

On the Practical Security of White-Box Cryptography

Thesis Defense

by Junwei Wang (王军委)

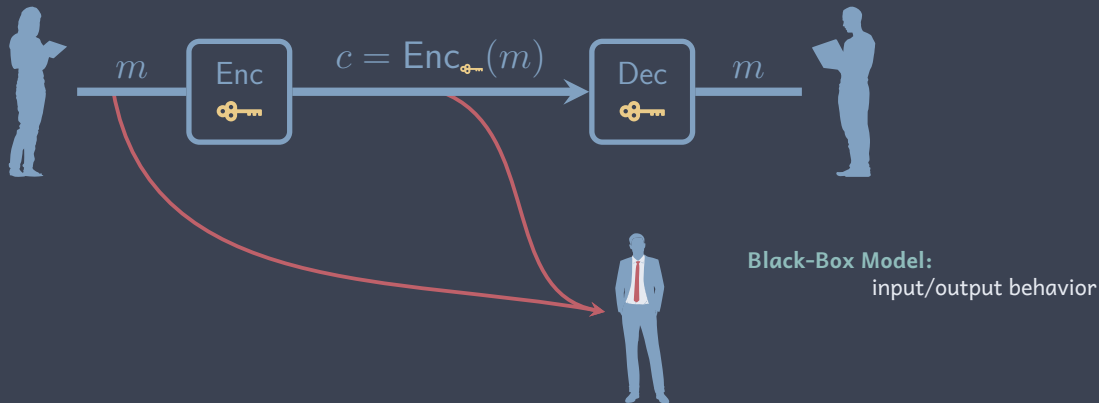
on June 24, 2020

Supervisors Jean-Sébastien Coron, Sihem Mesnager

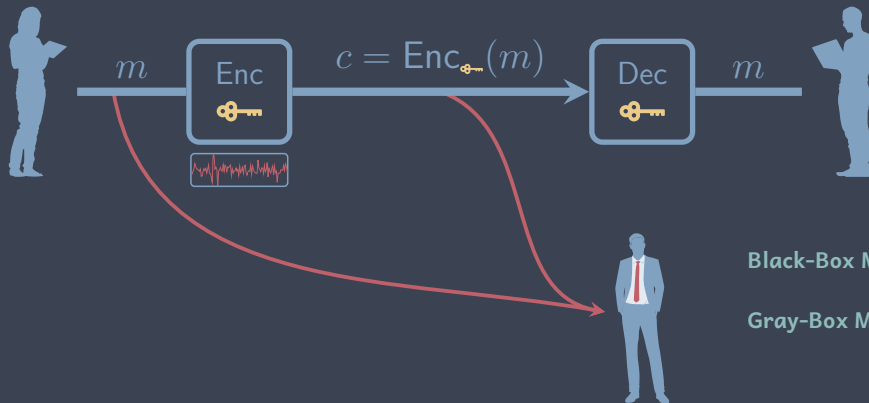
Advisors Pascal Paillier, Matthieu Rivain



» Security Models: Shades of Gray



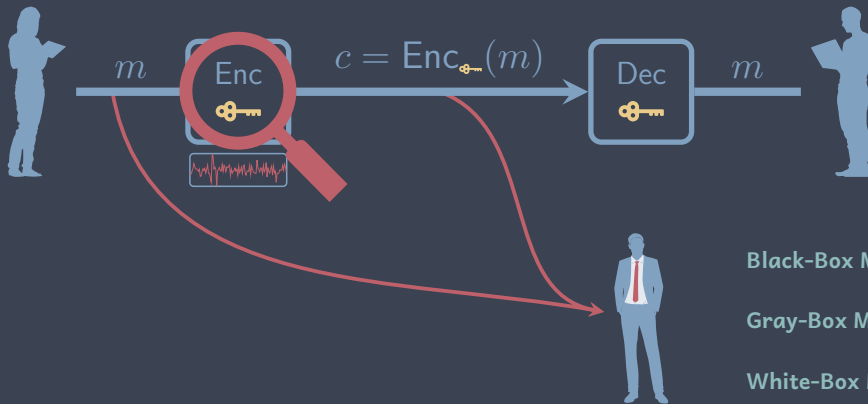
» Security Models: Shades of Gray



Black-Box Model:
input/output behavior

Gray-Box Model:
side-channel leakage

» Security Models: Shades of Gray



- Black-Box Model:**
input/output behavior
- Gray-Box Model:**
side-channel leakage
- White-Box Model:**
“full” control

» White-Box Threat Model

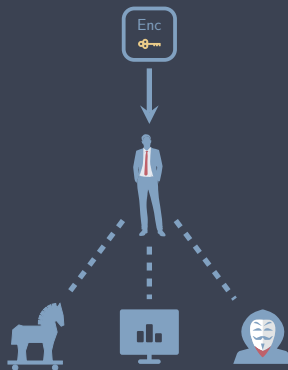
To extract a cryptographic key

Where from a software implementation of cipher

Whom by malwares, co-hosted applications, user themselves, ...

How by all kinds of means

- * analyze the code
- * spy on the memory
- * interfere the execution
- * ...



» Motivation and Real-World Applications

- * Why not using secure hardware ?
 - * not always available
 - * expensive (to produce, deploy, integrate, update)
 - * usually has a long lifecycle
 - * security breach is hard to mitigate
- * Applications
 - * Digital Content Distribution
 - * Mobile Payment
 - * Digital Contract Signing
 - * Blockchains and cryptocurrencies



Credits to [Shamir, van Someren 99]

» White-Box Compiler

[Delerablée et al. 14]

A **white-box compiler** takes as input a *secret key* and generates a “white-box secure” program implementing some specific crypto. algo. with the specified secret key.

