Single-Trace Side-Channel Attacks on the Toom-Cook

The Case Study of Saber

Yanbin Li[1], Jiajie Zhu [1], Yuxin Huang [1], Zhe Liu [2,3], and Ming Tang[4]

[1] *Nanjing Agricultural University,* [2] *Zhejiang Lab,* [3] *Nanjing University of Aeronautics and Astronautics,* [4] *Wuhan University*

CHES 2022, September 2022

Overview

1. Toom-Cook Overview
2. Vulnerabilities Analysis
3. Single-trace Attack
4. Evaluation
5. Conclusion

Toom-Cook

- Toom-Cook algorithm
  - A divide-and-conquer approach to implementing polynomial multiplication
- Toom-Cook-k
  - k segments to form a $k-1$ degree polynomial containing k coefficients
  - Karatsuba algorithm, a special form of Toom-Cook-2 algorithm
- NTRU-Prime and Saber

Toom-Cook-4

- $A(x)$ and $B(x)$: n-degree polynomials
  - $A(x) = a_{n-1} \cdot x^{n-1} + a_{n-2} \cdot x^{n-2} + \cdots + a_0$
  - $B(x) = b_{n-1} \cdot x^{n-1} + b_{n-2} \cdot x^{n-2} + \cdots + b_0$
- The parameter $n = 256$ and $k = 4$
  - $A(x) = A3 \cdot x^{64 \cdot 3} + A2 \cdot x^{64 \cdot 2} + A1 \cdot x^{64} + A0$
  - $B(x) = B3 \cdot x^{64 \cdot 3} + B2 \cdot x^{64 \cdot 2} + B1 \cdot x^{64} + B0$
    - $A3 = a_{255} \cdot x^{63} + \cdots + a_{192},\ A2 = a_{191} \cdot x^{63} + \cdots + a_{128}$
    - $A1 = a_{127} \cdot x^{63} + \cdots + a_{64},\ A0 = a_{63} \cdot x^{63} + \cdots + a_0$
- Define $x^{64} = y$
  - $A(y) = A3 \cdot y^3 + A2 \cdot y^2 + A1 \cdot y + A0$
  - $B(y) = B3 \cdot y^3 + B2 \cdot y^2 + B1 \cdot y + B0$

Toom-Cook-4

- $C(p_i) = A(p_i) \cdot B(p_i)$
  - $p_0 = 0, p_1 = 1/2, p_2 = -1/2, p_3 = 1, p_4 = -1, p_5 = 2, p_6 = \infty$
- $$\begin{bmatrix} C_0 \\ C_1 \\ \vdots \\ C_6 \end{bmatrix} = \begin{bmatrix} (p_0)^0 & (p_0)^1 & \cdots & (p_0)^6 \\ (p_1)^0 & (p_1)^1 & \cdots & (p_1)^6 \\ \vdots & \vdots & \ddots & \vdots \\ (p_6)^0 & (p_6)^1 & \cdots & (p_6)^6 \end{bmatrix}^{-1} \cdot \begin{bmatrix} C(p_0) \\ C(p_1) \\ \vdots \\ C(p_6) \end{bmatrix}$$
- $C(y) = C_6 \cdot y^6 + C_5 \cdot y^5 + \cdots + C_0$

Toom-Cook in Saber

```
void indcpa_kem_dec(const uint8_t sk[], const uint8_t ciphertext[], uint8_t m[])
   1. BS2POLVECq(sk, s); BS2POLVECp(ciphertext, b);
   2. InnerProd(b, s, v);
   3. /*processing results*/
void InnerProd(const uint16_t b[][], const uint16_t s[][], uint16_t res[])
   1. for (j = 0; j < SABER_L; j++)   poly_mul_acc(b[j], s[j], res);
void poly_mul_acc(const uint16_t a[], const uint16_t b[], uint16_t res[])
   1.toom_cook_4way(a, b, c);
static void toom_cook_4way(const uint16_t *a1, const uint16_t *b1, uint16_t *result)
   1. Split a1 to A0, A1, A2, A3; Split b1 to B0, B1, B2, B3;
   2. Calculate 7 points       //Evaluation
      aw1=A3;                          bw1=B3;
      aw2=8A3+4A2+2A1+A0;              bw2=8B3+4B2+2B1+B0;
      aw3=A0+A2+A1+A3;                 bw3=B0+B2+B1+B3;
      aw4=A0+A2-(A1+A3);               bw4=B0+B2-(B1+B3);
      aw5=8A0+2A2+4A1+A3;              bw5=8B0+2B2+4B1+B3;
      aw6=8A0+2A2-(4A1+A3);            bw6=8B0+2B2-(4B1+B3);
      aw7=A0;                          bw7=B0;
   3. karatsuba_simple(aw1, bw1, w1);…; karatsuba_simple(aw7, bw7, w7);   //MULTIPLICATION
   4. /*INTERPOLATION*/
static void karatsuba_simple(const uint16_t *a_1, const uint16_t *b_1, uint16_t *result_final)
   1. for (i = 0; i < 16; i++)
   2.    acc1=a_1[i]; acc2=a_1[i+16]; acc3=a_1[i+32]; acc4=a_1[i+48];
   3.      for (j = 0; j< 16; j++)
   4.          acc5=b_1[j]; acc6=b_1[j+16];
   5.          result_final[i+j]=result_final[i+j]+OVERFLOWING_MUL(acc1, acc5);
   6.          /*The same method to calculate the 9 multiplications in 2-level Karatsuba*/
   7.  /*processing the results*/
```

