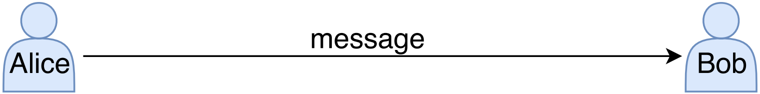
Design and Cryptanalysis of Symmetric-Key Algorithms in Black and White-box Models

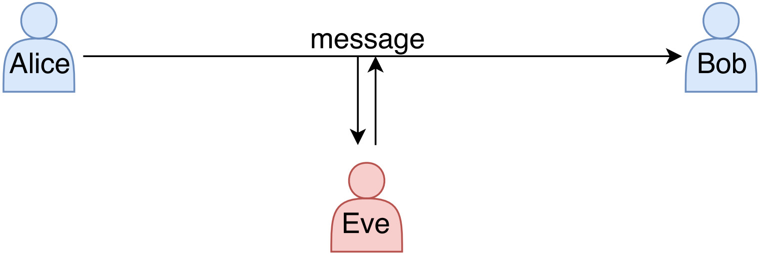
Aleksei Udovenko

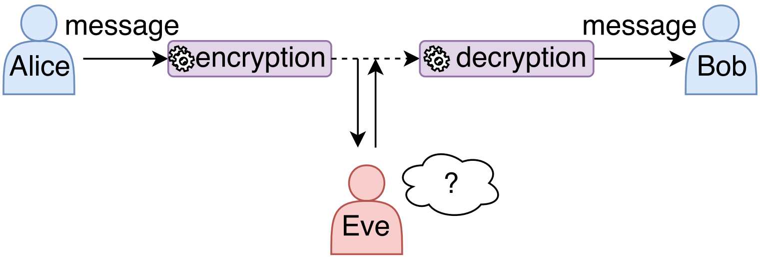
SnT, University of Luxembourg

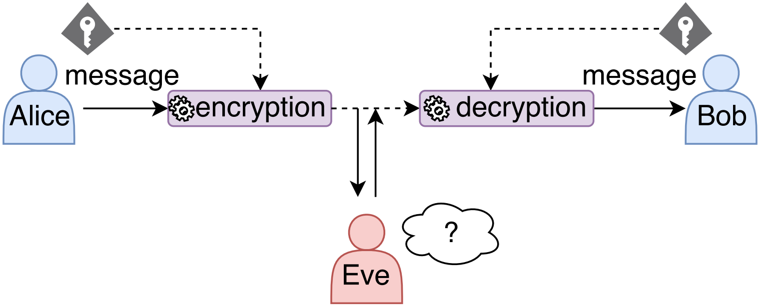
April 9, 2019
PhD Defense

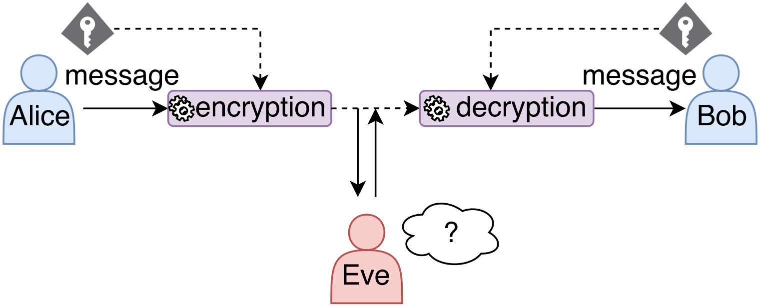

message
Alice
Bob


Alice
message
Bob

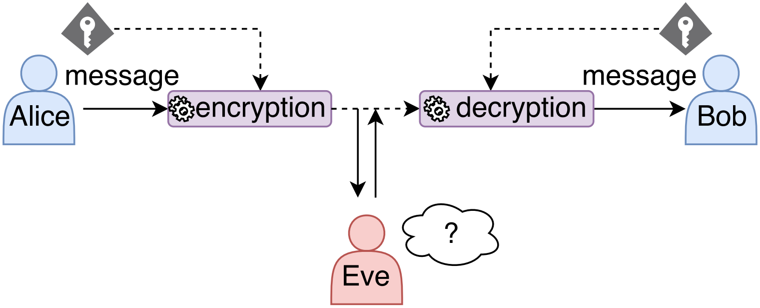

Alice
message
Bob
Eve


message
Alice
encryption
decryption
message
Bob
Eve
?


message
Alice
encryption
decryption
message
Bob
Eve
?



Symmetric-key Cryptography



Symmetric-key Cryptography

ensures that the message is:

1. secret (confidentiality)
2. unmodified (integrity)
3. from the correct person (authenticity)

$$\left.\begin{array}{l}\text{(confidentiality)}\\ \text{(integrity)}\\ \text{(authenticity)}\end{array}\right\}$$

$$\left.\begin{array}{l}\text{(confidentiality)}\\ \text{(integrity)}\\ \text{(authenticity)}\end{array}\right\} \quad \Longleftrightarrow \quad \begin{array}{c}\text{Authenticated}\\ \text{Encryption}\end{array}$$

(confidentiality)
(integrity)
(authenticity)

$\Bigg\} \quad \Leftrightarrow$ Authenticated Encryption

The main goal of symmetric-key cryptography!

How does it work?

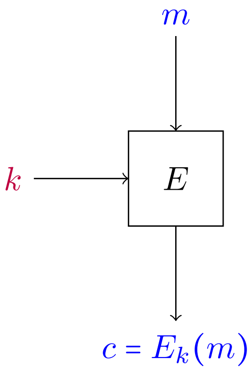
Construction 1:

Block Cipher + Mode of Operation

Block Cipher

An *Algorithm* E_k:

- n-bit message m
- κ-bit key k
- n-bit ciphertext c
- E_k is invertible

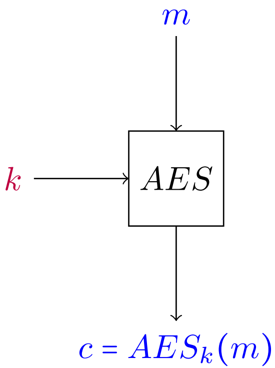


Example: Advanced Encryption Standard

AES Algorithm:

