AIMer

Seongkwang Kim[3] **Jincheol Ha**[1] Mincheol Son[1]
Byeonghak Lee[3] Dukjae Moon[3] Joohee Lee[2] Sangyub Lee[3]
Jihoon Kwon[3] Jihoon Cho[3] Hyojin Yoon[3] Jooyoung Lee[1]

[1]KAIST [2]Sungshin Women's University [3]Samsung SDS

2024. 02. 28.

Table of Contents

1. Introduction
2. Preliminaries
3. Recap on AIM
4. Change Log from KpqC Round 1
5. AIM2: Mitigation on AIM Cryptanalysis

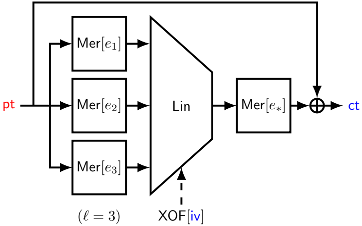
1. Introduction
2. Preliminaries
3. Recap on AIM
4. Change Log from KpqC Round 1
5. AIM2: Mitigation on AIM Cryptanalysis

MPCitH-based Digital Signature

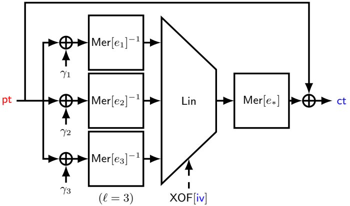
- ZKP-based digital signature is based on a zero-knowledge proof of knowledge of a solution to a certain hard problem
  - For example, finding a preimage of a one-way function
  - Efficiency of the ZKP-based signature is determined by choice of **one-way function** and **zero-knowledge proof system**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

AIMer Signature

- AIMer: MPCitH-based digital signature based on
  - (Ver.1.0) AIM and BN++ proof system
  - (Ver.2.0) AIM2 and customized BN++ proof system
- AIM (and AIM2): symmetric primitive based one-way function that fully exploits repeated multiplier technique to reduce a signature size



AIM



AIM2

1 Introduction

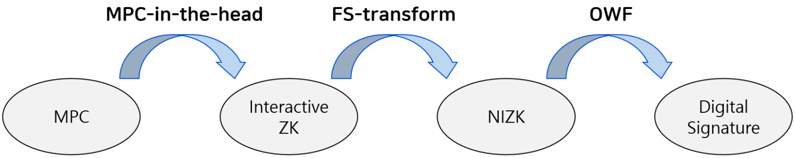
2 Preliminaries

3 Recap on AIM

4 Change Log from KpqC Round 1

5 AIM2: Mitigation on AIM Cryptanalysis

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					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 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					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 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

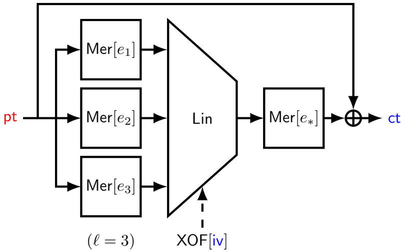
- Addition is almost *free*, so that efficiency is highly depend on the number of the multiplications
- Soundness error is proportional to $1/N$ and $1/|\mathbb{F}|$

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

1. Introduction
2. Preliminaries
3. Recap on AIM
4. Change Log from KpqC Round 1
5. AIM2: Mitigation on AIM Cryptanalysis

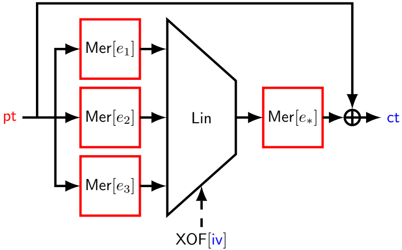
AIM - Specification



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

- Mersenne S-box: Mer$[e](x) = x^{2^e-1}$
- Randomized affine layer: Lin$(x) = Ax + b$
- Repetitive structure

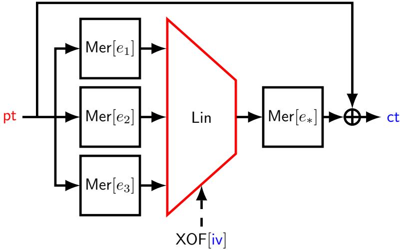
AIM - Design Rationale



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

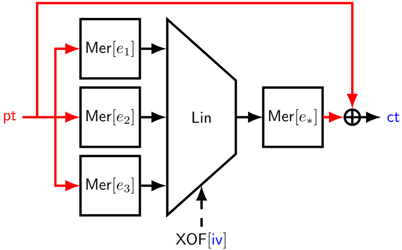
AIM - Design Rationale



Random Affine Layer

- Random affine layer increases 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

