MILP Modelings for Symmetric Cryptography and More

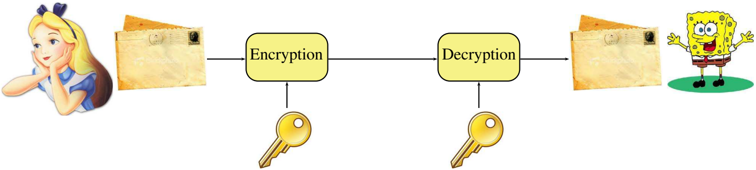
Christina Boura
(Joint-work with Daniel Coggia)

Université de Versailles and Inria Paris

29 April 2022

Symmetric-key encryption

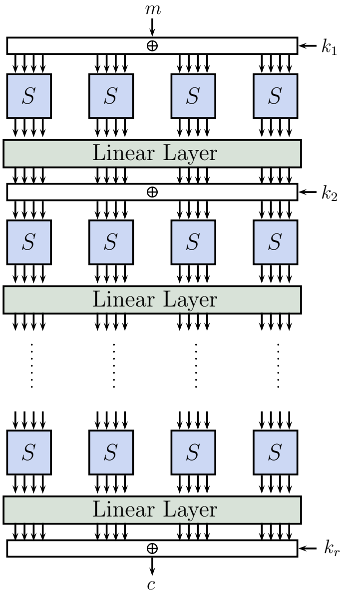
Alice and Bob share the **same secret key** for encryption and decryption.



Some well-known families of symmetric algorithms:

1. Stream ciphers
2. Block ciphers
3. Hash functions

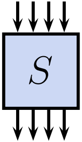
Substitution Permutation Network (SPN)



Sbox

An Sbox can be seen as a vectorial Boolean function

$$S : \mathbb{F}_2^n \to \mathbb{F}_2^m$$



- Typically $n = m$ and $n \in \{3, 4, 5, 6, 7, 8\}$

Example (Sbox of PRESENT)

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(x)$	12	5	6	11	9	0	10	13	3	14	15	8	4	7	1	2

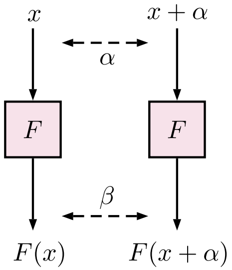
- An Sbox is usually the only **nonlinear** component of the cipher.
- Security arguments for the cipher heavily depend on the properties of the Sbox.

Differential attacks

Design strategy: A block cipher should resist **all** state-of-the-art attacks.

Differential cryptanalysis: one of the most prominent attacks against block ciphers [Biham - Shamir '90].

For an SPN cipher, the security against differential cryptanalysis **reduces** on the differential properties of the Sbox.



Difference Distribution Table (DDT)

$$DDT(\alpha, \beta) = \#\{x \in \mathbb{F}_2^n : F(x+\alpha) + F(x) = \beta\}$$

α/β	0	1	2	3	4	5	6	7
0	8	.	.	.	.	.	.	.
1	.	2	.	2	.	2	.	2
2	.	.	2	2	.	.	2	2
2	.	2	2	.	.	2	2	.
4	.	.	.	.	2	2	2	2
5	.	2	.	2	2	.	2	.
6	.	.	2	2	2	2	.	.
7	.	2	2	.	2	.	.	2

- Maximal differential probability $p_{max} = \frac{2}{2^3} = \frac{1}{4}$.

Mixed Integer Linear Programming (MILP)

Objectif	$c_1 x_1 + \cdots + c_n x_n$	$\boldsymbol{c} \cdot \boldsymbol{x}$
Constraints	$a_{1,1} x_1 + \cdots + a_{1,n} x_n \leq b_1$ $a_{2,1} x_1 + \cdots + a_{2,n} x_n \leq b_2$ $\vdots$ $a_{m,1} x_1 + \cdots + a_{m,n} x_n \leq b_m$	$\boldsymbol{A} \cdot \boldsymbol{x} \leq \boldsymbol{b}$
Domain	$x_1, \ldots, x_d \in \mathbb{Z}, \quad x_{d+1}, \ldots, x_n \in \mathbb{R}$	
	$x_1, \ldots, x_n \in \{0, 1\}$	

- Objective function and all constraints are linear.
- Some variables are integers, some variables are continuous.
- Typically in our applications, almost all variables are Boolean.