- 128-bit message m
- 128/192/256-bit key k
- 128-bit ciphertext c

- designed in 1998
  by V. Rijmen and J. Daemen



Mode of Operation

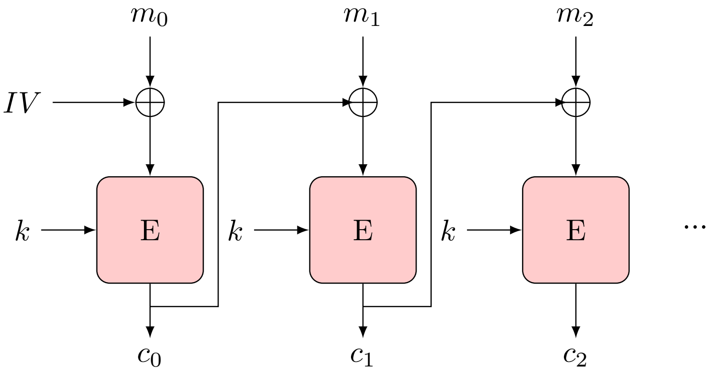


Figure credits: TikZ for Cryptographers

Example: COLM Mode of Operation

One of CAESAR competition winners (2019)

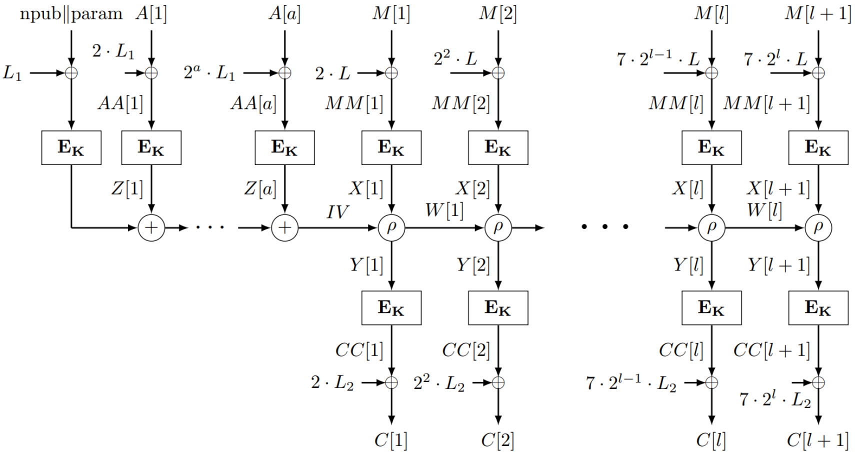
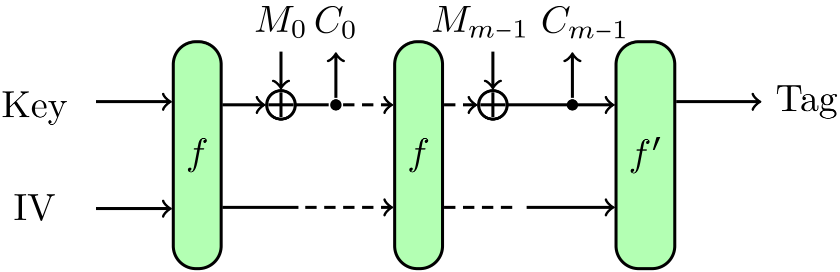


Figure credits: COLM v1 Specification

Construction 2:
Sponge Structure

(Duplexed) Sponge Structure



f: *keyless* invertible function (permutation)

Figure credits: TikZ for Cryptographers

Plan

1. Introduction

2. Thesis Overview
   - Design of Symmetric-key Algorithms
   - Structural and Decomposition Cryptanalysis
   - Nonlinear Invariant Cryptanalysis
   - White-box Cryptography

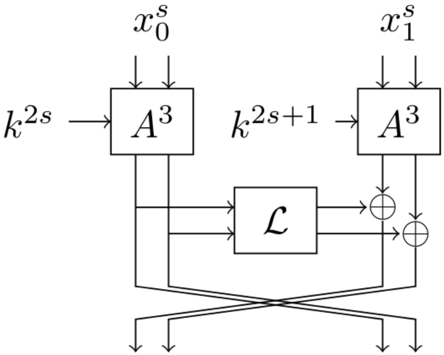
3. White-box Cryptography

Design of Symmetric-key Algorithms

Lightweight Cryptography:

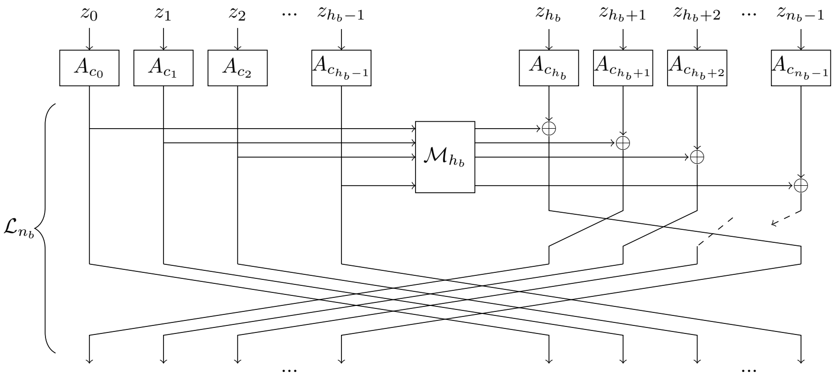
Cryptography for resource-constrained devices
(Internet of Things)

Design of Symmetric-key Algorithms



Sparx: a *lightweight* block cipher
based on a new design strategy

Design of Symmetric-key Algorithms



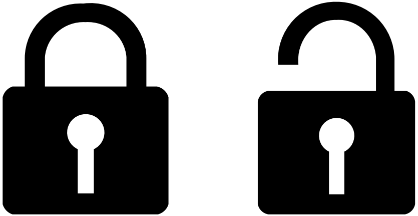
Sparkle, **Esch** and **Schwaemm**:
cryptographic permutations, hash functions
and authenticated encryption

Design of Symmetric-key Algorithms

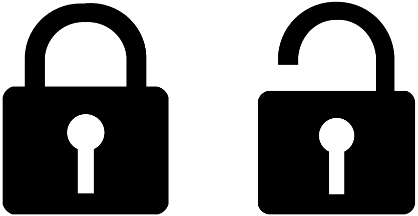
Daniel Dinu, Léo Perrin, Aleksei Udovenko, Vesselin Velichkov, Johann Großschädl, and Alex Biryukov.
Design Strategies for ARX with Provable Bounds: Sparx and LAX.
In *Advances in Cryptology - ASIACRYPT 2016*, pages 484–513.
https://www.cryptolux.org/index.php/SPARX.