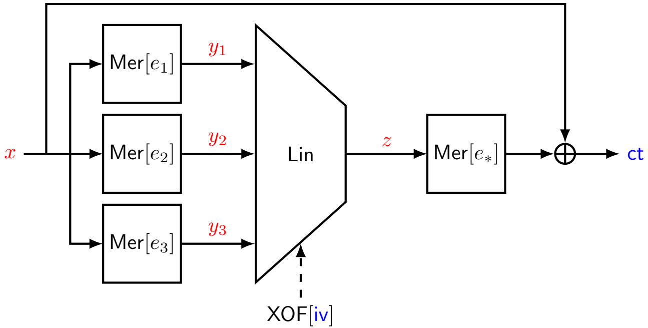
AIM - Design Rationale



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

Algebraic Analysis on AIM



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

Algebraic Analysis on AIM

Scheme	#Var	Variables	(# Eq, Deg)	Complexity
AIM-I	n	z	$(3n, 10)$	$2^{300.8}$
	$2n$	x, y_2	$(3n, 2) + (3n, 4)$	$2^{214.9}$
	$3n$	x, y_1, y_2	$(9n, 2)$	$2^{222.8}$
AIM-III	n	z	$(3n, 14)$	$2^{474.0}$
	$2n$	x, y_2	$(3n, 2) + (3n, 6)$	$2^{310.6}$
	$3n$	x, y_1, y_2	$(9n, 2)$	$2^{310.8}$
AIM-V	n	z	$(3n, 12)$	$2^{601.1}$
	$2n$	x, y_2	$(3n, 2) + (3n, 8)$	$2^{406.2}$
	$3n$	x, y_2, y_3	$(6n, 2) + (3n, 4)$	$2^{510.4}$
	$4n$	x, y_1, y_2, y_3	$(12n, 2)$	$2^{530.3}$

1 Introduction

2 Preliminaries

3 Recap on AIM

4 Change Log from KpqC Round 1

5 AIM2: Mitigation on AIM Cryptanalysis

Change of Specification

- We enhance the symmetric primitive AIM $\rightarrow$ AIM2 without performance degradation.
- The number of parameter sets are decreased from 4 to 2. The parameters are distinguished with name "-s" and "-f".
- Two hash functions with the same input is now integrated: Expand + Commit $\rightarrow$ CommitAndExpand.
- The salt size is now halved: $2\lambda \rightarrow \lambda$ bits.
- The message to be signed is now pre-hashed.
- Hash functions are now domain-separated.

Other Changes

Implementational Change

- We newly develop a reference code whose readability is significantly enhanced.
- There are now 4 types of source codes available: reference C, optimized C, AVX2, and ARM64.
- AVX2 optimization now enjoys a full parallelization of MPC simulations (30% sign time reduction).
- OpenSSL dependency is removed.
- Memory usage is reduced (195 KB $\rightarrow$ 150 KB for aimer128f).

Editorial Change

- The security proof (EUF-CMA) now guarantees full-bound security rather than birthday-bound security.
- Detailed specification which corresponds the reference code is now available.

1 Introduction

2 Preliminaries

3 Recap on AIM

4 Change Log from KpqC Round 1

5 AIM2: Mitigation on AIM Cryptanalysis

Recent Analysis on AIM

Recent algebraic analysis on AIM:

- Fukang Liu, et al. "Algebraic Attacks on RAIN and AIM Using Equivalent Representations", ToSC 2023.
- Private communication with Fukang Liu.
- Markku-Juhani O. Saarinen. "Round 1 (Additional Signatures) OFFICIAL_COMMENT: AIMER", pqc-forum[2].
- Kaiyi Zhang, et al. "Algebraic Attacks on Round-Reduced RAIN and Full AIM-III", ASIACRYPT 2023.

There are two vulnerabilities in the structure of AIM.

- Low degree equations in n variables.
- Structural vulnerability: common input to the parallel S-boxes.

[2] https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/BI2ilXblNy0

Low Degree Equations in n Variables

Fast exhaustive search by Fukang Liu. (ToSC 2023)

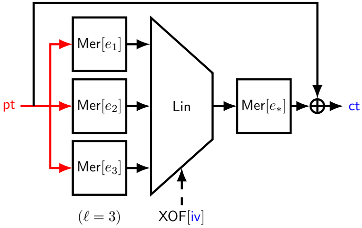
Scheme	Var	# Eq	Deg
AIM-I	z	$3n$	10
AIM-III	z	$3n$	14
AIM-V	z	$3n$	12

- Build low degree equations in n Boolean variables.
- Apply fast exhaustive search attack with memory-efficient Möbius transform.

Scheme	n	Brute-Force [bits]	Time [bits]	Memory [bits]
AIM-I	128	$2^{146.3}$	$2^{136.2}$ (-10.1)	$2^{61.7}$
AIM-III	192	$2^{211.8}$	$2^{200.7}$ (-11.1)	$2^{84.3}$
AIM-V	256	$2^{276.7}$	$2^{265.0}$ (-11.7)	$2^{95.1}$

Structural Vulnerability - System with New Variables

Private communication with Fukang Liu.

