Practical cube-attack against **nonce-misused** ASCON

Fast Software Encryption (FSE) 2023

Jules BAUDRIN, Anne CANTEAUT & Léo PERRIN

Inria, Paris, France

Thursday, March 23 2023

Contact: jules.baudrin@inria.fr

In this talk

Ascon rationale, its internal components and our attack setting

Cube attack, main problems, first part of the answer

Conditional cubes, second part of the answer

Overview of the **internal-state recovery**

Ascon [DEMS19] design rationale

Authenticated encryption

$\rightarrow$ one of the winners of CAESAR (2014 – 2019).

Lightweight

> *"meets the needs of most use cases where lightweight cryptography is required"* [*NIST webpage*]

$\rightarrow$ winner of NIST LWC standardization process (2018 – 2023).

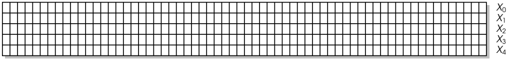
Permutation-based

Duplex Sponge mode [BDPA11] instantiated with permutation $p\colon \mathbb{F}_2^{320} \rightarrow \mathbb{F}_2^{320}$.

The permutation

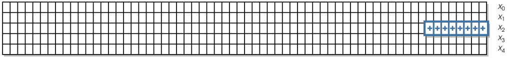
A confusion/diffusion structure…

The state

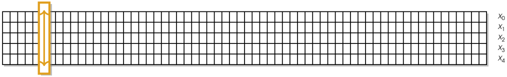


$$p = p_L \circ p_S \circ p_C$$

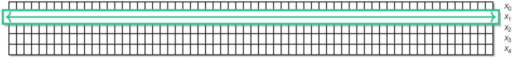
The constant addition p_C



The substitution layer p_S



The linear layer p_L



…studied algebraically

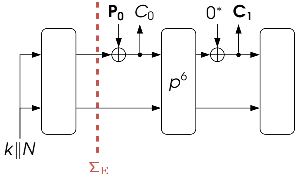
$$\begin{aligned}
y_0 &= x_4x_1 + x_3 + x_2x_1 + x_2 + x_1x_0 + x_1 + x_0\\
y_1 &= x_4 + x_3x_2 + x_3x_1 + x_3 + x_2x_1 + x_2 + x_1 + x_0\\
y_2 &= x_4x_3 + x_4 + x_2 + x_1 + 1\\
y_3 &= x_4x_0 + x_4 + x_3x_0 + x_3 + x_2 + x_1 + x_0\\
y_4 &= x_4x_1 + x_4 + x_3 + x_1x_0 + x_1
\end{aligned}$$

Algebraic Normal Form (ANF) of the S-box

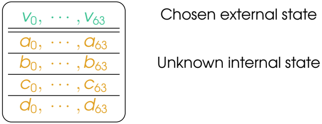
$$\begin{aligned}
X_0 &= X_0 \oplus (X_0 \ggg 19) \oplus (X_0 \ggg 28)\\
X_1 &= X_1 \oplus (X_1 \ggg 61) \oplus (X_1 \ggg 39)\\
X_2 &= X_2 \oplus (X_2 \ggg 1) \oplus (X_2 \ggg 6)\\
X_3 &= X_3 \oplus (X_3 \ggg 10) \oplus (X_3 \ggg 17)\\
X_4 &= X_4 \oplus (X_4 \ggg 7) \oplus (X_4 \ggg 41)
\end{aligned}$$

ANF of the linear layer p_L

Simplified setting of Ascon -128



Initialization Encryption



Σ_{E} State before encryption

- Many reuse of the same (k, N) pair.
- State recovery = compromised confidentiality without interaction.
- No trivial key-recovery nor forgery in that case.
- Different from the generic attack [VV18].

