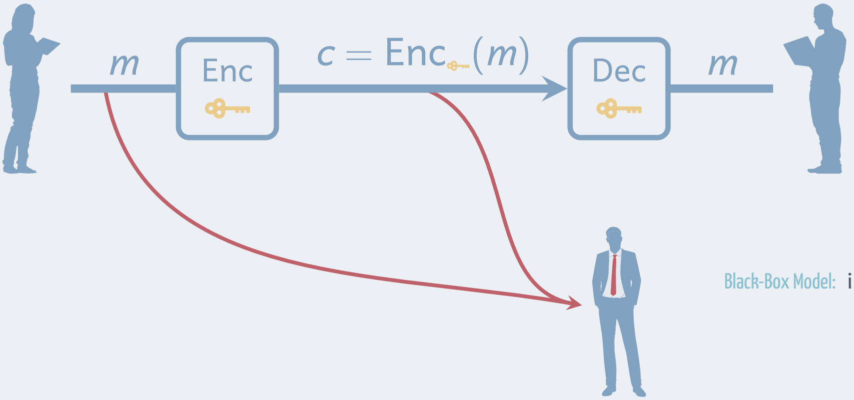
Defeating State-of-the-Art White-Box Countermeasures with Advanced Gray-Box Attacks

Louis Goubin[4] **Matthieu Rivain**[1] **Junwei Wang (王军委)**[1,2,3]

[1]CryptoExperts [2]University of Luxembourg [3]University Paris 8 [4]UVSQ

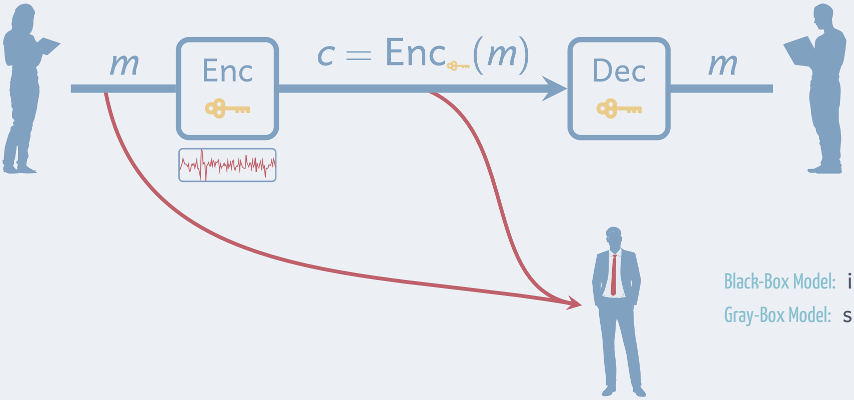
Prerecorded talk for **CHES 2020**, September 2020

» Security Models: Shades of Gray



Black-Box Model: input/output behavior

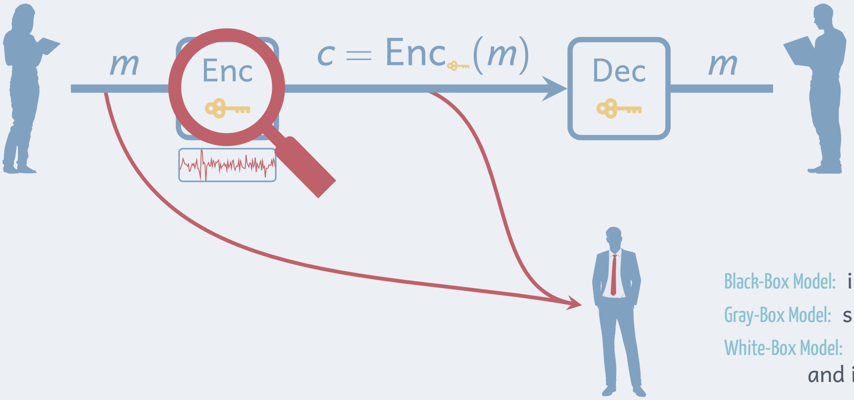
» Security Models: Shades of Gray



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 of impl. and its execution environment