Christof Beierle, Alex Biryukov, Luan Cardoso dos Santos, Johann Großschädl, Léo Perrin, Aleksei Udovenko, Vesselin Velichkov, and Qingju Wang.
Schwaemm and Esch: Lightweight Authenticated Encryption and Hashing using the Sparkle Permutation Family, 2019.
https://www.cryptolux.org/index.php/Sparkle.

How to make sure
that an encryption scheme is secure?

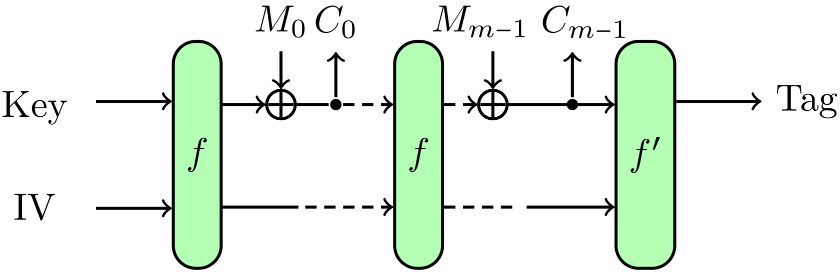


How to make sure
that an encryption scheme is secure?

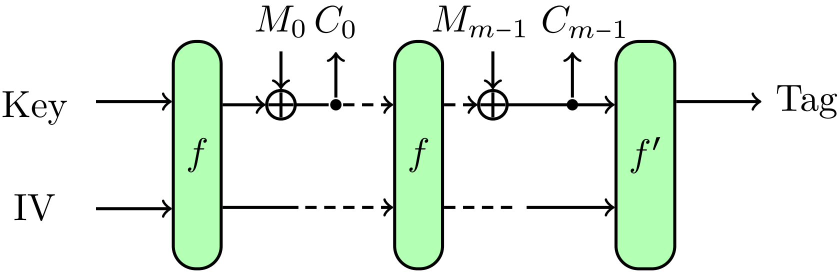


Security Proofs and Cryptanalysis!

Security Proofs: Modes and Structures



Security Proofs: Modes and Structures



secure **if** the permutation f is secure (random)

Cryptanalysis:

an attempt to invalidate
security claims of a cryptosystem
by developing an attack

Cryptanalysis:

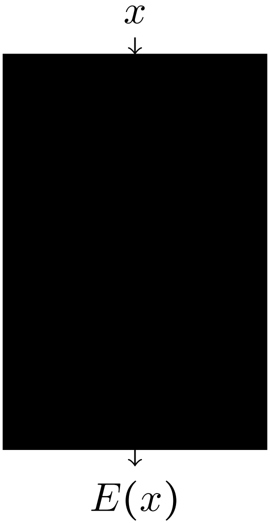
an attempt to invalidate
security claims of a cryptosystem
by developing an attack

- a large variety of methods: differential, linear, integral, ...
- attacks on simplified versions
- analysis of components

Structural and Decomposition Cryptanalysis

Distinguishing structures and recovering components

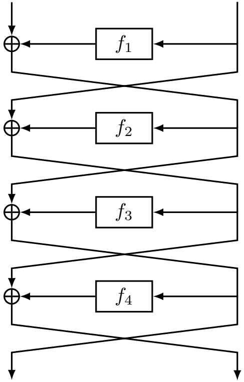
Structural and Decomposition Cryptanalysis



Structural and Decomposition Cryptanalysis

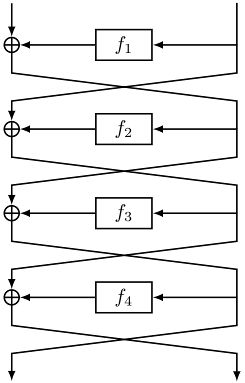
x	$E(x)$
0	182
1	210
2	78
3	251
4	97
...	
252	112
253	19
254	224
255	74

Structural and Decomposition Cryptanalysis



Feistel Networks

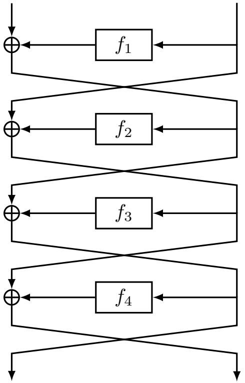
Structural and Decomposition Cryptanalysis



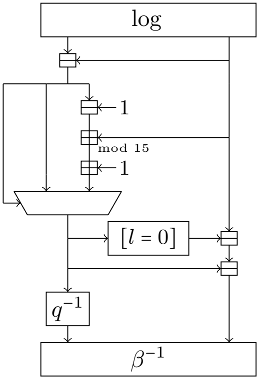
Feistel Networks

x	$E(x)$
0	182
1	210
2	78
3	251
4	97
...	
252	112
253	19
254	224
255	74

Structural and Decomposition Cryptanalysis

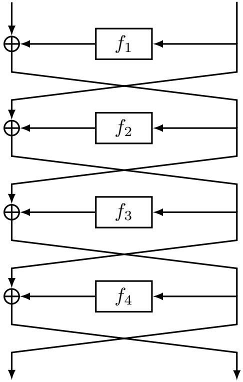


Feistel Networks



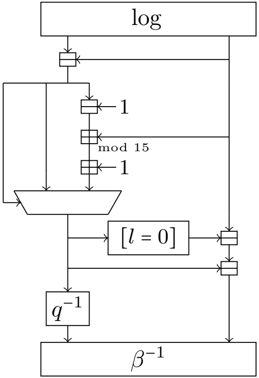
GOST S-Box

Structural and Decomposition Cryptanalysis



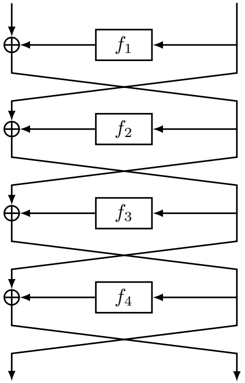
Feistel Networks

x	$E(x)$
0	182
1	210
2	78
3	251
4	97
...	
252	112
253	19
254	224
255	74

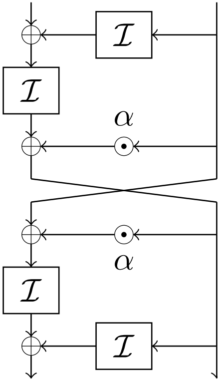


GOST S-Box

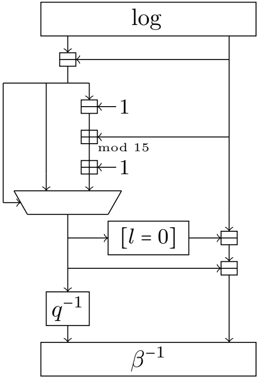
Structural and Decomposition Cryptanalysis