Example of a MILP Problem

$$
\begin{aligned}
\text{Minimize} \quad & -x_1 - x_2 \\
\text{Subject To} \quad & -2x_1 + 2x_2 \geq 1 \\
& -8x_1 + 10x_2 \leq 13 \\
\text{where} \quad & x_1, x_2 \in \mathbb{Z} \text{ and } x_1, x_2 \geq 0.
\end{aligned}
$$

Many good available solvers: Gurobi, CPLEX, ...

Modeling possible transitions through an Sbox

	0	1	2	3	4	5	6	7
0	8	.	.	.	.	.	.	.
1	.	2	.	2	.	2	.	2
2	.	.	2	2	.	.	2	2
3	.	2	2	*	.	2	2	*
4	.	.	.	.	2	2	2	2
5	.	2	.	2	2	.	2	.
6	.	.	2	2	2	2	.	.
7	.	2	2	*	2	.	.	2

Input diff. $x = (x_0, x_1, x_2)$
Output diff. $y = (y_0, y_1, y_2)$

$$-2x_0 - 2x_1 + x_2 - 2y_0 - 2y_1 + y_2 \geq -6$$

Modeling possible transitions through an Sbox

	0	1	2	3	4	5	6	7
0	8	.	.	.	.	.	.	.
1	.	2	.	2	.	2	.	2
2	.	.	2	2	.	.	2	2
3	.	2	2	.	.	2	2	.
4	.	.	.	.	2	2	2	2
5	.	2	.	2	2	*	2	*
6	.	.	2	2	2	2	.	.
7	.	2	2	.	2	*	.	2

Input diff. $x = (x_0, x_1, x_2, x_3)$
Output diff. $y = (y_0, y_1, y_2, y_3)$

$$-2x_0 - 2x_1 + x_2 - 2y_0 - 2y_1 + y_2 \geq -6$$

$$-2x_0 + x_1 - 2x_2 - 2y_0 + y_1 - 2y_2 \geq -6$$

Modeling possible transitions through an Sbox

	0	1	2	3	4	5	6	7
0	8	.	.	.	.	.	.	.
1	.	2	.	2	.	2	.	2
2	.	.	2	2	.	.	2	2
3	.	2	2	.	.	2	2	.
4	.	.	.	.	2	2	2	2
5	.	2	.	2	2	.	2	.
6	.	.	2	2	2	2	*	*
7	.	2	2	.	2	.	*	2

Input diff. $x = (x_0, x_1, x_2, x_3)$
Output diff. $y = (y_0, y_1, y_2, y_3)$

$$-2x_0 - 2x_1 + x_2 - 2y_0 - 2y_1 + y_2 \geq -6$$

$$-2x_0 + x_1 - 2x_2 - 2y_0 + y_1 - 2y_2 \geq -6$$

$$x_0 - 2x_1 - 2x_2 + y_0 - 2y_1 - 2y_2 \geq -6$$

Modeling possible transitions through an Sbox

	0	1	2	3	4	5	6	7
0	8	*	*	*	*	*	*	*
1	*	2	*	2	*	2	*	2
2	*	*	2	2	*	*	2	2
3	*	2	2	*	*	2	2	*
4	*	*	*	*	2	2	2	2
5	*	2	*	2	2	*	2	*
6	*	*	2	2	2	2	*	*
7	*	2	2	*	2	*	*	2

Input diff. $x = (x_0, x_1, x_2, x_3)$
Output diff. $y = (y_0, y_1, y_2, y_3)$

$$-2x_0 - 2x_1 + x_2 - 2y_0 - 2y_1 + y_2 \geq -6$$
$$-2x_0 + x_1 - 2x_2 - 2y_0 + y_1 - 2y_2 \geq -6$$
$$x_0 - 2x_1 - 2x_2 + y_0 - 2y_1 - 2y_2 \geq -6$$
$$x_0 + 2x_1 + 4x_2 + 3y_0 + 2y_1 - 4y_2 \geq 0$$
$$-3x_0 + 2x_1 - x_2 + 4y_0 + 2y_1 + 4y_2 \geq 0$$
$$4x_0 - 2x_1 + x_2 - 2y_0 + 4y_1 + 3y_2 \geq 0$$

Modeling a Boolean function

Modeling differential transitions through an Sbox, is equivalent to modeling the Boolean function.

$$\begin{array}{ccc} \mathbb{F}_2^{2n} & \rightarrow & \mathbb{F}_2 \\ (x,y) & \mapsto & \begin{cases} 0, & \text{if } \mathrm{DDT}(x,y)=0, \\ 1, & \text{otherwise.} \end{cases} \end{array}$$

General problem:

Construct an efficient MILP model for a given Boolean function.