Cube attack principle

f_j: j-th output coordinate, $f_j \in \mathbb{F}_2[a_0, \cdots, a_{63}][v_0, \cdots, v_{63}]$.

$$f_j = \sum_{(u_0, \cdots, u_{63}) \in \mathbb{F}_2^{64}} \alpha_{u,\,j} \left(\prod_{i=0}^{63} {v_i}^{u_i} \right), \text{ where } \alpha_{u,\,j} \in \mathbb{F}_2[a_0, \cdots, a_{63}].$$

Cube attack principle

f_j: j-th output coordinate, $f_j \in \mathbb{F}_2[a_0, \cdots, a_{63}][v_0, \cdots, v_{63}]$.

$$f_j = \sum_{(u_0, \cdots, u_{63}) \in \mathbb{F}_2^{64}} \alpha_{u,\, j} \left(\prod_{i=0}^{63} {v_i}^{u_i} \right), \text{ where } \alpha_{u,\, j} \in \mathbb{F}_2[a_0, \cdots, a_{63}].$$

Polynomial expression of $\alpha_{u,\, j}$ + value of $\alpha_{u,\, j}$

$=$

equation in unknown variables

$\simeq$

recovery of some information

- Offline recovery of the expression.
- Online recovery of the value: $\alpha_{u,\, j} = \sum_{v \preccurlyeq u} f_j(v)$ $2^{w(u)}$ chosen queries.

Main problems with the polynomials recovery

Problem 0: impossible access to the full ANF.

Main problems with the polynomials recovery

Problem 0: impossible access to the full ANF.

Problem 1: impossible access to $\alpha_{u,\,j}$ expression for fixed u and j.
Too many combinatorial possibilities.

Main problems with the polynomials recovery

Problem 0: impossible access to the full ANF.

Problem 1: impossible access to $\alpha_{u,j}$ expression for fixed u and j.

Too many combinatorial possibilities.

$v_0 v_1 = v_0 \times v_1 = (v_0 v_1) \times 1 = (v_0 v_1) \times v_0 = (v_0 v_1) \times v_1 = (v_0 v_1) \times (v_0 v_1)$

Main problems with the polynomials recovery

Problem 0: impossible access to the full ANF.

Problem 1: impossible access to $\alpha_{u,\,j}$ expression for fixed u and j.
Too many combinatorial possibilities.

$v_0v_1 = v_0 \times v_1 = (v_0v_1) \times 1 = (v_0v_1) \times v_0 = (v_0v_1) \times v_1 = (v_0v_1) \times (v_0v_1)$

Problem 2: finding $\alpha_{u,\,j}$ with simple enough expressions.

Main problems with the polynomials recovery

Problem 0: impossible access to the full ANF.

Problem 1: impossible access to $\alpha_{u,\,j}$ expression for fixed u and j.
Too many combinatorial possibilities.

$v_0 v_1 = v_0 \times v_1 = (v_0 v_1) \times 1 = (v_0 v_1) \times v_0 = (v_0 v_1) \times v_1 = (v_0 v_1) \times (v_0 v_1)$

Problem 2: finding $\alpha_{u,\,j}$ with simple enough expressions.

- Highest-degree terms (degree 2^{t-1} at round t) are easier to study!

Strong constraint: products of two highest-degree terms one round before.

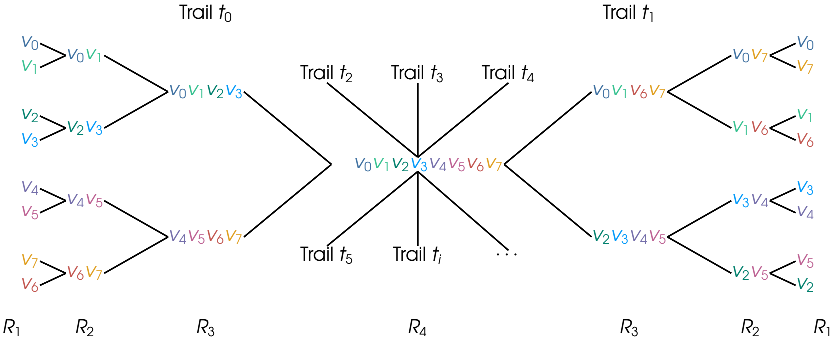
$v_0 v_1 = v_0 \times v_1 = \cancel{(v_0 v_1) \times 1} = \cancel{(v_0 v_1) \times v_0} = \cancel{(v_0 v_1) \times v_1} = \cancel{(v_0 v_1) \times (v_0 v_1)}$