Feistel Networks

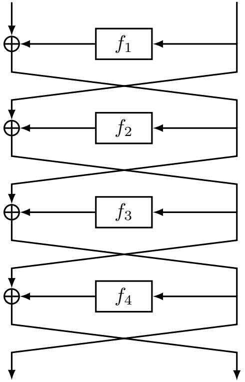


6-bit APN Permutation

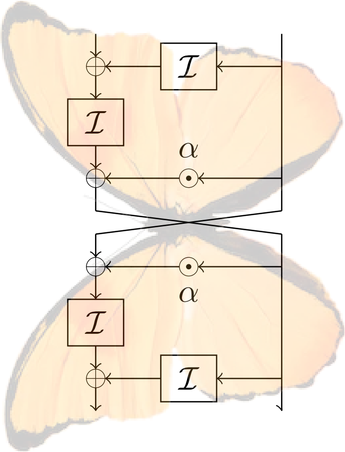


GOST S-Box

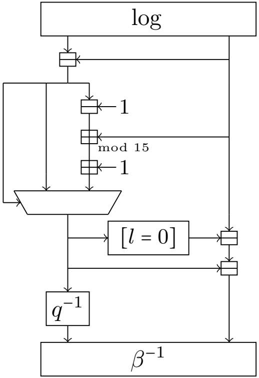
Structural and Decomposition Cryptanalysis



Feistel Networks



6-bit APN Permutation



GOST S-Box

Structural and Decomposition Cryptanalysis

Léo Perrin and Aleksei Udovenko.
Algebraic Insights into the Secret Feistel Network.
In *Fast Software Encryption - FSE 2016*, pages 378–398.

Léo Perrin, Aleksei Udovenko, and Alex Biryukov.
Cryptanalysis of a Theorem: Decomposing the Only Known Solution to the Big APN Problem.
In *Advances in Cryptology - CRYPTO 2016*, pages 93–122.

Alex Biryukov, Léo Perrin, and Aleksei Udovenko.
Reverse-Engineering the S-Box of Streebog, Kuznyechik and STRIBOBr1.
In *Advances in Cryptology - EUROCRYPT 2016*, pages 372–402.

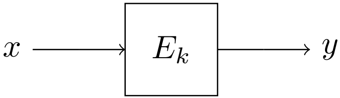
Léo Perrin and Aleksei Udovenko.
Exponential S-Boxes: a Link Between the S-Boxes of BelT and Kuznyechik/Streebog.
IACR Trans. Symmetric Cryptol., 2016(2):99–124.

Nonlinear Invariant Cryptanalysis

Properties of messages that are preserved through encryption

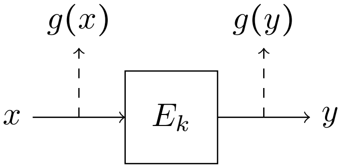
Nonlinear Invariant Cryptanalysis

Properties of messages that are preserved through encryption



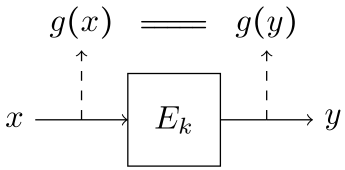
Nonlinear Invariant Cryptanalysis

Properties of messages that are preserved through encryption



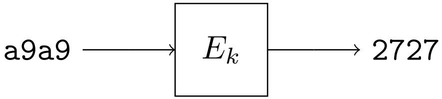
Nonlinear Invariant Cryptanalysis

Properties of messages that are preserved through encryption



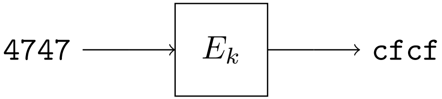
Nonlinear Invariant Cryptanalysis

Properties of messages that are preserved through encryption

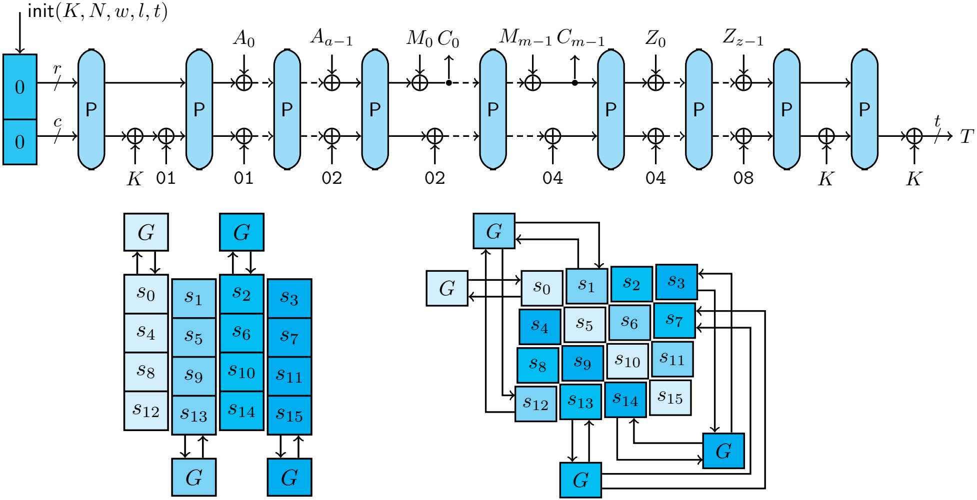


Nonlinear Invariant Cryptanalysis

Properties of messages that are preserved through encryption



Nonlinear Invariant Cryptanalysis



Analysis of the NORX Authenticated Encryption

Figure credits: TikZ for Cryptographers

Nonlinear Invariant Cryptanalysis

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Theoretical study of linear layers
preserving degree-d invariants

Nonlinear Invariant Cryptanalysis

Alex Biryukov, Aleksei Udovenko, and Vesselin Velichkov.
Analysis of the NORX Core Permutation.
Cryptology ePrint Archive, Report 2017/034, 2017.
https://eprint.iacr.org/2017/034.