Modeling: A two-step process

Goal: Model efficiently a Boolean function by a system of linear inequalities.

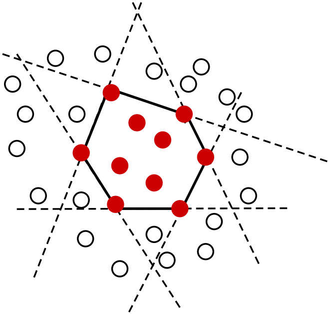
Two sub-problems:

- Problem 1 How to generate a (possibly large) set of inequalities that correctly models the function?
- Problem 2 How to choose a (typically much smaller) subset of this set of inequalities that still correctly represents the function but leads to more efficient MILP models?

Two different approches proposed in 2014 by Sun et al. for Problem 1:

1. Convex hull approach
2. Logical condition modeling

Convex Hull Method



Let F be an m-bit Boolean function.

Input of F : $(x_0, \ldots, x_{m-1}) \in \mathbb{R}^m$.

- Compute the H-representation of the convex hull of all possible transitions seen as vectors of $\mathbb{R}^m$.
- The $(m-1)$-dimensional faces of the convex hull yields a correct set of linear inequalities excluding all impossible points.

Compute the H-representation with an algebra computer system (eg. Sage).

Logical Condition Modeling

Let $a = (a_0, \ldots, a_{m-1}) \in \mathbb{F}_2^m$ be such that $F(a) = 0$. The inequality

$$\sum_{i=0}^{m-1} (1 - a_i)x_i + a_i(1 - x_i) = \sum_{i=0}^{m-1} x_i \oplus a_i \geq 1$$

only discards a.

Example Suppose $a =$ `0x16 = (100011)` $\in \mathbb{F}_2^6$ is such that $F(a) = 0$. Then,

$$-x_0 + x_1 + x_2 + x_3 - x_4 - x_5 \geq -2$$

is satisfied by all points in $\mathbb{F}_2^6$ but a.

- This method yields easily a system of inequalities with as many constraints as the number of points for which $F(a) = 0$.

Problem for large Boolean functions

Advantage: Both methods provide a solution for Problem 1, that is relatively efficient for small Boolean functions ($n \leq 10$).

Disadvantage: Not efficient for modeling 8-bit Sboxes (i.e. 16-bit Boolean functions), very popular in cryptography.

- Computing the convex hull for 16-bit Boolean functions is computationally hard.
- The second method yields a very high number of initial inequalities with by construction no hope for a correct subset for Problem 2.

For example:

`AES`	**33150** impossible transitions
`SKINNY-128`	**54067** impossible transitions

Modeling for large Sboxes

Abdelkhalek et al. made in 2017 a step forwards for the large function Boolean problem (8-bit Sboxes):

> Search for good inequalities for 16-bit Boolean functions
> $=$
> Minimize the product-of-sum representation of a Boolean function

Example

$x = (x_0, x_1, x_2)$	(000)	(100)	(010)	(110)	(001)	(000)	(011)	(111)
F(x)	0	0	1	0	1	1	0	1

$$(x_0 + x_1 + x_2)(\overline{x_0} + x_1 + x_2)(\overline{x_0} + \overline{x_1} + x_2)(x_0 + \overline{x_1} + \overline{x_2})$$

Quine-McCluskey (QM) algorithm

Minimize the product-of-sum representation of a Boolean function.

Example

$$(x_0 + x_1 + x_2)(\overline{x_0} + x_1 + x_2)(\overline{x_0} + \overline{x_1} + x_2)(x_0 + \overline{x_1} + \overline{x_2})$$

$$\Leftrightarrow$$

$$(x_1 + x_2)(\overline{x_0} + \overline{x_1} + x_2)(x_0 + \overline{x_1} + \overline{x_2})$$

Solve at once the two steps of the modelization problem:

1. Find many good inequalities (the prime implicants in the QM vocabulary)
2. Keep among them a good representative set.

About the QM approach

Advantages

1. First interesting method for 16-bit Boolean functions
2. Good results for some Sboxes (e.g. `SKINNY-128`)

But:

- QM needs high memory ressources and it can be slow.
- Some heuristic algorithm (e.g. Espresso) must be used instead.
- The number of inequalities given with this method for some functions is still too high to be efficient.