* White-box security notions

- * **Unbreakability** (hardness of key-extraction)
- * One-wayness
- * Incompressibility
- * Traceability



No provably secure unbreakable white-box compiler for standard block ciphers is known.

» Historical White-Box Compilers

Transform a cipher into a series of randomized key-dependent lookup tables.

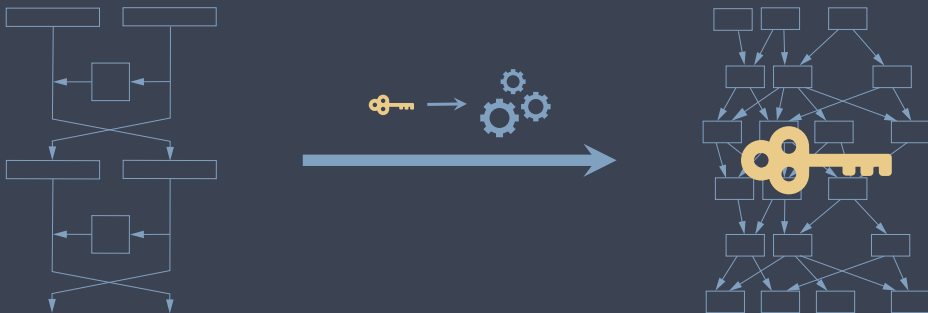
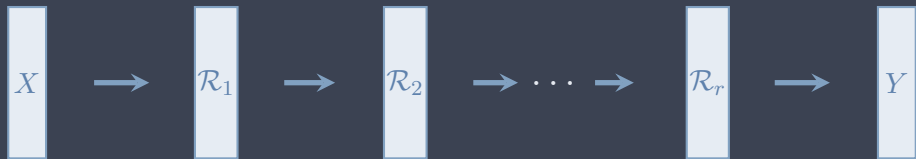
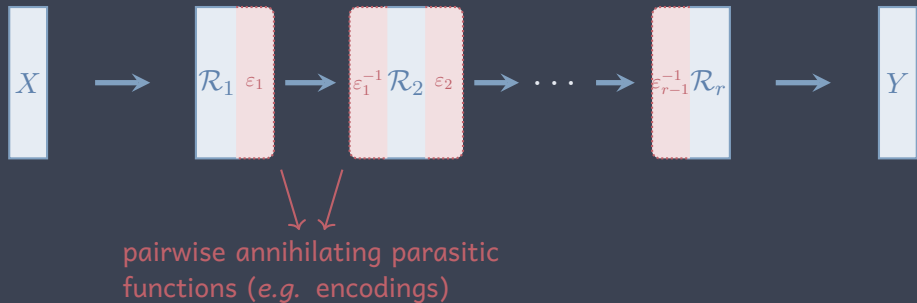


Illustration from [Wyseur12]

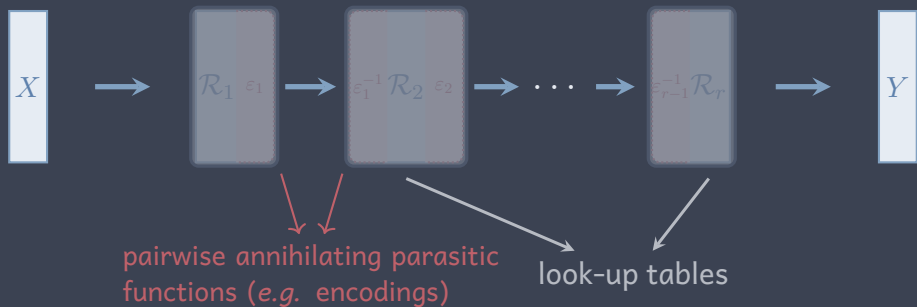
» Historical White-Box Compilers: Internal Encodings



» Historical White-Box Compilers: Internal Encodings



» Historical White-Box Compilers: Internal Encodings



» Timeline: A Cat-And-Mouse Game



- * 2002: seminal wb-DES [Chow et al. 02]
- * 2003: seminal wb-AES [Chow et al. 03]
- * 2005: variant wb-DES [Link, Neumann 05]
- * 2006: variant wb-AES [Bringer et al. 06]
- * 2009: variant wb-AES [Xiao, Lai 09]
- * 2010: variant wb-AES [Karroumi 10]
- * ...
- * 2002: fault attack against wb-DES [Jacob et al. 02]
- * 2004: BGE attack [Billet et al. 04]
- * 2007: attack wb-DES [Goubin et al 07, Wyseur et al. 07]]
- * 2009: attack wb-AES [Michiels et al. 09]
- * 2010: attack Bringer et al. variant [De Mulder et al. 10]
- * 2012: attack Xiao-Lai variant [De Mulder et al. 12]
- * 2013: attack improvements and Karroumi variant [Lepoint et al.13]
- * ...

» Obscurity as a Solution

- * All public designs are broken
- * No provably secure solution

- * Growing demand in industry
- * Huge application potential



Security through obscurity: home-made design + obfuscation



Time consuming reverse engineering + structural analysis

» Differential Computation Analysis (DCA)

[Bos et al. 2016, Sanfelix et al. 2015]

Differential power analysis (DPA) techniques on computational leakages.