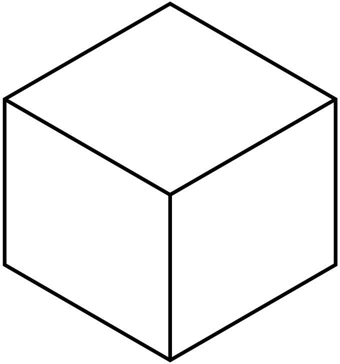
Christof Beierle, Alex Biryukov, and Aleksei Udovenko.
On Degree-d Zero-Sum Sets of Full Rank.
Cryptology ePrint Archive, Report 2018/1194, 2018.
https://eprint.iacr.org/2018/1194.

Plan

White-box Cryptography

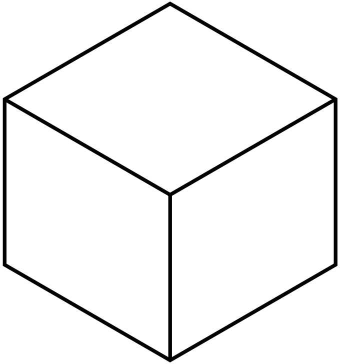
Alex Biryukov and Aleksei Udovenko.
Attacks and Countermeasures for White-box Designs.
In *Advances in Cryptology - ASIACRYPT 2018 II*, pages 373–402.

White-box model



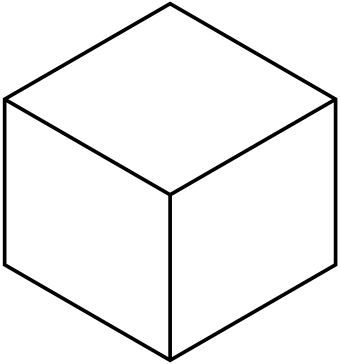
White-box model

- implementation is fully available to an adversary
- secret key should be unextractable
- **extra**: one-wayness, incompressibility, traitor traceability, ...



White-box model

- implementation is fully available to an adversary
- secret key should be unextractable
- **extra**: one-wayness, incompressibility, traitor traceability, ...



- The most **challenging** direction (this work):
  white-box implementations of
  existing symmetric primitives,
  e.g. the AES block cipher

Example: Secure White-box

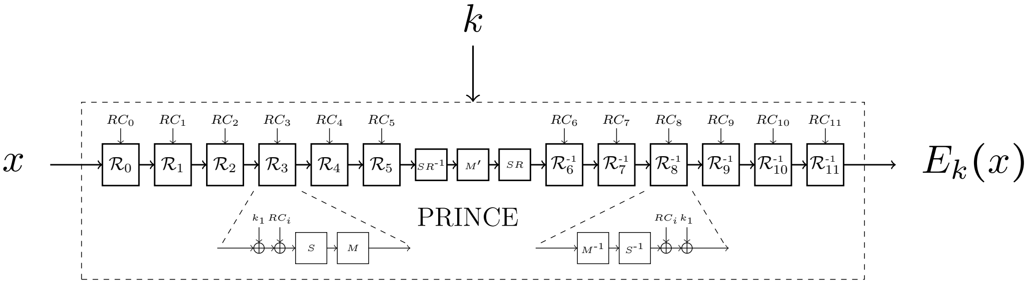


Figure credits: TikZ for Cryptographers

Example: Secure White-box

x	$E(x)$
0000000000000000	9333dd078833edd3
0000000000000001	7072b89243c84359
0000000000000002	7838040f2b7f9af6
0000000000000003	0b502e4231f42da3
0000000000000004	c39ea8c9434252aa
...	
fffffffffffffffb	8f1a82bc7af09497
fffffffffffffffc	9aaf33009a8e9a2f
fffffffffffffffd	5cd335922f9f0236
fffffffffffffffe	39d0e8b9a0eded09
ffffffffffffffff	daf2ced4ab8fc658

Figure credits: TikZ for Cryptographers

Example: Secure White-box

x	$E(x)$
0000000000000000	9333dd078833edd3
0000000000000001	7072b89243c84359
0000000000000002	7838040f2b7f9af6
0000000000000003	0b502e4231f42da3
0000000000000004	c39ea8c9434252aa
...	
fffffffffffffffb	8f1a82bc7af09497
fffffffffffffffc	9aaf33009a8e9a2f
fffffffffffffffd	5cd335922f9f0236
fffffffffffffffe	39d0e8b9a0eded09
ffffffffffffffff	daf2ced4ab8fc658

Impractical! 128 exbibytes for a 64-bit cipher!

Figure credits: TikZ for Cryptographers

White-box: Industry vs Academia

White-box: Industry vs Academia

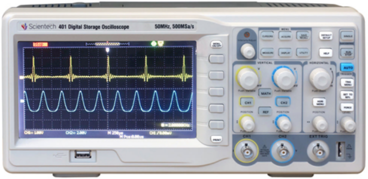
- WB has many applications
- strong need for *efficient* WB
- industry **does** WB:
  hidden designs

White-box: Industry vs Academia

- WB has many applications
- strong need for *efficient* WB
- industry **does** WB:
  hidden designs

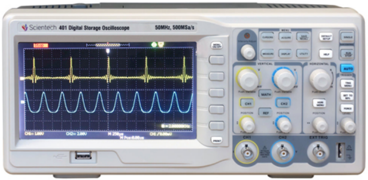
- **theory**: approaches using iO/FE, currently *impractical*
- **practical WB-AES**:
  few attempts (2002-2017),
  all broken
- powerful DCA attack
  (CHES 2016)

White-Box: Differential Computation Analysis (DCA)



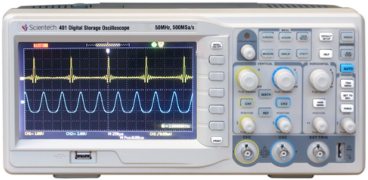
- **DCA = Differential Power Analysis (DPA)**
  applied to white-box implementations
- Most of the implementations broken automatically

White-Box: Differential Computation Analysis (DCA)



- **DCA = Differential Power Analysis (DPA)** applied to white-box implementations
- Most of the implementations broken automatically
- Side-channel protection: **masking schemes**

White-Box: Differential Computation Analysis (DCA)



- **DCA = Differential Power Analysis (DPA)** applied to white-box implementations
- Most of the implementations broken automatically
- Side-channel protection: **masking schemes**

this work:

Can we apply the masking protection for white-box impl.?