A partial answer: highest-degree terms

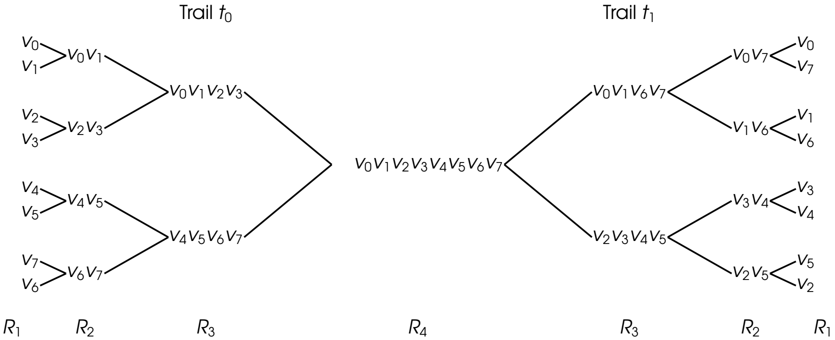
Strong constraint: products of two former highest-degree terms.

A partial answer: highest-degree terms

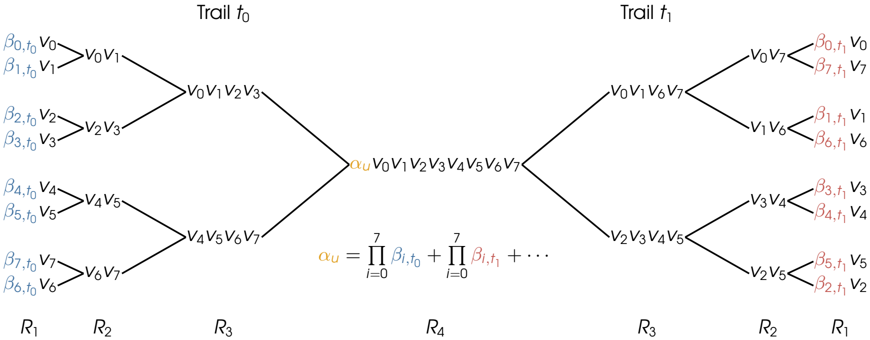
Strong constraint: products of two former highest-degree terms.



Strong constraint: products of two former highest-degree terms.



Strong constraint: products of two former highest-degree terms.



For r = 6, still too many trails and α_u usually looks horrible!

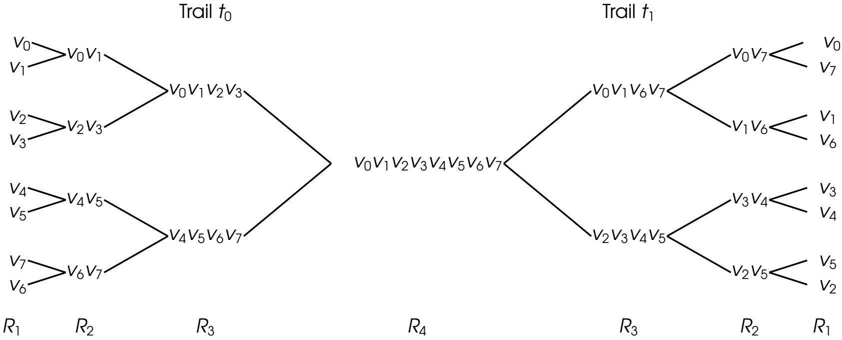
- Cheaper / easier recovery: conditional cubes [HWX+17, LDW17, CHK22]

Conditional cube

- We look for α_u with a simple divisor: β_0.
- α_u mostly unknown, but we still get: $\alpha_u = 1 \implies \beta_0 = 1$.
- If β_0 is linear, we get a linear system.