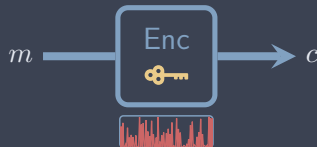
gray-box model



side-channel leakages (noisy)

e.g. power / EM / time / ...

white-box model

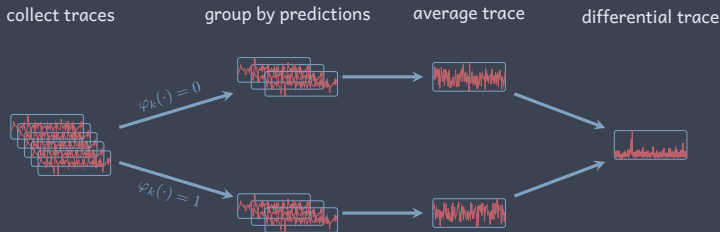


computational leakages (noisy-free)

e.g. registers / accessed memory / ...

» Differential Computation Analysis (DCA) (cont.)

[Bos et al. 2016, Sanfelix et al. 2015]



Implying strong *linear correlation* between the sensitive variables φ_k and the leaked samples in the computational traces.

Many publicly available implementations are broken by DCA.

» WhibOx Competitions

- * Organized as CHES CTF events

The competition gives an opportunity for researchers and practitioners to confront their (secretly designed) white-box implementations to state-of-the-art attackers

– WhibOx 2017

- * Designer: to submit the C source codes of AES-128 with secret key
- * Attacker: to reveal the hidden key
- * No need to disclose identity or underlying techniques

» WhibOx Competitions (cont.)

* WhibOx 2017

- * 94 submissions were **all broken** by 877 individual breaks
- * most (86%) of them were alive for < 1 day
- * mostly broken by DCA [Bock and Treff 20]

* WhibOx 2019

- * New rules encourage designers to submit “smaller” and “faster” implementations
 - * 27 submissions with 124 individual breaks
 - * 3 implementations survived, but broken after the competition
- * Both winning implementations due to Biryukov and Udovenko, and broken in this thesis with Goubin, Paillier and Rivain

» Thesis Contribution

- * Analyze in-depth why DCA can break internal encodings and propose new efficient attacks against internal encodings
- * Propose new different advanced gray-box attack paths
- * Analyze advanced gray-box countermeasures against new attack paths
- * Propose data-dependency analysis with substantially improved complexity over the existing attacks
- * Break the winning challenges from two editions of WhibOx competitions

» Publications

- [RW19] Rivain and Wang. “Analysis and Improvement of Differential Computation Attacks against Internally-Encoded White-Box Implementations”. In: **TCHES 2019 Issue 2**.
- [BRVW19] Bogdanov, Rivain, Vejre, and Wang. “Higher-Order DCA against Standard Side-Channel Countermeasures”. In: **COSADE 2019**.
- [GPRW20] Goubin, Paillier, Rivain, and Wang. “How to reveal the secrets of an obscure white-box implementation”. In: **Journal of Cryptographic Engineering Volume 10 Issue 1**.
- [GRW20] Goubin, Rivain, and Wang. “Defeating State-of-the-Art White-Box Countermeasures with Advanced Gray-Box Attacks”. In: **TCHES 2020 Issue 3**.

» Passive Gray-Box Adversary Model

[GRW20]

- * For each of N chosen $(x^{(i)})_{1 \leq i \leq N}$, collect a computational trace of t samples

$$\mathbf{v} = (v_1, v_2, \dots, v_t)$$

- * Build a distinguisher D :

$$(\gamma_k)_{k \in \mathcal{K}} = D \left((x^{(i)})_i, (\mathbf{v}^{(i)})_i \right)$$

- * Choose key candidate: $\operatorname{argmax}_{k \in \mathcal{K}} \gamma_k$

- * Number samples in attacked trace (window): t

- * Required number traces: N

» Outline

DCA Analysis and Improvements against Internal Encodings

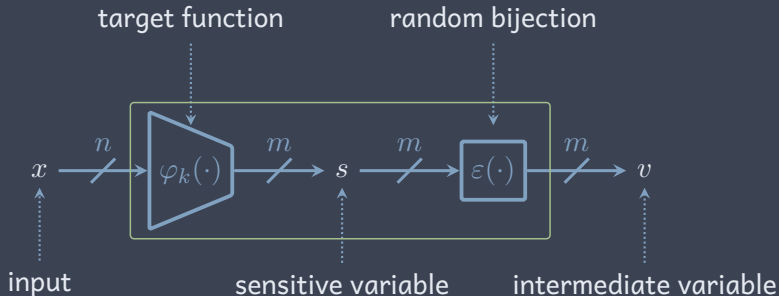
Advanced Gray-Box Countermeasures and Attacks

Data-Dependency Analysis

DCA Analysis and Improvements against Internal Encodings

- * DCA against Internal Encodings
- * Collision Attack

» Internal Encoding Leakage



- * $\varepsilon \circ \varphi_k$, as a result of some **table look-ups**, is **leaked in the memory**
- * To exploit the leakage of $\varepsilon \circ \varphi_k$, it is necessary that $n > m$

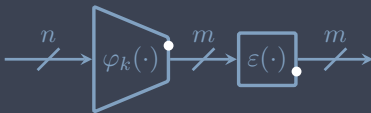
DCA Analysis and Improvements against Internal Encodings

- * DCA against Internal Encodings
- * Collision Attack

» DCA Analysis

Based on well-established theory – *Boolean correlation*, instead of *difference of means*: for any key guess k

$$\rho_k = \text{Cor}\left(\varphi_k(\cdot)[i] \ , \ \varepsilon \circ \varphi_{k^*}(\cdot)[j] \right)$$



DCA success (roughly) requires:

$$|\rho_{k^*}| > \max_{k^{\times}} |\rho_{k^{\times}}|$$

» Distributions of ρ_{k^*} and ρ_{k^\times}

- * Ideal assumption: $(\varphi_k)_k$ are mutually independent random (n, m) functions

Correct key guess k^* ,

$$\rho_{k^*} = 2^{2-m} N^* - 1$$

where

$$N^* \sim \mathcal{HG}(2^m, 2^{m-1}, 2^{m-1}) .$$

Only depends on m .