Vulnerabilities Analysis

- Incomplete key recovery
  - Its intermediate values depend on the known ciphertext and unknown secret key.
  - Reveal the first and last $\frac{1}{k}$ of private-key coefficients
- Indistinguishable guessing keys

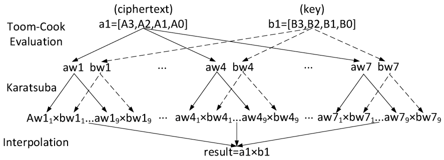


Figure: The dataflow of Toom-Cook multiplication in Saber.

s_{coeff}		guessing key						
		1	2	3	8188	8189	8190	8191
correct key	1	1	1	0.48	0.75	0.14	0.75	0.74
	2	1	1	0.48	0.75	0.14	0.75	0.74
	3	0.48	0.48	1	0.33	0.79	0.33	0.33
	8188	0.75	0.75	0.33	1	0.42	0.99	0.99
	8189	0.14	0.14	0.79	0.42	1	0.42	0.43
	8190	0.75	0.75	0.33	0.99	0.42	1	0.99
	8191	0.74	0.74	0.33	0.99	0.43	0.99	1

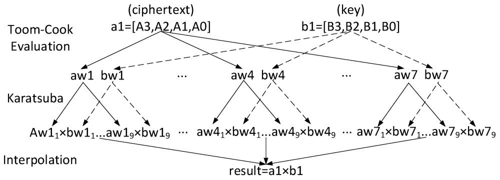
Figure: The Pearsons correlation coefficient among different guessing keys.

Soft-analytical side-channel attack (SASCA)

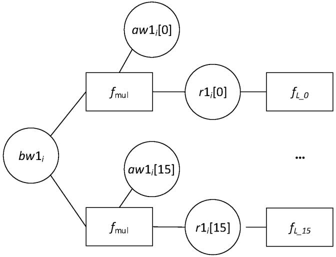
- Factor graphs
  - Variables nodes by circles
  - Factor nodes by squares (two groups)
    - Corresponds to the probabilities of the variables by observable side-channel leakages
    - Modeling the relationships between the variables nodes
- Belief propagation
  - $u_{x_n \to f_m}(v_n) = \prod_{m' \in \mathcal{M}(x_n) \setminus m} u_{f_{m'} \to x_n}(v_n)$
  - $u_{f_m \to x_n}(v_n) = \sum_{\mathrm{x}_{m \setminus n}} \left(f_m(\mathrm{x}_{m \setminus n}, v_n) \prod_{n' \in \mathcal{N}(f_m) \setminus n} u_{x_{n'} \to f_m}(v_{n'}) \right)$

SASCA on Toom-Cook

- Schoolbook multiplication with factor graph representation (SFG)
  - $f_{mul}(aw1_i[0], bw1_i, r1_i) = \begin{cases} 1 & \textit{if } \; r1_i[0] = OVERFLOWING\_MUL(aw1_i[0], bw1_i) \\ 0 & \textit{otherwise} \end{cases}$
  - $f_{L\_0} = Pr(r1_i[0] | L\_0)$



(a) aw and bw.



(b) SFG.