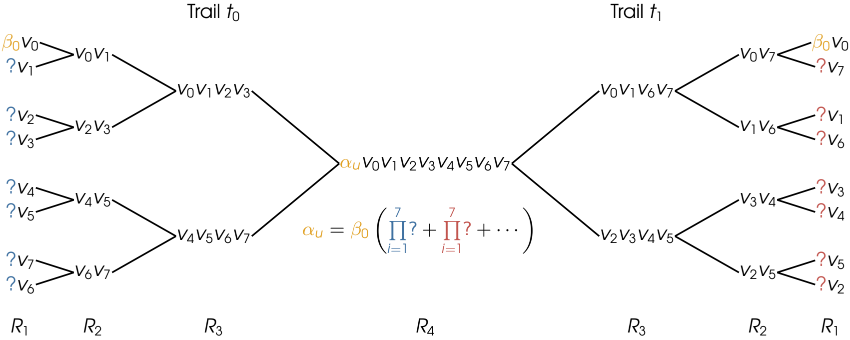
Conditional cube

- We look for α_u with a simple divisor: β_0.
- α_u mostly unknown, but we still get: $\alpha_u = 1 \implies \beta_0 = 1$.
- If β_0 is linear, we get a linear system.



Conditional cube

- We look for α_u with a simple divisor: β_0.
- α_u mostly unknown, but we still get: $\alpha_u = 1 \implies \beta_0 = 1$.
- If β_0 is linear, we get a linear system.



Choosing conditional cubes by forcing linear divisors

1^{st} round

$$\begin{array}{c} v_0 \\ \hline a_0 \\ b_0 \\ c_0 \\ d_0 \end{array} \xrightarrow{S_1} \begin{array}{c|l} (a_0+1)v_0 + \cdots & \longleftarrow \beta_0 := a_0 + 1 \\ \hline v_0 + \cdots & \\ \hline \cdots & \\ \hline (c_0+d_0+1)v_0 + \cdots & \longleftarrow \gamma_0 := c_0 + d_0 + 1 \\ \hline a_0 v_0 + \cdots & \end{array}$$

Choosing conditional cubes by forcing linear divisors

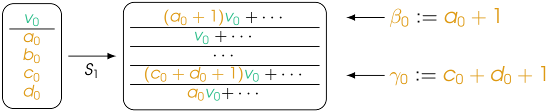
1^{st} round

$$\begin{array}{|c|}\hline v_0\\ \hline a_0\\ b_0\\ c_0\\ d_0\\ \hline\end{array} \xrightarrow{S_1} \begin{array}{|c|}\hline (a_0+1)v_0+\cdots\\ \hline v_0+\cdots \\ \hline \cdots \\ \hline (c_0+d_0+1)v_0+\cdots \\ \hline a_0v_0+\cdots\\ \hline\end{array} \quad \begin{array}{l} \longleftarrow \beta_0 := a_0+1 \\ \\ \\ \longleftarrow \gamma_0 := c_0+d_0+1 \\ \\ \end{array}$$

2^{nd} round

- For any $v_0 v_i$, $i \neq 0$: $\beta_0 P + 1Q + \gamma_0 R + (\beta_0 + 1)S$.
- But for some i: $\beta_0 P$ or $\gamma_0 R$.

1^{st} round



2^{nd} round

- For any $v_0 v_i$, $i \neq 0$: $\beta_0 P + 1Q + \gamma_0 R + (\beta_0 + 1)S$.
- But for some i: $\beta_0 P$ or $\gamma_0 R$.

6^{th} round

- With chosen u, $\alpha_{u,\,j} = \beta_0(\ldots) + \gamma_0(\ldots)$, for all output coordinates.