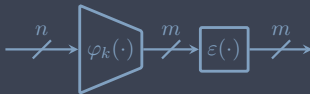
Incorrect key guess k^\times ,

$$\rho_{k^\times} = 2^{2-n} N^\times - 1$$

where

$$N^\times \sim \mathcal{HG}(2^n, 2^{n-1}, 2^{n-1}) .$$

Only depends on n .



» Lemma

Lemma

Let $\mathcal{B}(n)$ be the set of balanced n -bit Boolean functions. If $f \in \mathcal{B}(n)$ and $g \overset{\$}{\leftarrow} \mathcal{B}(n)$ independent of f , then the balanceness of $f + g$ is $B(f + g) = 4 \cdot N - 2^n$ where $N \sim \mathcal{HG}(2^n, 2^{n-1}, 2^{n-1})$ denotes the size of $\{x : f(x) = g(x) = 0\}$.

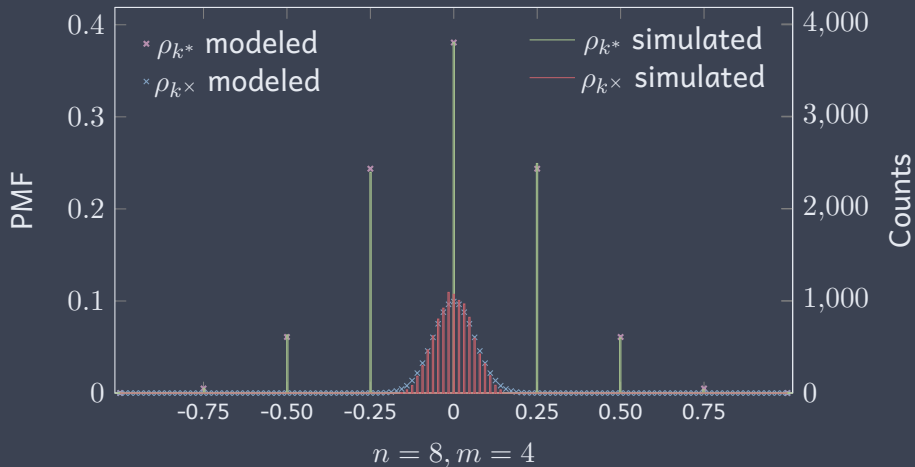
With

$$\text{Cor}(f + g) = \frac{1}{2^n} B(f + g)$$

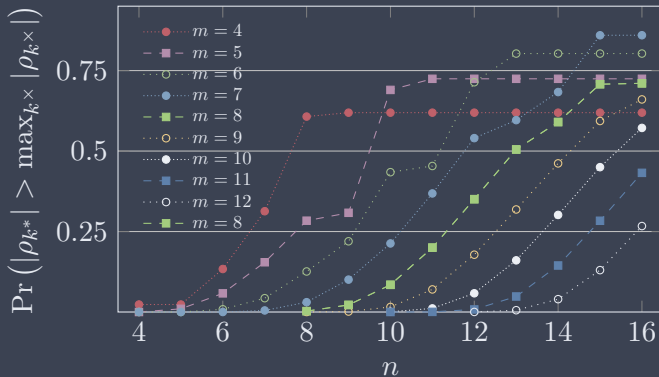
 \Rightarrow

$$\rho_{k^*} = 2^{2-m} N^* - 1 \quad \text{and} \quad \rho_{k^\times} = 2^{2-n} N^\times - 1$$

where $N^* \sim \mathcal{HG}(2^m, 2^{m-1}, 2^{m-1})$ and $N^\times \sim \mathcal{HG}(2^n, 2^{n-1}, 2^{n-1})$.

» Distributions of ρ_{k^*} and ρ_{k^\times} 

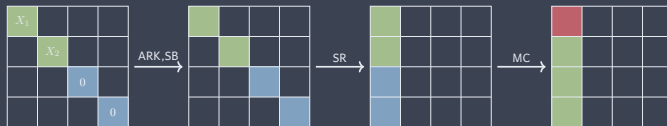
» DCA Success Rate: $|\rho_{k^*}| > \max_{k \neq k^*} |\rho_k|$



DCA success probability converges towards $\approx 1 - \Pr_{N^*}(2^{m-2})$ for $n \geq 2m + 2$.

