - The XL attack works when the number of linearly independent equations in the extended system is greater than or equal to T_D, and its complexity is given by $O(T_D^{\omega})$.

Systems for AIM-V: n variables



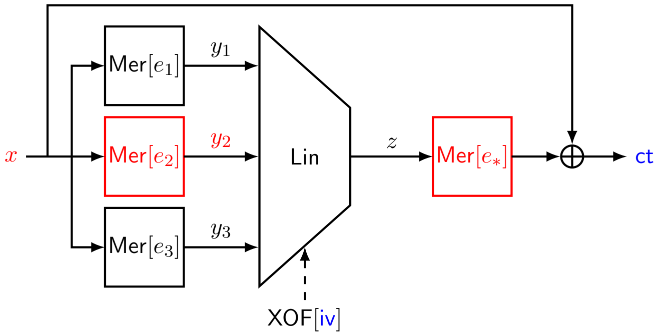
$$\left(\mathsf{Mer}[e_*](z) \oplus \mathsf{ct}\right)^{2^{e_2}} = \left(\mathsf{Mer}[e_*](z) \oplus \mathsf{ct}\right)$$
$$\times\ \mathsf{Lin}'\left(\mathsf{Mer}[e_1](\mathsf{Mer}[e_*](z) \oplus \mathsf{ct}), \mathsf{Mer}[e_3](\mathsf{Mer}[e_*](z) \oplus \mathsf{ct}), z\right)$$

where Lin' denotes a linear function such that $y_2 = \mathsf{Lin}'(y_1, y_3, z)$.

- $3n$ equations of degree

$$e_* + \max(\deg(\mathsf{Mer}[e_1] \circ \mathsf{Mer}[e_*]), \deg(\mathsf{Mer}[e_3] \circ \mathsf{Mer}[e_*]))$$

Systems for AIM-V: $2n$ variables

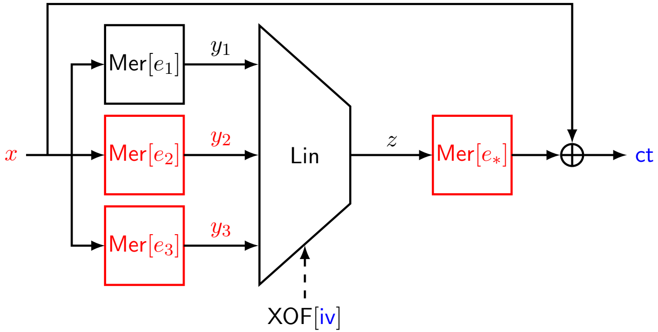


$$x \cdot y_2 = x^{2^{e_2}},$$

$$\mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, \mathsf{Mer}[e_3](x)) \cdot (x \oplus \mathsf{ct}) = \mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, \mathsf{Mer}[e_3](x))^{2^{e_*}}$$

- $3n$ quadratic equations
- $3n$ equations of degree $\max(e_1, e_3) + 1$

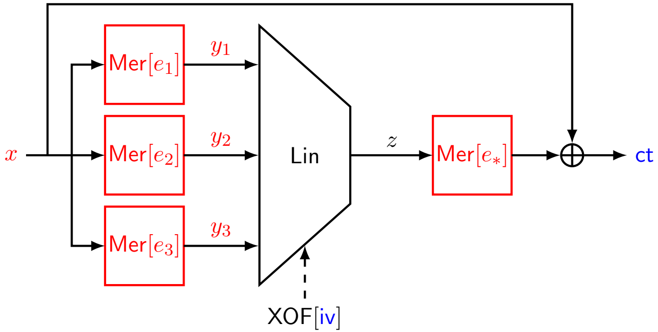
Systems for AIM-V: $3n$ variables



$$x \cdot y_2 = x^{2^{e_2}},$$
$$x \cdot y_3 = x^{2^{e_3}},$$
$$\mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, y_3) \cdot (x \oplus \mathsf{ct}) = \mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, y_3)^{2^{e_*}}$$

- $6n$ quadratic equations
- $3n$ equations of degree $e_1 + 1$

Systems for AIM-V: $4n$ variables



$$x \cdot y_1 = x^{2^{e_1}}, \qquad x \cdot y_2 = x^{2^{e_2}}, \qquad x \cdot y_3 = x^{2^{e_3}},$$
$$\mathsf{Lin}(y_1, y_2, y_3) \cdot (x \oplus \mathsf{ct}) = \mathsf{Lin}(y_1, y_2, y_3)^{2^{e_*}}$$

- $12n$ quadratic equations

Optimal Systems on AIM

Scheme	#Var	Variables	Gröbner Basis		XL	
			d_{reg}	Time	D	Time
AIM-I	n	z	51	300.8	52	244.8
	$2n$	x, y_2	22	**214.9**	14	150.4
	$3n$	x, y_1, y_2	20	222.8	12	**148.0**
AIM-III	n	z	82	474.0	84	375.3
	$2n$	x, y_2	31	**310.6**	18	203.0
	$3n$	x, y_1, y_2	27	310.8	15	**194.1**
AIM-V	n	z	100	601.1	101	489.7
	$2n$	x, y_2	40	**406.2**	26	289.5
	$3n$	x, y_2, y_3	47	510.4	20	**260.6**
	$4n$	x, y_1, y_2, y_3	45	530.3	19	266.1