Algorithm	# impossible trans.	QM	Espresso
AES	33150	-	8302
SKINNY-128	54067	372	376

How to solve Problem 2

Once Problem 1 solved, one must choose among the initial set a good representative set for covering the Sbox (Problem 2).

- **Necessary step**: High number of inequalities $\Rightarrow$ important impact on the optimization time.
- **Not evident**: How to determine how many and which inequalities to keep?

Two approaches in the literature:

Approach 1 Greedy algorithm: Choose at each step the inequality removing the highest number of points.

Approach 2 Modelize Problem 2 as a MILP problem itself [Sasaki-Todo 17].

Our approach for Problem 2

- [Sasaki-Todo 2017]: The smallest subset of inequalities does not necessarily provide the overall best performance when running a complete cipher modeling.
- This auxiliary MILP problem can be too heavy when the initial set of inequalities is large.

Our approach: Use Approach 1 for our applications and Approach 2 for benchmarking reasons.

Our contributions

1. Different new heuristic methods for efficiently modeling large Sboxes
2. New better modelings for linear layers

Outline

Improved convex hull method for up to 12-bit functions

- Compute the H-representation of the convex hull of all points $a \in \mathbb{F}_2^m$ such that $F(a) = 1$.

$\Rightarrow$ Get a set of initial inequalities for F.

Idea: Compute other, potentially better*, linear inequalities from this initial set by summing up some of them.

* Better = Inequalities removing more points.

If $x \in \mathbb{F}_2^m$ satisfies the k inequalities $C_1, \ldots, C_k$:

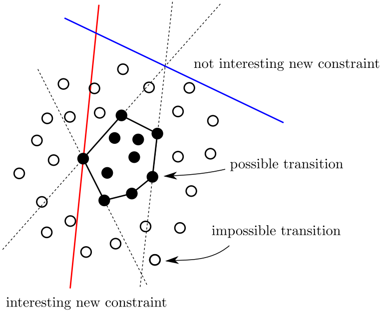
$$c_0^k x_0 + \cdots + c_{m-1}^k x_{m-1} + b_k \geq 0,$$

then it also satisfies

$$(\sum_{i=1}^{k} c_0^i) x_0 + \cdots + (\sum_{i=1}^{k} c_{m-1}^i) x_{m-1} + \sum_{i=1}^{k} b_i \geq 0$$

Produce meaningful inequalities

Most of the inequalities produced by randomly summing k inequalities are not interesting.



But, if k hyperplanes of the H-representation share a vertex on the cube $\{0,1\}^m$, (i.e. a possible transition), then the addition of the k corresponding inequalities will probably yield an interesting new inequality.

Results on 4-bit Sboxes

Sbox	# Inequalities			Sbox	# Inequalities		
	[SHW+14]	[ST17]	Our		[SHW+14]	[ST17]	Our
Present	22	21	**17**	Serpent S0	23	21	**17**
Klein	22	21	**19**	Serpent S1	24	21	**17**
Twine	23	23	**19**	Serpent S2	25	21	**18**
Prince	26	22	**19**	Serpent S3	31	27	**20**
Piccolo	23	21	**16**	Serpent S4	26	23	**19**
MIBS	27	23	**20**	Serpent S5	25	23	**19**
LBlock S0	28	24	**17**	Serpent S6	22	21	**17**
LBlock S1	27	24	**17**	Serpent S7	30	27	**20**
LBlock S2	27	24	**17**	Lilliput	–	23	**19**
LBlock S3	27	24	**17**	Minalpher	–	22	**19**
LBlock S4	28	24	**17**	Midori S0	–	21	**16**
LBlock S5	27	24	**17**	Midori S1	–	22	**20**
LBlock S6	27	24	**17**	Rectangle	–	21	**17**
LBlock S7	27	24	**17**	Skinny	–	21	**16**
LBlock S8	28	24	**17**	Gift	–	–	**17**
LBlock S9	27	24	**17**	Pride	–	–	**16**

Spaces of the form $a \oplus \mathsf{Prec}(u)$

For $u = (u_0, u_1, \ldots, u_{m-1}) \in \mathbb{F}_2^m$ denote by

- $\mathsf{supp}(u) = \{i : u_i = 1\} \subseteq \{0, m-1\}$.
- $$\mathsf{Prec}(u) = \{x \in \mathbb{F}_2^m : x \preceq u\},$$
  where $x \preceq u$ means that $x_i \leq u_i$ for all $i \in \{0, m-1\}$.