» Attack a NSC Variant: a White-Box AES

- * Byte encoding protected AES
- * DCA has failed to break it *before this work*
- * Our approach: target an output byte of MixColumn in the first round



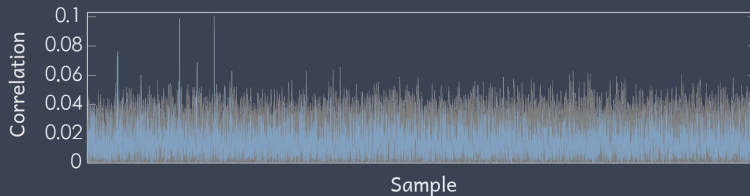
$$\varphi_{k_1||k_2}(x_1||x_2) = 2 \cdot \mathbf{Sbox}(x_1 \oplus k_1) \oplus 3 \cdot \mathbf{Sbox}(x_2 \oplus k_2) \oplus \mathbf{Sbox}(k_3) \oplus \mathbf{Sbox}(k_4) \oplus c$$

$$\varepsilon' = \varepsilon \circ \oplus_c ,$$

$$n = 16, m = 8 , |\mathcal{K}| = 2^{16}.$$

» Attack a NSC Variant: a White-Box AES

- * Attack results: ~ 1800 traces



- * Byte encoding can be efficiently broken

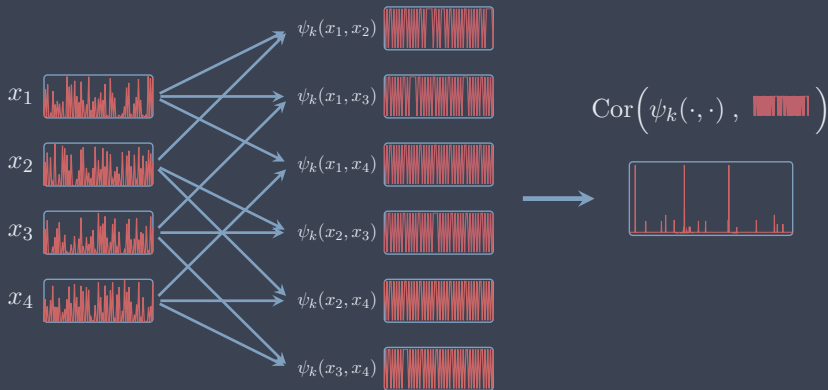
DCA Analysis and Improvements against Internal Encodings

- * DCA against Internal Encodings
- * Collision Attack

» Collision Attack

N inputs & raw traces

$\binom{N}{2}$ collision predictions & traces



$$\psi_k(x_1, x_2) := \left(\varphi_k(x_1) = \varphi_k(x_2) \right)$$

» Collision Attack Explanation

Based on the principle:

$$\varphi_k(x_1) = \varphi_k(x_2) \Leftrightarrow \varepsilon \circ \varphi_k(x_1) = \varepsilon \circ \varphi_k(x_2)$$

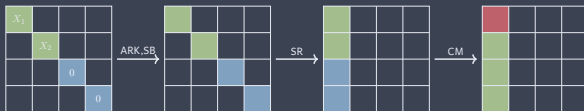
Trace Complexity:

$$N = \mathcal{O}\left(2^{\frac{m}{2}}\right)$$

		Predictions					
		1	2	3	4	5	6
key guesses	k_1	○○○○	●○○○	●●○○	○○○○	●●○○	○○○○
	k_2	●●○○	○○○○	○○○○	●○○○	○○○○	●●○○
	k_3	●○○○	●○○○	●○○○	●○○○	○○○○	●○○○
	k_4	○○○○	●●○○	●●○○	○○○○	●○○○	○○○○

» Attack the NSC Variant

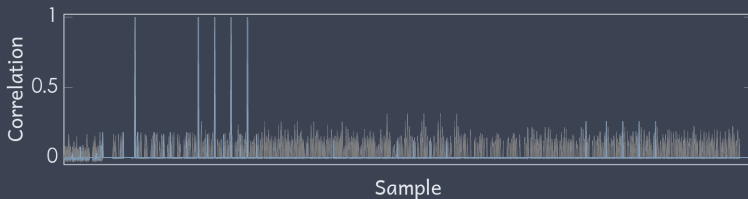
- * Same to DCA: targeting at one 1-st round MixColumn output byte



$$\varphi_{k_1||k_2}(x_1||x_2) = 2 \cdot \mathbf{Sbox}(x_1 \oplus k_1) \oplus 3 \cdot \mathbf{Sbox}(x_2 \oplus k_2)$$

$$\varepsilon' = \varepsilon \circ \oplus_c$$

- * Attack results: 60 traces



» Contribution Summary

- * DCA against internal encodings has been analysed in depth
 - * Allows to attack wider encodings
- * Propose new class of collision attacks with very low trace complexity
- * Mutual information analysis with similar trace complexity but higher computation complexity
- * Hence, protecting AES with internal encodings in the beginning rounds is insufficient

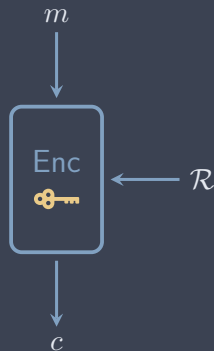
Advanced Gray-Box Countermeasures and Attacks

- * Linear Masking, Higher-Order DCA, and Linear Decoding Analysis
- * Algebraic Security and Non-Linear Masking
- * Shuffling

» Random Source

[BRVW19]

- * Countermeasures need randomness.