SASCA on Toom-Cook

- Factor graph corresponding to Karatsuba (KFG)
  - $f_{add}^{1}(bw1_1, bw1_2, bw1_3) = \begin{cases} 1 & \textit{if } bw1_3 = bw1_1 + bw1_2 \textit{ mod } q \\ 0 & \textit{otherwise} \end{cases}$

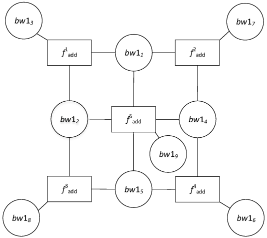


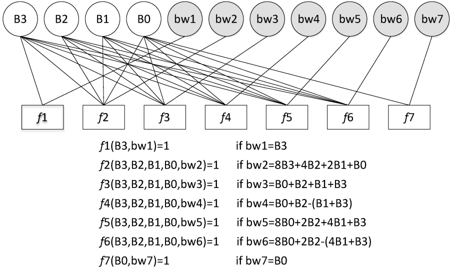
Figure: KFG.

$$
\begin{aligned}
bw1_1 &= bw1\_3 \\
bw1_2 &= bw1\_2 \\
bw1_3 &= bw1\_3 + bw1\_2 \\
bw1_4 &= bw1\_1 \\
bw1_5 &= bw1\_0 \\
bw1_6 &= bw1\_1 + bw1\_0 \\
bw1_7 &= bw1\_3 + bw1\_1 \\
bw1_8 &= bw1\_2 + bw1\_0 \\
bw1_9 &= bw1\_3 + bw1\_2 + bw1\_1 + bw1\_0
\end{aligned}
$$

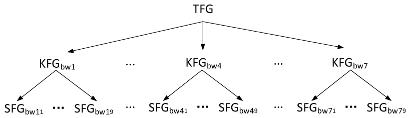
Figure: The 9 polynomials of degree 16.

SASCA on Toom-Cook

- Factor graph corresponding to Toom-Cook evaluation (TFG)
- The construction of the full algorithm



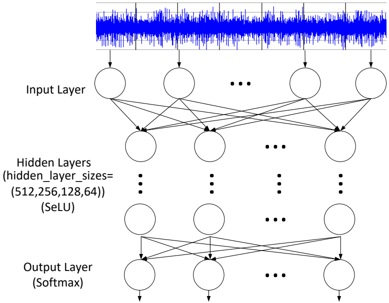
(a) TFG.



(b) Relationships.

Decreasing the Number of Templates

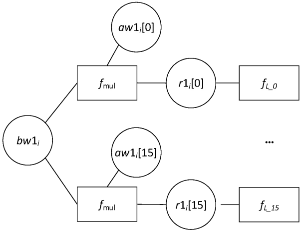
- Original templates: $2^{16} \cdot 144$, $f_{L\_0} = Pr(r1_i[0] = v|I)$
- Hamming weight templates: $7 \cdot 144 \cdot 17 = 17136$, $f_{L\_0} = Pr(HW(r1_i[0]) = HW(v)|I)$
- Deep Learning: MLP



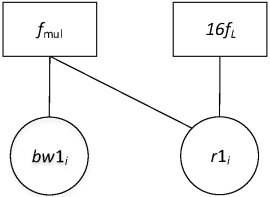
Factor Graph Optimization

- Cost: influenced by the number of nodes and edges of factor graph
- $p(bw1_i) = p(bw1_i|t_0) \cdot p(bw1_i|t_1) \ldots p(bw1_i|t_{15}) = p(bw1_i|t_0, \ldots t_{15}) \cdot \mathcal{C}$

$$\mathcal{C} = \frac{\sum_l ((\prod_j p(t_j|bw1_i^l))p(bw1_i^l)) \prod_j p(bw1_i)}{\prod_j ((\sum_l p(t_j|bw1_i^l))p(bw1_i^l))}$$



(c) Original SFG



(d) Bayes-based SFG

Improving Belief Propagation

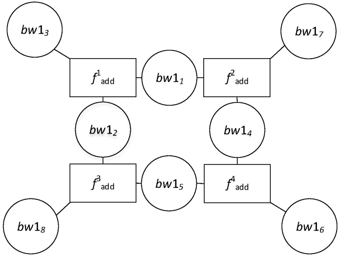
- In LDPC, short cycles especially, cycles of length 4, influence the performance using the BP algorithm [Chung et al, 2006]
- Parity-check matrix

$$H = \begin{bmatrix} \mathbf{1} & \mathbf{1} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & \mathbf{1} & 0 & 0 & 1 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & \mathbf{1} & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 1 & 0 & 0 & 0 \\ \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 & 0 & 1 \end{bmatrix}$$

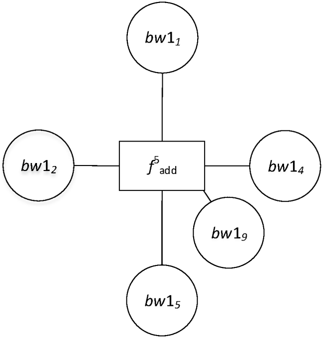
Kyuhyuk Chung and Jun Heo (2006)
Improved Belief Propagation (BP) Decoding for LDPC Codes with a large number of short cycles
2006 IEEE 63rd Vehicular Technology Conference 3, 1464 – 1466.