Example: $u = (0110)$: $\mathsf{Prec}(u) = \{(0000), (0100), (0010), (0110)\}$.

Goal: Derive inequalities to remove spaces of the form $a \oplus \mathsf{Prec}(u)$.

Inequalities for such spaces

Proposition: Let $a, u \in \mathbb{F}_2^m$ such that supp$(a) \bigcap$ supp$(u) = \emptyset$ and let $I = \{0, m-1\} \setminus \left(\text{supp}(a) \bigcup \text{supp}(u)\right)$. For all $x \in \mathbb{F}_2^m$,

$$-\sum_{i \in \text{supp}(a)} x_i + \sum_{i \in I} x_i \geq 1 - \text{wt}(a) \Leftrightarrow x \notin a \oplus \text{Prec}(u).$$

Inequalities for such spaces

Proposition: Let $a, u \in \mathbb{F}_2^m$ such that $\mathsf{supp}(a) \bigcap \mathsf{supp}(u) = \emptyset$ and let $I = \{0, m-1\} \setminus \left(\mathsf{supp}(a) \bigcup \mathsf{supp}(u)\right)$. For all $x \in \mathbb{F}_2^m$,

$$- \sum_{i \in \mathsf{supp}(a)} x_i + \sum_{i \in I} x_i \geq 1 - \mathsf{wt}(a) \Leftrightarrow x \notin a \oplus \mathsf{Prec}(u).$$

Example: Let $a = \texttt{0x1}, u = \texttt{0x94} \in \mathbb{F}_2^8$. Then,

$$\mathsf{Prec}(u) = \{\texttt{0x0}, \texttt{0x4}, \texttt{0x10}, \texttt{0x14}, \texttt{0x80}, \texttt{0x84}, \texttt{0x90}, \texttt{0x94}\}.$$

Further, as $\mathsf{supp}(a) = \{0\}$ and $\mathsf{supp}(u) = \{2, 4, 7\}$, $I = \{1, 3, 5, 6\}$. The equation

$$-x_0 + x_1 + x_3 + x_5 + x_6 \geq 0$$

removes the points

$$a \oplus \mathsf{Prec}(u) = \{\texttt{0x1}, \texttt{0x5}, \texttt{0x11}, \texttt{0x15}, \texttt{0x81}, \texttt{0x85}, \texttt{0x91}, \texttt{0x95}\}.$$

Relation with the Quine McCluskey algorithm

The Quine-McCluskey (QM) algorithm has two steps:

1. Finding all prime implicants of the function.
2. Use a prime implicant chart to find the prime implicants that are necessary to cover the function.

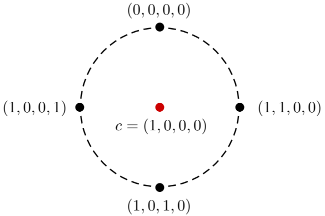
Remarks:

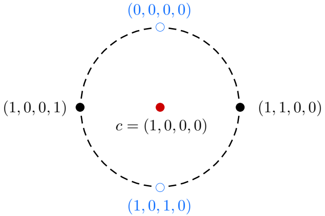
- The first step of QM corresponds to finding all spaces $a \oplus \mathsf{Prec}(u)$ (solving Problem 1).
- The second step of QM, corresponds to Problem 2. The way it is solved is very memory consuming and not efficient.

Our approach: Find all spaces $a \oplus \mathsf{Prec}(u)$ for solving Problem 1 together with a greedy algorithm or a MILP-based algorithm for solving Problem 2.
$\Rightarrow$ Faster + potentially much less inequalities.