1^{st} round

$$\begin{array}{c} v_0 \\ \hline a_0 \\ b_0 \\ c_0 \\ d_0 \end{array} \xrightarrow{S_1} \begin{array}{c} (a_0+1)v_0 + \cdots \\ \hline v_0 + \cdots \\ \hline \cdots \\ \hline (c_0+d_0+1)v_0 + \cdots \\ \hline a_0 v_0 + \cdots \end{array} \quad \begin{array}{l} \longleftarrow \beta_0 := a_0 + 1 \\ \\ \\ \longleftarrow \gamma_0 := c_0 + d_0 + 1 \\ \\ \end{array}$$

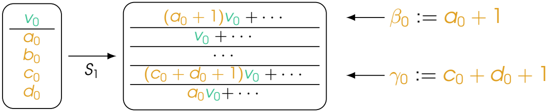
2^{nd} round

- For any $v_0 v_i$, $i \neq 0$: $\beta_0 P + 1Q + \gamma_0 R + (\beta_0 + 1)S$.
- But for some i: $\beta_0 P$ or $\gamma_0 R$.

6^{th} round

- With chosen u, $\alpha_{u,\,j} = \beta_0(\dots) + \gamma_0(\dots)$, for all output coordinates.
- $(\alpha_{u,0}, \cdots, \alpha_{u,63}) \neq (0, \cdots, 0) \implies \beta_0 = 1$ or $\gamma_0 = 1$

1^{st} round

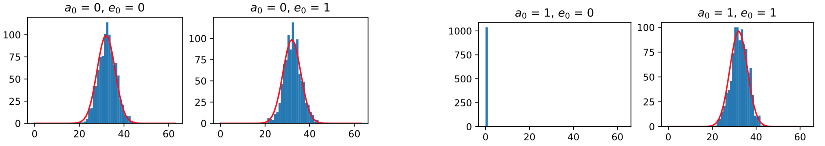


2^{nd} round

- For any $v_0 v_i$, $i \neq 0$: $\beta_0 P + 1Q + \gamma_0 R + (\beta_0 + 1)S$.
- But for some i: $\beta_0 P$ or $\gamma_0 R$.

6^{th} round

- With chosen u, $\alpha_{u,\,j} = \beta_0(\ldots) + \gamma_0(\ldots)$, for all output coordinates.
- $(\alpha_{u,0}, \cdots, \alpha_{u,63}) \neq (0, \cdots, 0) \implies \beta_0 = 1$ or $\gamma_0 = 1$
- In practice, reciprocal also true! $[\,\alpha_{u,\,j} = 0,\ \forall\, j\,] \implies \beta_0 = 0$ and $\gamma_0 = 0$



Step 1, non-adaptative: 32-degree conditional cubes

Recovery of all $\gamma_i = c_i + d_i + 1$, and half of the $\beta_i = a_i + 1$.

Overview of the internal-state recovery

Step 1, non-adaptative: 32-degree conditional cubes

Recovery of all $\gamma_i = c_i + d_i + 1$, and half of the $\beta_i = a_i + 1$.

Step 2, adaptative: 32-degree cubes

- The coefficients depend only on γ_i and β_i.
- Thanks to Step 1, the coefficients drastically simplifies.
- Simple-enough to be effectively-solved (Cryptominisat, [SNC09]).

- Recovery of the remaining β_i.

Overview of the internal-state recovery

Step 1, non-adaptative: 32-degree conditional cubes

Recovery of all $\gamma_i = c_i + d_i + 1$, and half of the $\beta_i = a_i + 1$.

Step 2, adaptative: 32-degree cubes

- The coefficients depend only on γ_i and β_i.
- Thanks to Step 1, the coefficients drastically simplifies.
- Simple-enough to be effectively-solved (Cryptominisat, [SNC09]).

▶ Recovery of the remaining β_i.