General Setting

- Boolean **circuits**
- **obfuscated** reference implementation

General Setting

- Boolean **circuits**
- **obfuscated** reference implementation
- **predictable values**: computations from ref. impl., e.g.

$$s = Bit_1(SBox(pt_1 \oplus k_1))$$

General Setting

- Boolean **circuits**
- **obfuscated** reference implementation
- **predictable values**: computations from ref. impl., e.g.

$$s = Bit_1(SBox(pt_1 \oplus k_1))$$

- **masking**: $\exists v_1, \ldots, v_t$ nodes (*shares*), $f : \mathbb{F}_2^t \to \mathbb{F}_2$ s.t. for any encryption

$$f(v_1, \ldots, v_t) = s$$

Masking Schemes

- **Example** Boolean masking: linear decoder $f = \bigoplus_i v_i$
- **Example** FHE: complex non-linear decoder f

Masking Schemes

- **Example** Boolean masking: linear decoder $f = \bigoplus_i v_i$
- **Example** FHE: complex non-linear decoder f
- Aim for efficient schemes: relatively small t (number of shares)

Masking Schemes

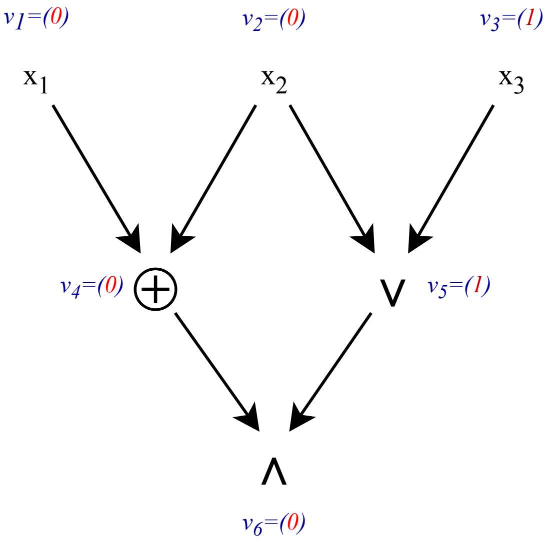
- **Example** Boolean masking: linear decoder $f = \bigoplus_i v_i$
- **Example** FHE: complex non-linear decoder f
- Aim for efficient schemes: relatively small t (number of shares)

$\Rightarrow$ can be secure only if
the locations of the shares in the circuit are unknown!

this work: exploring this possibility

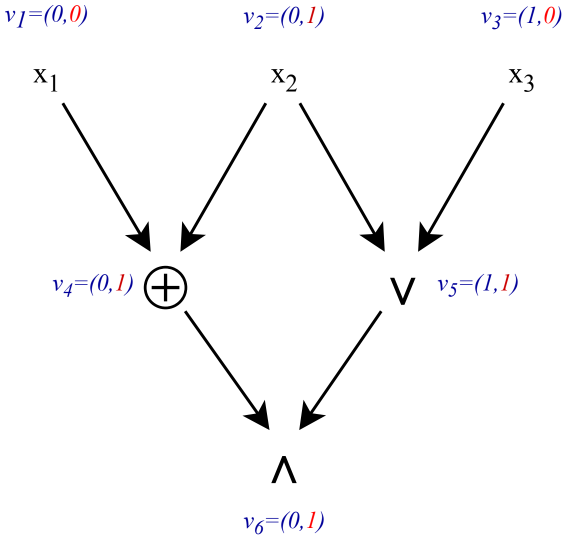
Plan

(Generalized) Differential Computation Analysis (DCA)



$x^{(1)} = (0,0,1)$

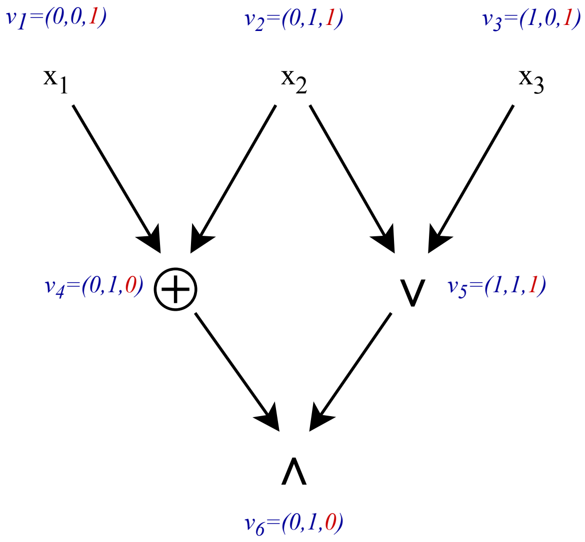
(Generalized) Differential Computation Analysis (DCA)



$x^{(1)} = (0,0,1)$

$x^{(2)} = (0,1,0)$

(Generalized) Differential Computation Analysis (DCA)



$$x^{(1)} = (0,0,1)$$

$$x^{(2)} = (0,1,0)$$

$$x^{(3)} = (1,1,1)$$

The Linear Algebra Attack

- consider Boolean masking (**linear** decoder)
- matching with a predictable value s:
  a basic linear algebra problem:

$$M \times z = s, \qquad M = [v_1 \mid \ldots \mid v_n]$$

The Linear Algebra Attack

- consider Boolean masking (**linear** decoder)
- matching with a predictable value s:
  a basic linear algebra problem:

$$M \times z = s, \qquad M = [v_1 \mid \dots \mid v_n]$$

- v_i is the vector of values computed in a node i of the circuit
- z is the vector indicating locations of shares among nodes of the circuit

The Linear Algebra Attack

- consider Boolean masking (**linear** decoder)
- matching with a predictable value s:
  a basic linear algebra problem:

$$M \times z = s, \qquad M = [v_1 \mid \ldots \mid v_n]$$

- v_i is the vector of values computed in a node i of the circuit
- z is the vector indicating locations of shares among nodes of the circuit

higher number of shares does not prevent the attack...

The Linear Algebra Attack

- consider Boolean masking (**linear** decoder)
- matching with a predictable value s:
  a basic linear algebra problem:

$$M \times z = s, \qquad M = [v_1 \mid \ldots \mid v_n]$$

- v_i is the vector of values computed in a node i of the circuit
- z is the vector indicating locations of shares among nodes of the circuit

higher number of shares does not prevent the attack...