Balls and distorted balls

$$\mathcal{B}(d,c) = \left\{ x \in \mathbb{F}_2^m \,\middle|\, \mathsf{wt}(x \oplus c) \leq d \right\}$$



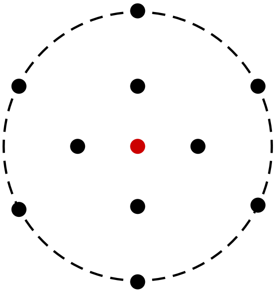


$$\mathcal{B}(1,c) = \{(1,0,0,0),(0,0,0,0),(1,1,0,0),(1,0,1,0),(1,0,0,1)\}.$$

All five points of the above ball can be removed by

$$(1-x_0)+x_1+x_2+x_3 \geq 2.$$

Discard a ball of radius d



Let $c \in \mathbb{F}_2^m$. The inequality

$$\sum_{i=0}^{m-1} (1-c_i)x_i + c_i(1-x_i) = \sum_{i=0}^{m-1} x_i \oplus c_i = \mathsf{wt}(x \oplus c) \geq d+1$$

removes all points in $\mathcal{B}(d, c)$.

Distorted balls

- Be sure not to remove points $a \in \mathbb{F}_2^m$ such that $F(a) = 1$ inside a ball.
- Useful if the table of F is dense (many 1's).

Exploit distorted balls!

Example: $\mathcal{DB} = \mathcal{B}(1, (1,0,0,0)) \setminus \{(0,0,0,0), (1,0,1,0)\}$.

Inequality removing $\mathcal{B}(1, (1,0,0,0))$: $(1 - x_0) + x_1 + x_2 + x_3 \geq 2$

The inequality

$$2(1 - x_0) + x_1 + 2x_2 + x_3 \geq 2$$

removes $\mathcal{DB}$.

Inequality corresponding to a distorted ball

Let $\mathcal{B}(d,c) \subset \mathbb{F}_2^m$ and $\mathcal{Q} = (c \oplus \mathsf{Prec}(q)) \bigcap \mathcal{S}(d,c)$. Lets $a \in \mathbb{Q}^m$ such that

$$a_i = \begin{cases} \frac{d+1}{d} & \text{if } q_i = 1, \\ 1 & \text{otherwise.} \end{cases}$$

Then the inequality

$$\sum_{i=0}^{m-1} a_i \big[(1-c_i)x_i + c_i(1-x_i)\big] \geq d+1$$

removes all points in $\mathcal{B}(d,c) \backslash \mathcal{Q}$.

Remove 3 distorted balls together

Example on PRESENT.

$$\mathcal{B}(1,[0,11]) = \{[0,11],[0,10],[0,9],[0,15],[0,3],[1,11],[2,11],[4,11],[8,11]\},$$
$$\mathcal{B}(1,[0,15]) = \{[0,15],[0,14],[0,13],[0,11],[0,7],[1,15],[2,15],[4,15],[8,15]\}$$
$$\mathcal{B}(1,[0,10]) = \{[0,10],[0,11],[0,8],[0,14],[0,2],[1,10],[2,10],[4,10],[8,10]\}.$$

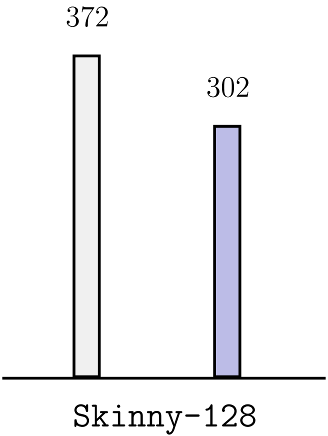
The inequality

$$3x_0+4x_1+4x_2+6x_3+2(1-y_0)+3(1-y_1)+y_2+3(1-y_3) \geq 6$$

removes the 17 points of

$$(\mathcal{B}(1,[0,11])\bigcup\mathcal{B}(1,[0,15])\bigcup\mathcal{B}(1,[0,10]))\setminus\{[2,10],[4,10],[8,10],[8,11],[8,15]\}.$$

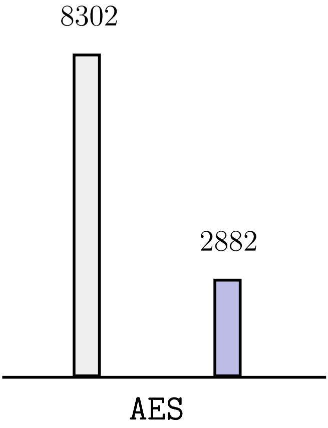
Results on 8-bit Sboxes



Quine-McCluskey

vs.

Combination of our new 3 methods



Outline

XOR modeling

- The XOR operation is the central element of most diffusion layers.

Proposition. Modeling $x_0 \oplus x_1 \oplus \ldots \oplus x_{n-1} = 0$ needs at least 2^{n-1} $\mathbb{R}$-linear inequalities.