Step 3, adaptative: **31-degree cubes**

- Other cubes needed to recover b_i and c_i.
- Same principle as Step 2.

▶ Recovery of all b_i and c_i.

Conclusion

- Full-state recovery on the full 6-round encryption.
- About 2^{40} online time and data, but nonce-misuse.
- Hard to study the complexity of the solving of equations. However effective.
- Does not threaten Ascon directly … if used properly!

Main questions/openings

- Be careful with implementation : nonce $\neq$ constant!
- Can it lead to key-recovery or forgery attacks?
- Free counter-measure : changing the external state row.

Conclusion

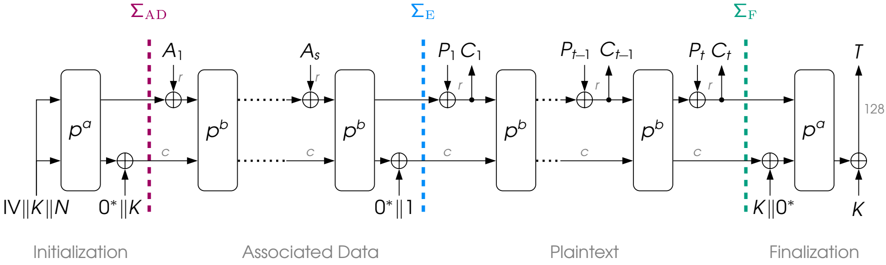
- Full-state recovery on the full 6-round encryption.
- About 2^{40} online time and data, but nonce-misuse.
- Hard to study the complexity of the solving of equations.
  However effective.
- Does not threaten Ascon directly ... if used properly!

Main questions/openings

- Be careful with implementation : nonce $\neq$ constant!
- Can it lead to key-recovery or forgery attacks?
- Free counter-measure : changing the external state row.

Thank you for
your attention!

The whole ASCON AEAD mode

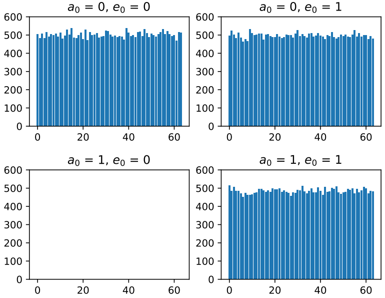


[DEMS, Jea16]

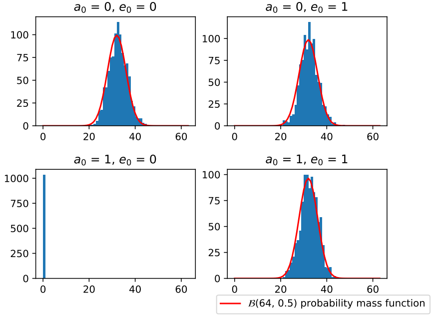
$$\alpha_{u,\,j} = (a_0 + 1)p_{j,1} + (c_0 + d_0 + 1)p_{j,2} \;\forall\, j \in [\![0, \cdots, 63]\!].$$

When $(a_0 + 1, c_0 + d_0 + 1) \neq (0, 0)$, $\alpha_{u,\,j}$ are not expected to be **all** canceled at the same time.

Whenever we observe that $\alpha_{u,\,j} = 0 \;\forall\, j$, we guess that $(a_0, c_0 + d_0) = (1, 1)$.