Improving Belief Propagation

- Avoid those shortest cycles of length 4
- Two steps of BP



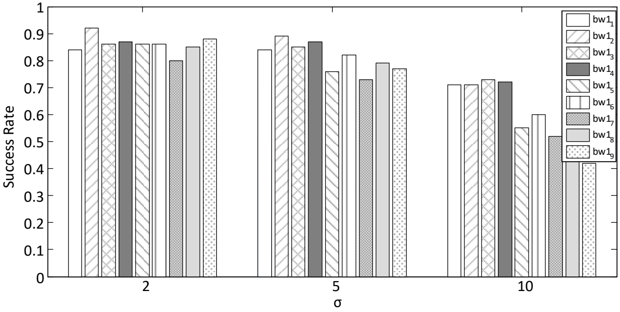
(e) First step of BP on the subgraph



(f) Second step of BP on the subgraph

Evaluation

- Evaluate the success rates under different noise levels
- Success rates of attacking $bw1_1, \ldots, bw1_9$



Evaluation

- Evaluate the Bayes-based SFG

metric	method	$bw1$	$bw2$	$bw3$	$bw4$	$bw5$	$bw6$	$bw7$	sum
success rate	Original SFG	0.86	0.88	0.83	0.88	0.87	0.87	0.86	0.86
	Bayes-based SFG	0.86	0.88	0.83	0.88	0.87	0.87	0.86	0.86
time(s)	Original SFG	1.88	4.12	1.86	2.30	3.71	3.79	2.43	20.08
	Bayes-based SFG	0.10	2.68	0.47	0.49	2.66	2.81	0.09	9.30

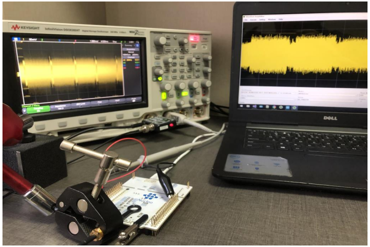
Evaluation

- Evaluate the improved BP algorithm

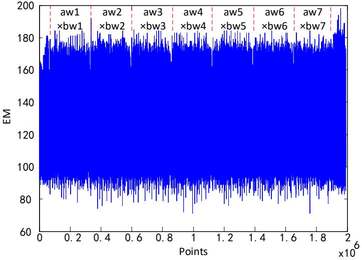
metric	success rate					
noise	2		5		10	
method	Original	Improved BP	Original	Improved BP	Original	Improved BP
*bw*1_3	0.84	0.94	0.81	0.95	0.71	0.81
*bw*1_2	0.92	0.94	0.80	0.94	0.71	0.80
*bw*1_1	0.86	0.97	0.68	0.97	0.73	0.87
*bw*1_0	0.87	0.94	0.67	0.95	0.72	0.78
metric	time in seconds					
noise	2		5		10	
method	Original	Improved BP	Original	Improved BP	Original	Improved BP
time	0.12	0.07	0.18	0.07	0.13	0.06

Evaluation

- The measured EM trace of implementation



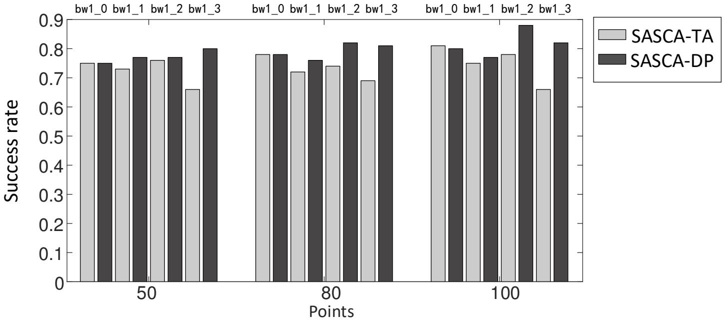
(g) Measurement setup.



(h) EM trace.

Evaluation

- Evaluate the practical attacks with MLP



Conclusion

- Investigate the security of the Toom-Cook
- Single-trace attacks
- Optimized SASCA

THANK YOU!