» Random Source

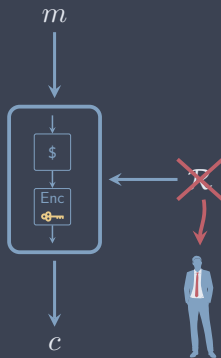
[BRVW19]

- * Countermeasures need randomness.
- * Plaintext is the only source of randomness
- * Security criteria:

Pseudorandomness no statistical flaw

Obscurity the design should be kept secret

Obfuscation hard to distinguish from other intermediate variables



Advanced Gray-Box Countermeasures and Attacks

- * Linear Masking, Higher-Order DCA, and Linear Decoding Analysis
- * Algebraic Security and Non-Linear Masking
- * Shuffling

» Linear Masking

[Ishai et al. 03]

- * Intermediate value x is split into n shares

$$x = x_1 \oplus x_2 \cdots \oplus x_n$$

original states



Masking



masked states

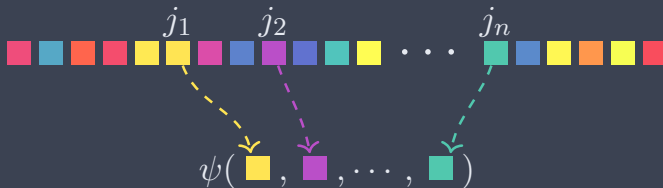


- * Shares are manipulated separately such that any subset of at most $n - 1$ shares is independent of x
- * Resistant against $(n - 1)$ -th order DCA attacks

» Higher-Order DCA (HO-DCA)

[BVRW19]

- * Trace **pre-processing**: an n -th order trace contains $q = \binom{t}{n}$ points:



- * The natural combination function ψ is XOR sum
- * Perform DCA attacks on the higher-order traces
- * Linear masking can be broken
 - * \exists fixed n positions in which the shares are

$$\binom{1000}{5} \approx 2^{43}$$

» Linear Decoding Analysis (LDA)

[GPRW20]

- * Assumption: there exists a linear decoding function

$$D(v_1, v_2, \dots, v_t) = a_0 \oplus \left(\bigoplus_{1 \leq i \leq t} a_i \cdot v_i \right) = \varphi_k(x)$$

for some sensitive variable φ_k and some fixed coefficients a_0, a_1, \dots, a_t .

- * Record the v_i 's over N executions:

$$\begin{bmatrix} 1 & v_1^{(1)} & \cdots & v_t^{(1)} \\ 1 & v_1^{(2)} & \cdots & v_t^{(2)} \\ 1 & \vdots & \ddots & \vdots \\ 1 & v_1^{(N)} & \cdots & v_t^{(N)} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \varphi_k(x^{(1)}) \\ \varphi_k(x^{(2)}) \\ \vdots \\ \varphi_k(x^{(N)}) \end{bmatrix}$$

» Linear Decoding Analysis (LDA) (cont.)

[GPRW20]

- * Record the v_i 's over N executions:

$$\begin{bmatrix} 1 & v_1^{(1)} & \cdots & v_t^{(1)} \\ 1 & v_1^{(2)} & \cdots & v_t^{(2)} \\ 1 & \vdots & \ddots & \vdots \\ 1 & v_1^{(N)} & \cdots & v_t^{(N)} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \varphi_k(x^{(1)}) \\ \varphi_k(x^{(2)}) \\ \vdots \\ \varphi_k(x^{(N)}) \end{bmatrix}$$

- * Linear masking is vulnerable to LDA

- * system solvable for k^*
- * but not for incorrect key guess k^\times

- * Trace Complexity $t + \mathcal{O}(1)$

- * Computation complexity $\mathcal{O}(t^{2.8} \cdot |\mathcal{K}|)$

$$1000^{2.8} \approx 2^{28}$$

» Breaking Whib0x 2017 Winning Challenge with LDA

[GPRW20]

- * Small windows located for target variables
- * The 3rd s-box $t = 35, \min(n) = 2 \implies$ HO-DCA: 2^{18} , and LDA: 2^{22}

Bit	Encoding coefficients
1	0000001010111000101010000000000000
2	0000001001101111111000000000000000
3	0000000010100011110111000000000000
4	0000000001100011101110000000000000
5	0000000110010000001110000000000000
6	0000000000100000000001000000000000
7	0000001000100101010100000000000000
8	0000000100001001100000000000000000

- * The 14th s-box: $t = 45, \min(n) = 9 \implies$ HO-DCA: 2^{49} , and LDA: 2^{23}

Advanced Gray-Box Countermeasures and Attacks

- * Linear Masking, Higher-Order DCA, and Linear Decoding Analysis
- * Algebraic Security and Non-Linear Masking
- * Shuffling

» Algebraic Security and Non-Linear Masking

[Biryukov and Udovenko 18]

- * Introduced by Biryukov and Udovenko at Asiacrypt 2018
- * To capture LDA like algebraic attack

A d -th degree algebraically-secure non-linear masking ensures that any function of up to d degree to the intermediate variables should not compute a “predictable” variable.

» First-Degree Secure Non-Linear Masking

[Biryukov and Udovenko 18]

- * Quadratic decoding function

$$(a, b, c) \mapsto ab \oplus c$$

- * Secure gadgets for bit XOR, bit AND, and refresh
- * Provably secure composition
- * But vulnerable to DCA attack

$$\text{Cor}(ab \oplus c, c) = \frac{1}{2}$$

- * Empirically, suggest using a combination of linear masking and non-linear masking to thwart both DCA (probing security) and LDA (algebraic security).

» Combination of Linear Masking and Non-linear Masking

[GRW20]

* Three possible natural combinations:

1. apply linear masking on top of non-linear masking

$$x = (a_1 \oplus a_2 \oplus \cdots \oplus a_n)(b_1 \oplus b_2 \oplus \cdots \oplus b_n) \oplus (c_1 \oplus c_2 \oplus \cdots \oplus c_n)$$

2. apply non-linear masking on top of linear masking

$$x = (a_1 b_1 \oplus c_1) \oplus (a_2 b_2 \oplus c_2) \oplus \cdots \oplus (a_n b_n \oplus c_n) .$$

3. merge the two maskings into a new encoding

$$x = ab \oplus c_1 \oplus c_2 \oplus \cdots \oplus c_n .$$

* For first two combinations, the combined masking gadgets can be simply derived from the original gadgets of both schemes.