Individual cancellations of each $\alpha_{u,j}$
(1000 random internal states)



Hamming weight of the cube-sum vectors
(1000 random internal states)

Counter-Measure: Changing the Input Row

State after initialization	Linear terms after S_1	Size of the sets	Analysis
a_0	$(a_0 + b_0 + d_0 + 1)v_0$	5	$5 + 3 + 5 + 12 < 31$ No conditional cube as we describe.
v_0	$(b_0 + c_0 + 1)v_0$	3	
b_0	v_0		
c_0	v_0		
d_0	$(a_0 + d_0 + 1)v_0$	5	
Nb of variables not multiplied by v_0 after S_2		12	
a_0	$(b_0 + 1)v_0$	4	$4 + 6 + 23 > 31$. Cubes can be built as described but less effective. (32 of the 256-bit state in avg.)
b_0	$(b_0 + c_0 + 1)v_0$	6	
v_0	v_0		
c_0	v_0		
d_0	*		
Nb of variables not multiplied by v_0 after S_2		23	

Table: Example : the first row states that, for 5 indices i, the coefficients of all $v_0 v_i$ share $(a_0 + b_0 + d_0 + 1)$ as a factor.

Counter-Measure: Changing the Input Row

<table>
<tr><th>State after initialization</th><th>Linear terms after S_1</th><th>Size of the sets</th><th>Analysis</th></tr>
<tr><td>a_0
b_0
c_0
v_0
d_0</td><td>v_0
$(b_0 + c_0 + 1)v_0$
$d_0 v_0$
$(a_0 + 1)v_0$
v_0</td><td>
3
4
5
</td><td rowspan="2">$3 + 4 + 5 + 12 < 31$
No conditional cube
as we describe.</td></tr>
<tr><td colspan="2">Nb of variables not multiplied by v_0 after S_2</td><td>12</td></tr>
<tr><td>a_0
b_0
c_0
d_0
v_0</td><td>$b_0 v_0$
v_0
$(d_0 + 1)v_0$
$(a_0 + 1)v_0$
$(b_0 + 1)v_0$</td><td>5

4
5
5</td><td>$5 + 4 + 5 + 5 + 12 = 31$
but b_0 and $b_0 + 1$ cannot be used at the same time.</td></tr>
<tr><td colspan="2">Nb of variables not multiplied by v_0 after S_2</td><td>12</td><td>No conditional cube
as we describe.</td></tr>
</table>

Table: Example : the second row states that, for 3 indices i, the coefficients of all $v_0 v_i$ share $(b_0 + c_0 + 1)$ as a factor.

Bibliography

Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche.
Cryptographic sponge functions, 2011.
https://keccak.team/sponge_duplex.html.

Donghoon Chang, Deukjo Hong, and Jinkeon Kang.
Conditional Cube Attacks on Ascon-128 and Ascon-80pq in a Nonce-misuse Setting.
Cryptology ePrint Archive, Report 2022/544, 2022.
https://ia.cr/2022/544.

Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer.
Ascon TikZ figures.
https://ascon.iaik.tugraz.at/resources.html.

Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer.
Ascon v1.2.
Technical report, National Institute of Standards and Technology, 2019.
https://csrc.nist.gov/Projects/lightweight-cryptography/finalists.

Senyang Huang, Xiaoyun Wang, Guangwu Xu, Meiqin Wang, and Jingyuan Zhao.
Conditional cube attack on reduced-round Keccak sponge function.
In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 259–288, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany.

Jérémy Jean.
TikZ for Cryptographers.
https://www.iacr.org/authors/tikz/, 2016.

Zheng Li, Xiaoyang Dong, and Xiaoyun Wang.
Conditional cube attack on round-reduced ASCON.
IACR Trans. Symm. Cryptol., 2017(1):175–202, 2017.

Mate Soos, Karsten Nohl, and Claude Castelluccia.
Extending SAT solvers to cryptographic problems.
In Oliver Kullmann, editor, *Theory and Applications of Satisfiability Testing - SAT 2009*, volume 5584 of *Lecture Notes in Computer Science*, pages 244–257. Springer, 2009.

Serge Vaudenay and Damian Vizár.
Can caesar beat galois? - Robustness of CAESAR candidates against nonce reusing and high data complexity attacks.
In Bart Preneel and Frederik Vercauteren, editors, *ACNS 18*, volume 10892 of *LNCS*, pages 476–494, Leuven, Belgium, July 2–4, 2018. Springer, Heidelberg, Germany.