Goubin et al: How to Reveal the Secrets of an Obscure White-Box Implementation. (ePrint 2018/098)

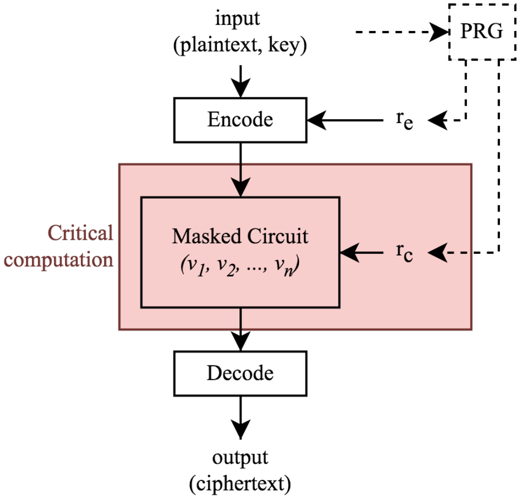
Plan

1. Introduction
2. Thesis Overview
3. White-box Cryptography
   - Introduction
   - Attack Methods
   - Countermeasures

Algebraic Security (1/3)

Security Model:

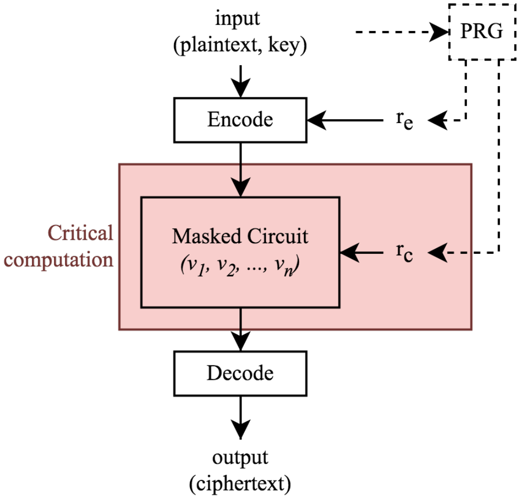
1. random bits allowed
   - as in classic masking
   - model unpredictability
   - in WB impl. as **pseudorandom**



Algebraic Security (1/3)

Security Model:

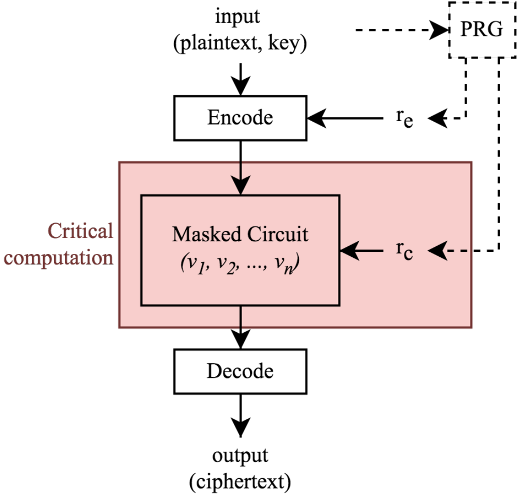
1. random bits allowed
   - as in classic masking
   - model unpredictability
   - in WB impl. as **pseudorandom**
2. **Goal:**
   any $f \in span\{v_i\}$ is unpredictable



Algebraic Security (1/3)

Security Model:

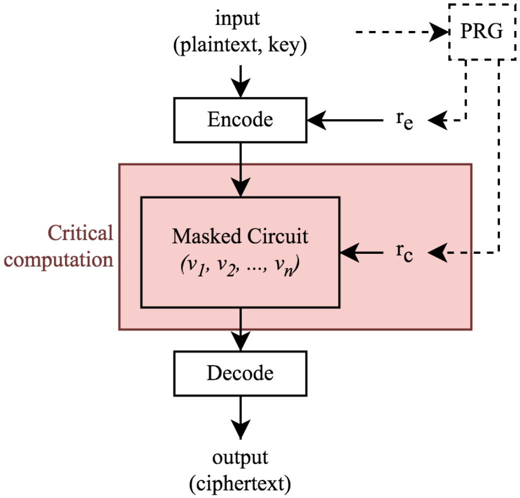
1. random bits allowed
   - as in classic masking
   - model unpredictability
   - in WB impl. as **pseudorandom**
2. **Goal:**
   any $f \in span\{v_i\}$ is unpredictable
3. **isolated** from obfuscation problems



Algebraic Security (2/3)

Adversary:

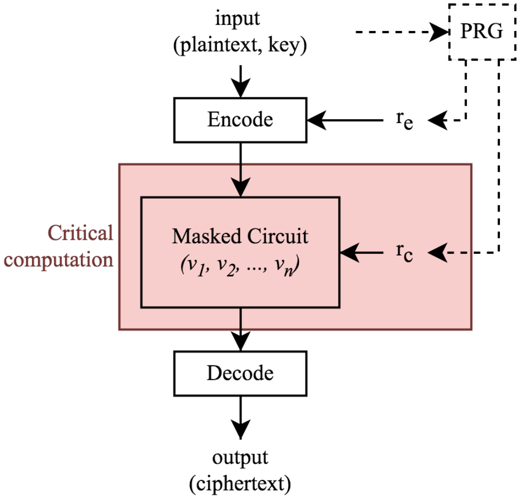
1. chooses plaintext/key pairs



Algebraic Security (2/3)

Adversary:

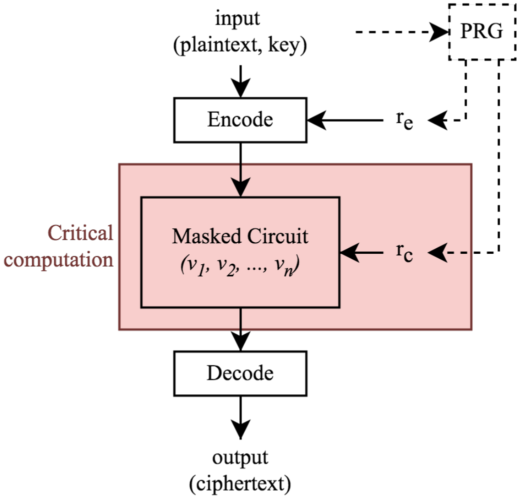
1. chooses plaintext/key pairs
2. chooses $f \in span\{v_i\}$



Algebraic Security (2/3)

Adversary:

1. chooses plaintext/key pairs
2. chooses $f \in span\{v_i\}$
3. tries to **predict** values of this function
   (i.e. before random bits are sampled)



Algebraic Security (3/3)

Proposition

Let $F = \{f(x,\cdot,\cdot) \mid f(x, r_e, r_c) \in span\{v_i\},\ x \in \mathbb{F}_2^N\}$.
Let $e = -\log_2\left(1/2 + \max_{f\in F} bias(f)\right)$.
Then for any adversary $\mathcal{A}$ choosing Q inputs

$$\mathrm{Adv}[\mathcal{A}] \leq min(2^{Q-|r_c|}, 2^{-eQ}).$$

Algebraic Security (3/3)