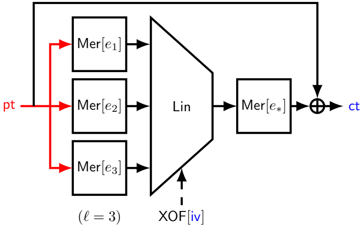


- $w :=$ pt$^{-1} \Rightarrow$ Mer$[e]$(pt) $=$ pt$^{2^e} w$
- $2n$-variable system having
  - $5n$ quadratic eqs from $w =$ pt^{-1}
  - $5n$ cubic eqs from Mer$[e_*]$

No practical attack exists on the above system, but it was not considered in the first proposal.

Structural Vulnerability - Efficient Brute-Force Search

NIST official comment on the additional signature by Saarinen.

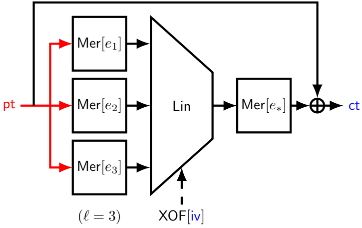


- $w :=$ pt$^{-1} \Rightarrow$ Mer$[e]$(pt) $=$ pt$^{2^e} w$
- Mer$[e_i]$(pt) can be computed by precomputing the linear matrix for E_i : pt $\mapsto$ pt$^{2^{e_i}}$.
- It might enable faster exhaustive search.

We analyzed the gate-complexity of AIM using this approach and verified that it is still larger than that of AES.

Structural Vulnerability - Linearization Attack

Linearization attack by Zhang et al. (ASIACRYPT 2023)

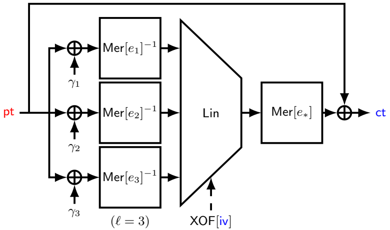


- Mer$[e_i]$(pt) $= (\text{pt}^d)^{s_i} \cdot \text{pt}^{2^{t_i}}$ for some $d \mid 2^n - 1$.
- Guessing pt^d can linearize the first round S-boxes.

Scheme	n	Brute-Force [bits]	d	Time [bits][3]	
AIM-I	128	$2^{146.3}$	5	$2^{146.0}$	(-0.3)
AIM-III	192	$2^{211.8}$	45	$2^{210.4}$	(-1.4)
AIM-V	256	$2^{276.7}$	3	$2^{277.0}$	

[3] It is re-analyzed complexity: https://eprint.iacr.org/2023/1474

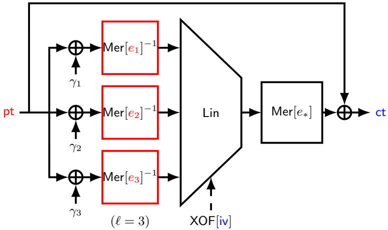
AIM2: Secure Patch for Algebraic Attacks



Scheme	λ	n	ℓ	e_1	e_2	e_3	e_*
AIM2-I	128	128	2	49	91	-	3
AIM2-III	192	192	2	17	47	-	5
AIM2-V	256	256	3	11	141	7	3

- Inverse Mersenne S-box
- Larger exponents
- Fixed constant addition

Inverse Mersenne S-box with Large Exponents

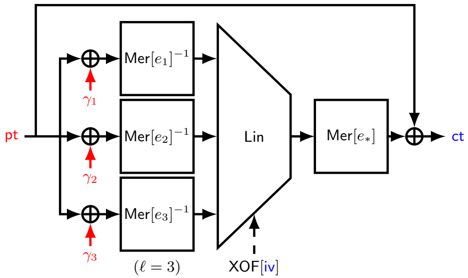


Scheme	λ	n	ℓ	e_1	e_2	e_3	e_*
AIM2-I	128	128	2	49	91	-	3
AIM2-III	192	192	2	17	47	-	5
AIM2-V	256	256	3	11	141	7	3
AIM-I	128	128	2	3	27	-	5
AIM-III	192	192	2	5	29	-	7
AIM-V	256	256	3	3	53	7	5

Inverse Mersenne S-box with large exponents

- $\text{Mer}[e]^{-1}(x) = x^a$ where $a = (2^e - 1)^{-1} \bmod (2^n - 1)$
- One multiplication for its proof ($\text{Mer}[e]^{-1}(x) = y \iff xy = y^{2^e}$)
- More resistance to algebraic attacks.
- Use larger e to mitigate the fast exhaustive search.