» White-Box Threat Model

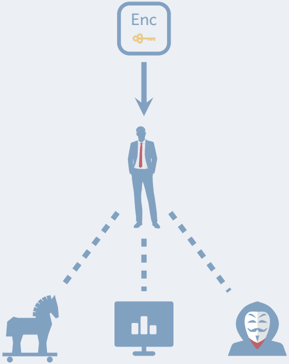
To extract a cryptographic key

Where from a software implementation of cipher

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

How by all kinds of means

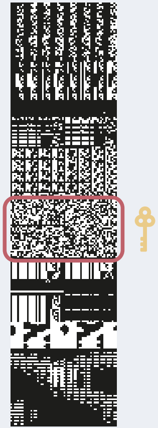
- analyze the code
- spy on the memory
- interfere the execution
- cut external randomness
- . . .



» 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]

» Security through Obscurity

* All public white-box designs broken
* No provably secure solution

* Growing demand in industry
* Huge application potential

$\Downarrow$

Security through obscurity: home-made design + obfuscation

Time consuming reverse engineering + structural analysis

» Differential Computation Analysis (DCA) [BHMT16]

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 / ···

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 [BT20]
* 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 in this article

» Outline

Advanced Gray-Box Countermeasures and Attacks

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

» Linear Masking [ISW03]

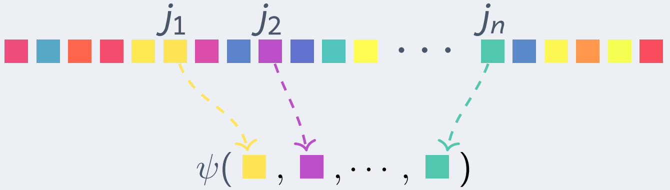
* Intermediate value x is split into n shares

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

* 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 (affine) decoding function

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

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

* 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:

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

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

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

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

[BU18]

- 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 [BU18]

* 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

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

* They 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

We suggest 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

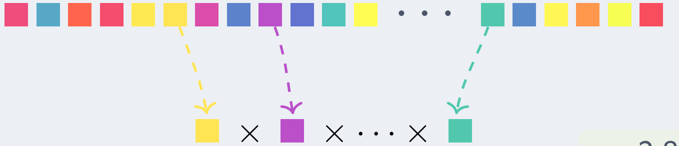
$$x = (a_1 b_1 \oplus c_1) \oplus (a_2 b_2 \oplus c_2) \oplus \cdots \oplus (a_n b_n \oplus c_n) \, .$$

3. merge the two maskings into a new encoding

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

» 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: $\sum_{i=0}^{d} \binom{t}{i} = \binom{t+d}{d} \ll t^d$
- Complexity: $\mathcal{O}\left(t^{2.8d} \cdot |\mathcal{K}|\right)$, practical when t, d are small.

$$t^{2.8d} < 2^{50}$$
$$\Downarrow$$
$$d = 2 \quad \Rightarrow \quad t < 487$$
$$d = 3 \quad \Rightarrow \quad t < 62$$

Advanced Gray-Box Countermeasures and Attacks

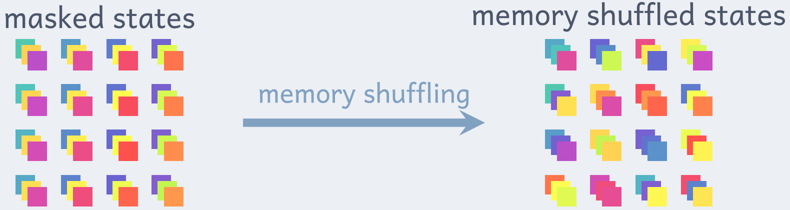
* Linear Masking, Higher-Order DCA, and Linear Decoding Analysis
* Algebraic Security and Non-Linear Masking
* Shuffling

» Shuffling

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

» 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.