» Higher-Degree Decoding Analysis (HDDA)

[GPRW20]

- * Assume the decoding function is of degree d
- * Trace **pre-processing**: a d -th degree trace contains all monomials of degree $\leq d$



- * Perform LDA attacks on the higher-degree traces
- * Higher-degree trace samples: $\sum_{i=0}^d \binom{t}{d} = \binom{t+d}{d} \ll t^d$
- * Complexity: $\mathcal{O}(t^{2.8d} \cdot |\mathcal{K}|)$, practical when t, d are small.

$$t^{2.8d} < 2^{50}$$



$$\begin{aligned} d = 2 &\Rightarrow t < 487 \\ t = 100 &\Rightarrow d \leq 5 \end{aligned}$$

Advanced Gray-Box Countermeasures and Attacks

- * Linear Masking, Higher-Order DCA, and Linear Decoding Analysis
- * Algebraic Security and Non-Linear Masking
- * **Shuffling**

» Shuffling

- * The order of execution is randomly chosen for each run of the implementation.
- * To increase noise in the adversary's observation

masked states



iteration in *normal* order



iteration in *randomized* order

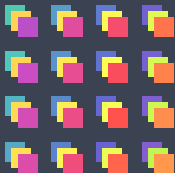


» Shuffling (cont.)

[BRVW19]

- * Not enough in white-box model: traces can be aligned by memory
- * Thus, the memory location of shares has to be shuffled.

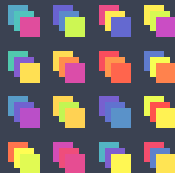
masked states



memory shuffling



memory shuffled states



» HO-DCA and Integrated HO-DCA against Masking and Shuffling

[BRVW19]

- * $\nexists n$ fixed locations for all shares
- * Shuffling degree is λ
 - * correlation score decreased by a factor of λ
 - * attack slow down by a factor of λ^2
- * Integrate values from all λ slots
 - * correlation score decreased by a factor of $\sqrt{\lambda}$
 - * attack slow down by a factor of λ

» Multivariate HO-DCA

[BRVW19]

- * The multivariate HO-DCA optimizes the attack by exploiting joint information of the higher-order samples on the secrets
- * Based on a maximum likelihood distinguisher

$$\gamma_k = \Pr(K = k | (V^{(i)})_i = (v^{(i)})_i \wedge (X^{(i)})_i = (x^{(i)})_i)$$

- * We show that

$$\gamma_k \propto \prod_{i=1}^N C_k(v^{(i)}, x^{(i)})$$

where *the counter*

$C_k(v, x) :=$ the number of n -tuples *s.t.* $v_{j_1} \oplus \cdots \oplus v_{j_n} = \varphi_k(x)$ in one trace.

» Attack Comparison

	linear masking		linear + NL masking	
	trace	computation	trace	computation
<i>without shuffling</i>				
LDA / HDDA	$t + \mathcal{O}(1)$	$\mathcal{O}(\mathcal{K} \cdot t^{2.8})$	$\mathcal{O}(t^2)$	$\mathcal{O}(\mathcal{K} \cdot t^{5.6})$
HODCA	c	$\mathcal{O}(\mathcal{K} \cdot t^n)$	$4c$	$\mathcal{O}(\mathcal{K} \cdot t^n)$
<i>with shuffling of degree λ</i>				
HO-DCA	$c \lambda^2$	$\mathcal{O}(\mathcal{K} \cdot t^n \cdot \lambda^2)$	$4c \lambda^2$	$\mathcal{O}(\mathcal{K} \cdot t^n \cdot \lambda^2)$
Intg. HO-DCA	$c \lambda$	$\mathcal{O}(\mathcal{K} \cdot t^n \cdot \lambda)$	$4c \lambda$	$\mathcal{O}(\mathcal{K} \cdot t^n \cdot \lambda)$
MV HO-DCA	$\mathcal{O}(t^n)$	$\mathcal{O}(\mathcal{K} \cdot t^{2n})$		

Note that c is some small empirical factor

Data-Dependency Analysis

- * Data-Dependency Graph
- * Data-Dependency Analysis against Masking Combinations

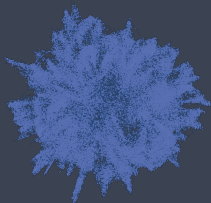
Data-Dependency Analysis

- * Data-Dependency Graph
- * Data-Dependency Analysis against Masking Combinations

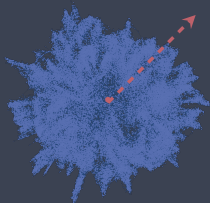
» Data Dependency Graph

[GPRW20]

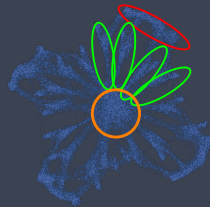
- * White-box adversary also observes data-flow.
- * Data-dependency graph (DDG) can visually reveal the structure of the implementation.



first 20%



first 10%

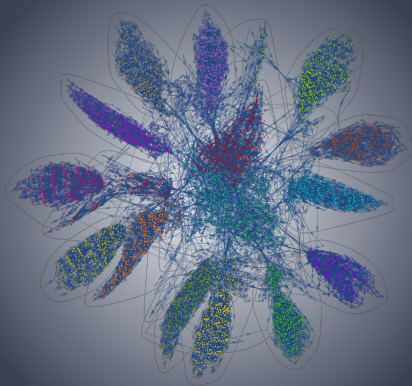


first 5%

» Data Dependency Clusters

[GPRW20]