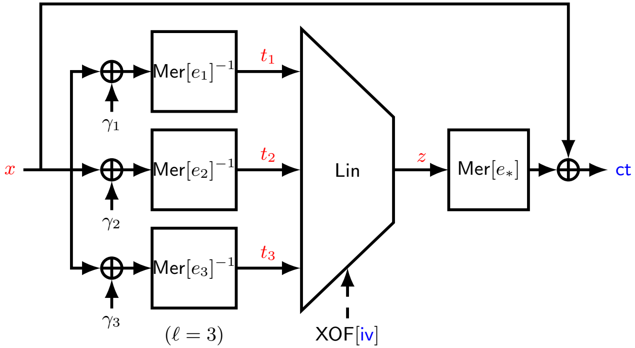
Constant Addition



Fixed Constant Addition

- Differentiate inputs of the S-boxes in the first round.
- Mitigate the structural vulnerability of AIM while maintaining the repetitive structure.

Algebraic Analysis on AIM2



- $t_i = \mathsf{Mer}[e_i]^{-1}(x \oplus \gamma_i) \iff x \oplus \gamma_i = \mathsf{Mer}[e_i](t_i) \iff (x \oplus \gamma_i)t_i = t_i^{2^{e_i}}$
- $x \oplus \mathsf{ct} = \mathsf{Mer}[e_*](z) \iff z = \mathsf{Mer}[e_*]^{-1}(x \oplus \mathsf{ct}) \iff (x \oplus \mathsf{ct})z = z^{2^{e_*}}$
- $t_i = \mathsf{Mer}[e_i]^{-1}\left(\mathsf{Mer}[e_j](t_j) \oplus \gamma_j \oplus \gamma_i\right)$

Algebraic Analysis on AIM2

Scheme	#Var	Variables	(# Eq, Deg)	Complexity
AIM2-I	n	t_1	$(n, 60)$	-
	$2n$	t_1, t_2	$(3n, 2)$	$2^{207.9}$
	$3n$	x, t_1, t_2	$(12n, 2)$	$2^{185.3}$
AIM2-III	n	x	$(2n, 114)$	-
	$2n$	t_1, t_2	$(3n, 2)$	$2^{301.9}$
	$3n$	x, t_1, t_2	$(12n, 2)$	$2^{262.4}$
AIM2-V	n	x	$(2n, 172)$	-
	$2n$	t_2, z	$(n, 2) + (2n, 38)$	$2^{513.5}$
	$3n$	t_1, t_2, t_3	$(6n, 2)$	$2^{503.7}$
	$4n$	x, t_1, t_2, t_3	$(18n, 2)$	$2^{411.4}$

AIMer ver.2.0 with AIM2

Scheme		Keygen (ms)	Sign (ms)	Verify (ms)	Size (B)
aimer128f	(ver.1.0)	0.02	0.60	0.53	5904
	(ver.2.0)	0.03	0.42	0.41	5888
aimer128s	(ver.1.0)	0.02	4.60	4.47	4176
	(ver.2.0)	0.03	3.18	3.13	4160
aimer192f	(ver.1.0)	0.03	1.39	1.28	13080
	(ver.2.0)	0.05	1.04	1.03	13056
aimer192s	(ver.1.0)	0.03	10.04	9.90	9144
	(ver.2.0)	0.05	7.94	7.86	9120
aimer256f	(ver.1.0)	0.08	2.50	2.34	25152
	(ver.2.0)	0.10	2.07	2.03	25120
aimer256s	(ver.1.0)	0.08	19.93	18.68	17088
	(ver.2.0)	0.10	15.26	14.81	17056

- Experiments are measured in Intel Xeon E5-1650 v3 @ 3.50GHz with 128 GB memory, AVX2 enabled

AIMer ver.2.0 with AIM2

Type	Scheme	$\|pk\|$ (B)	$\|sig\|$ (B)	Sign (ms)	Verify (ms)
Lattice-based	Dilithium2	1312	2420	0.10	0.03
	Falcon-512	897	690	0.27	0.04
	HAETAE-120†	992	1474	0.56	0.03
	NCC-Sign-cyclo (ref)†	1564	2458	0.24	0.06
MQ-based	MQ-Sign-RR†	328441	134	0.05	0.02
Hash-based	SPHINCS$^+$-128s*	32	7856	315.74	0.35
	SPHINCS$^+$-128f*	32	17088	16.32	0.97
MPCitH-based	aimer128s (ver.2.0)	32	4160	3.18	3.13
	aimer128f (ver.2.0)	32	5888	0.42	0.41

*: -SHAKE-simple

†: performances in CPU cycles are converted into ms

- Experiments are measured in Intel Xeon E5-1650 v3 @ 3.50GHz with 128 GB memory, AVX2 enabled
- A memory-optimized version requires up to 174 KB of memory for all the parameter sets, which fits well into ARM Cortex-M4

Thank you!
Check out our website!