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

	shuffling degree λ	
	correlation decrease	attack slowdown
HODCA	λ	λ^2
Integrated HODCA	$\sqrt{\lambda}$	λ

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

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

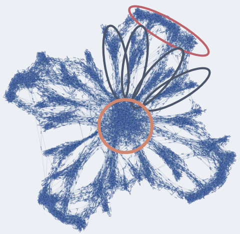


Illustration from [GPRW20]

Data-Dependency Analysis

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

» Linear Masking Gadget for AND [ISW03]

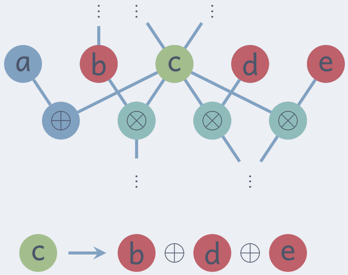
$$(x_1, x_2, \cdots, x_n), (y_1, y_2, \cdots, y_n) \mapsto (z_1, z_2, \cdots, z_n) \quad \text{s.t.} \bigoplus_i x_i \cdot \bigoplus_i y_i = \bigoplus_i z_i \,.$$

$$\begin{bmatrix} x_1y_1 & 0 & 0 \\ x_1y_2 & x_2y_2 & 0 \\ x_1y_3 & x_2y_3 & x_3y_3 \end{bmatrix} \oplus \begin{bmatrix} 0 & x_2y_1 & x_3y_1 \\ 0 & 0 & x_3y_2 \\ 0 & 0 & 0 \end{bmatrix}^T \oplus \begin{bmatrix} 0 & r_{1,2} & r_{1,3} \\ r_{1,2} & 0 & r_{2,3} \\ r_{1,3} & r_{2,3} & 0 \end{bmatrix} \xrightarrow{\text{sum rows}} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}$$

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

» Data-Dependency Analysis against Masking Combinations

- 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 recovered 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
without shuffling				
LDA/HDDA	$t + \mathcal{O}(1)$	$\mathcal{O}\left(\lvert\mathcal{K}\rvert \cdot t^{2.8}\right)$	$\mathcal{O}\left(t^2\right)$	$\mathcal{O}\left(\lvert\mathcal{K}\rvert \cdot t^{5.6}\right)$
HODCA	c	$\mathcal{O}(\lvert\mathcal{K}\rvert \cdot t^n)$	$4\,c$	$\mathcal{O}(\lvert\mathcal{K}\rvert \cdot t^n)$
DD-DCA	c	$\mathcal{O}(\lvert\mathcal{K}\rvert \cdot t)$	$4\,c$	$\mathcal{O}(\lvert\mathcal{K}\rvert \cdot t)$
with shuffling of degree λ				
HO-DCA	$c\,\lambda^2$	$\mathcal{O}\left(\lvert\mathcal{K}\rvert \cdot t^n \cdot \lambda^2\right)$	$4\,c\,\lambda^2$	$\mathcal{O}\left(\lvert\mathcal{K}\rvert \cdot t^n \cdot \lambda^2\right)$
Intg. HO-DCA	$c\,\lambda$	$\mathcal{O}(\lvert\mathcal{K}\rvert \cdot t^n \cdot \lambda)$	$4\,c\,\lambda$	$\mathcal{O}(\lvert\mathcal{K}\rvert \cdot t^n \cdot \lambda)$
DD-DCA	$c\,\lambda^2$	$\mathcal{O}\left(\lvert\mathcal{K}\rvert \cdot t \cdot \lambda^2\right)$	$4\,c\,\lambda^2$	$\mathcal{O}\left(\lvert\mathcal{K}\rvert \cdot t \cdot \lambda^2\right)$
Intg. DD-DCA	$c\,\lambda$	$\mathcal{O}(\lvert\mathcal{K}\rvert \cdot t \cdot \lambda)$	$4\ \ \lambda$	$\mathcal{O}(\lvert\mathcal{K}\rvert \cdot t \cdot \lambda)$

Note that c is some small empirical factor

Conclusion

» Conclusion

* Revisited state-of-the-art countermeasures employed in practice
  * Linear masking, non-linear masking, shuffling and how to combine them
* Quantified different (advanced) gray-box attack performance against different countermeasures
  * (Higher-order) DCA, (higher-degree) Decoding Analysis, · · ·
* Proposed new attacks based on data-dependency with substantial computation complexity improvement
* Broke three WhibOx 2019 winning challenges

paper ia.cr/2020/413

attack **CryptoExperts** / breaking-winning-challenges-of-whibox2019