- * Data-dependency can extract computation clusters and determines the location of sensitive computation
 - * Round: ~ 28 k nodes
 - * S-box cluster: ~ 500 nodes
 - * Trace windows containing targets: ~ 50 nodes



Data-Dependency Analysis

- * Data-Dependency Graph
- * **Data-Dependency Analysis against Masking Combinations**

» **AND Gadget for Linear Masking**

[Ishai et al. 03]

$$(x_1, x_2, \dots, x_n), (y_1, y_2, \dots, y_n) \mapsto (z_1, z_2, \dots, z_n) \quad \text{s.t.} \quad \bigoplus_i x_i \cdot \bigoplus_i y_i = \bigoplus_i z_i.$$

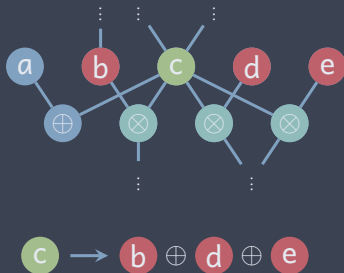
$$\begin{bmatrix} x_1 y_1 & 0 & 0 \\ x_1 y_2 & x_2 y_2 & 0 \\ x_1 y_3 & x_2 y_3 & x_3 y_3 \end{bmatrix} \oplus \begin{bmatrix} 0 & x_2 y_1 & x_3 y_1 \\ 0 & 0 & x_3 y_2 \\ 0 & 0 & 0 \end{bmatrix}^T \oplus \begin{bmatrix} 0 & r_{1,2} & r_{1,3} \\ r_{1,2} & 0 & r_{2,3} \\ r_{1,3} & r_{2,3} & 0 \end{bmatrix} \xrightarrow{\text{sum rows}} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}$$

Each x_i is multiplied with all shares of y : $(y_j)_j$, vice versa.

» Data-Dependency Analysis against Masking Combinations

[GRW20]

- * Find co-operands of each node for \otimes
- * Collecting data-dependency (DD) traces
 - * Sum co-operands values
- * Launch HO-DCA attacks on DD traces
 - * Biased variables can be covered in DD trace
- * Computation complexity substantially improved
- * Successfully applied to break WhibOx 2019 winning implementations



» Attack Comparison

	linear masking		linear + NL masking	
	trace	computation	trace	computation
<i>without shuffling</i>				
LDA/HDDA	$t + \mathcal{O}(1)$	$\mathcal{O}(\mathcal{K} \cdot t^{2.8})$	$\mathcal{O}(t^2)$	$\mathcal{O}(\mathcal{K} \cdot t^{5.6})$
HODCA	c	$\mathcal{O}(\mathcal{K} \cdot t^n)$	$4c$	$\mathcal{O}(\mathcal{K} \cdot t^n)$
DD-DCA	c	$\mathcal{O}(\mathcal{K} \cdot t)$	$4c$	$\mathcal{O}(\mathcal{K} \cdot t)$
<i>with shuffling of degree λ</i>				
HO-DCA	$c\lambda^2$	$\mathcal{O}(\mathcal{K} \cdot t^n \cdot \lambda^2)$	$4c\lambda^2$	$\mathcal{O}(\mathcal{K} \cdot t^n \cdot \lambda^2)$
Intg. HO-DCA	$c\lambda$	$\mathcal{O}(\mathcal{K} \cdot t^n \cdot \lambda)$	$4c\lambda$	$\mathcal{O}(\mathcal{K} \cdot t^n \cdot \lambda)$
MV HO-DCA	$\mathcal{O}(t^n)$	$\mathcal{O}(\mathcal{K} \cdot t^{2n})$		
DD-DCA	$c\lambda^2$	$\mathcal{O}(\mathcal{K} \cdot t \cdot \lambda^2)$	$4c\lambda^2$	$\mathcal{O}(\mathcal{K} \cdot t \cdot \lambda^2)$
Intg. DD-DCA	$c\lambda$	$\mathcal{O}(\mathcal{K} \cdot t \cdot \lambda)$	4λ	$\mathcal{O}(\mathcal{K} \cdot t \cdot \lambda)$

Note that c is some small empirical factor

» Thesis Summary

- * This thesis concentrates on practical security of white-box cryptography
 - * Demonstrated the capabilities of gray-box adversary in white-box model
 - * Understood why gray-box attacks work against internal encodings
 - * Proposed new gray-box attacks
 - * Quantified different gray-box attack performance against different countermeasures
- * A good level of practical resistance against these attacks can be achieved
 - * under an assumption of adversary's uncertainty on attacked window within a full computation trace
 - * for some choice of the parameters for countermeasures
 - * Stressed the importance to hide structural knowledge of implementation

» Future Research Perspectives

- * To ensure the uncertainty assumption on the attacked trace window
 - * to counter data-dependency leakage
 - * circuit obfuscation
- * To build formal security arguments in passive gray-box attack model, *e.g.* to show that
 - * the proposed attacks are somehow optimal
 - * and the best the adversary can do can be made arbitrarily hard
- * To construct higher-degree algebraically-secure gadgets

On the Practical Security of White-Box Cryptography

Thesis Defense

by Junwei Wang (王军委)

on June 24, 2020

Supervisors Jean-Sébastien Coron, Sihem Mesnager

Advisors Pascal Paillier, Matthieu Rivain