A better way to modelize a matrix M

- A linear layer can be represented by a matrix M.

$$\begin{pmatrix} x_{n+1} \\ \vdots \\ x_{2n} \end{pmatrix} = M \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \Rightarrow \underbrace{(M|I)}_{A} \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_{2n} \end{pmatrix} = 0.$$

First Approach: Model the equation given by each row of A with the naive XOR modeling. $\Rightarrow$ **Inefficient**

Idea: Since for any matrix $P \in \mathsf{GL}_n(\mathbb{F}_2), \mathrm{Ker}(P \cdot A) = \mathrm{Ker}\, A$, find a matrix P that minimizes

$$\sum_{i=1}^{n} 2^{\mathsf{wt}(P \cdot A)_{i,\star} - 1}, \tag{1}$$

where $(P \cdot A)_{i,\star}$ is the i-th row of $P \cdot A$.

Application to SKINNY

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Naive modeling : $2^3 + 2 + 2^2 + 2^2 = \mathbf{18}$ inequalities

New modeling : $\mathbf{14}$ inequalities

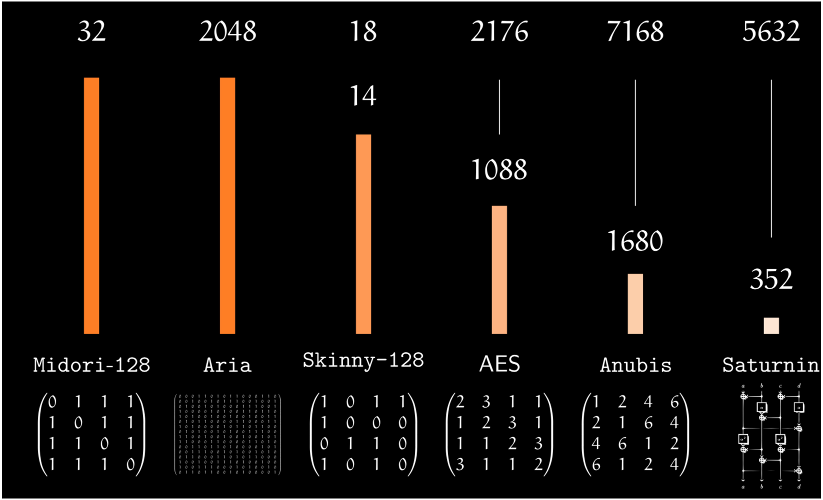
Changing the Sbox modeling for improving the linear one

- Find a block-diagonal matrix Q, an invertible matrix P, minimizing the modeling of

$$P \cdot (M|I) \cdot \begin{pmatrix} Q_1 & & \\ & \ddots & \\ & & Q_{2b} \end{pmatrix}$$

- Change S into $Q_i^{-1} \circ S \circ Q_{i+b}^{-1}$ for all $i \in [1, b]$

Results on different linear layers

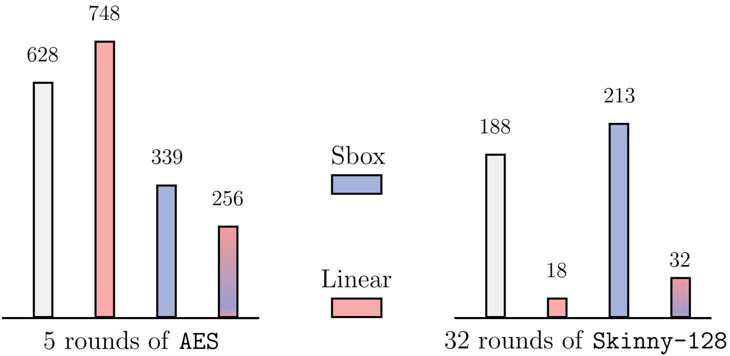


Slide by Daniel Coggia.

Outline

Applications

$$\begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ \alpha & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix} \xrightarrow{r \text{ rounds}} \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \beta & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix}$$



Open problems

- Provide more efficient modelization techniques.
- Understand what type of inequalities lead to a faster solving time.
- Understand and influence the solving strategies used by the solver.

Open problems

- Provide more efficient modelization techniques.
- Understand what type of inequalities lead to a faster solving time.
- Understand and influence the solving strategies used by the solver.

Thanks for your attention!