Proposition

Let $F = \{f(x,\cdot,\cdot) \mid f(x, r_e, r_c) \in span\{v_i\},\ x \in \mathbb{F}_2^N\}$.
Let $e = -\log_2\left(1/2 + \max_{f \in F} bias(f)\right)$.
Then for any adversary $\mathcal{A}$ choosing Q inputs

$$\mathrm{Adv}[\mathcal{A}] \leq min(2^{Q-|r_c|}, 2^{-eQ}).$$

Corollary

Let k be a positive integer. Then for any adversary $\mathcal{A}$

$$\mathrm{Adv}[\mathcal{A}] \leq 2^{-k} \text{ if } e > 0 \text{ and } |r_c| \geq k \cdot (1 + \frac{1}{e}).$$

Algebraic Security (3/3)

Proposition

Let $F = \{f(x,\cdot,\cdot) \mid f(x, r_e, r_c) \in span\{v_i\},\ x \in \mathbb{F}_2^N\}$.
Let $e = -\log_2\left(1/2 + \max_{f \in F} bias(f)\right)$.
Then for any adversary $\mathcal{A}$ choosing Q inputs

$$\mathrm{Adv}[\mathcal{A}] \leq min(2^{Q-|r_c|}, 2^{-eQ}).$$

Corollary

Let k be a positive integer. Then for any adversary $\mathcal{A}$

$$\mathrm{Adv}[\mathcal{A}] \leq 2^{-k} \text{ if } e > 0 \text{ and } |r_c| \geq k \cdot (1 + \frac{1}{e}).$$

Information-theoretic security!

Minimalist Quadratic Masking Scheme

Masking scheme

- quadratic decoder: $(a, b, c) \mapsto ab \oplus c$
- set of **gadgets**
- provably secure **composition**

function EvalXOR$((a, b, c), (d, e, f), (r_a, r_b, r_c), (r_d, r_e, r_f))$
$\quad (a, b, c) \leftarrow$ Refresh$((a, b, c), (r_a, r_b, r_c))$
$\quad (d, e, f) \leftarrow$ Refresh$((d, e, f), (r_d, r_e, r_f))$
$\quad x \leftarrow a \oplus d$
$\quad y \leftarrow b \oplus e$
$\quad z \leftarrow c \oplus f \oplus ae \oplus bd$
$\quad$ **return** (x, y, z)

function EvalAND$((a, b, c), (d, e, f), (r_a, r_b, r_c), (r_d, r_e, r_f))$
$\quad (a, b, c) \leftarrow$ Refresh$((a, b, c), (r_a, r_b, r_c))$
$\quad (d, e, f) \leftarrow$ Refresh$((d, e, f), (r_d, r_e, r_f))$
$\quad m_a \leftarrow bf \oplus r_c e$
$\quad m_d \leftarrow ce \oplus r_f b$
$\quad x \leftarrow ae \oplus r_f$
$\quad y \leftarrow bd \oplus r_c$
$\quad z \leftarrow am_a \oplus dm_d \oplus r_c r_f \oplus cf$
$\quad$ **return** (x, y, z)

function Refresh$((a, b, c), (r_a, r_b, r_c))$
$\quad m_a \leftarrow r_a \cdot (b \oplus r_c)$
$\quad m_b \leftarrow r_b \cdot (a \oplus r_c)$
$\quad r_c \leftarrow m_a \oplus m_b \oplus (r_a \oplus r_c)(r_b \oplus r_c) \oplus r_c$
$\quad a \leftarrow a \oplus r_a$
$\quad b \leftarrow b \oplus r_b$
$\quad c \leftarrow c \oplus r_c$
$\quad$ **return** (a, b, c)

Minimalist Quadratic Masking Scheme

Security

1. algorithm to verify that bias $\neq 1/2$
2. max. degree on r: 4

function EvalXOR($(a, b, c), (d, e, f), (r_a, r_b, r_c), (r_d, r_e, r_f)$)
  $(a, b, c) \leftarrow$ Refresh($(a, b, c), (r_a, r_b, r_c)$)
  $(d, e, f) \leftarrow$ Refresh($(d, e, f), (r_d, r_e, r_f)$)
  $x \leftarrow a \oplus d$
  $y \leftarrow b \oplus e$
  $z \leftarrow c \oplus f \oplus ae \oplus bd$
  return (x, y, z)

function EvalAND($(a, b, c), (d, e, f), (r_a, r_b, r_c), (r_d, r_e, r_f)$)
  $(a, b, c) \leftarrow$ Refresh($(a, b, c), (r_a, r_b, r_c)$)
  $(d, e, f) \leftarrow$ Refresh($(d, e, f), (r_d, r_e, r_f)$)
  $m_a \leftarrow bf \oplus r_c e$
  $m_d \leftarrow ce \oplus r_f b$
  $x \leftarrow ae \oplus r_f$
  $y \leftarrow bd \oplus r_c$
  $z \leftarrow am_a \oplus dm_d \oplus r_c r_f \oplus cf$
  return (x, y, z)

function Refresh($(a, b, c), (r_a, r_b, r_c)$)
  $m_a \leftarrow r_a \cdot (b \oplus r_c)$
  $m_b \leftarrow r_b \cdot (a \oplus r_c)$
  $r_c \leftarrow m_a \oplus m_b \oplus (r_a \oplus r_c)(r_b \oplus r_c) \oplus r_c$
  $a \leftarrow a \oplus r_a$
  $b \leftarrow b \oplus r_b$
  $c \leftarrow c \oplus r_c$
  return (a, b, c)

Minimalist Quadratic Masking Scheme

Security

1. algorithm to verify that bias $\neq 1/2$
2. max. degree on r: 4

$\Rightarrow$ bias $\leq 7/16$

for 80-bit security we need $|r_c| \geq 940$

```
function EvalXOR((a, b, c), (d, e, f), (r_a, r_b, r_c), (r_d, r_e, r_f))
    (a, b, c) ← Refresh((a, b, c), (r_a, r_b, r_c))
    (d, e, f) ← Refresh((d, e, f), (r_d, r_e, r_f))
    x ← a ⊕ d
    y ← b ⊕ e
    z ← c ⊕ f ⊕ ae ⊕ bd
    return (x, y, z)

function EvalAND((a, b, c), (d, e, f), (r_a, r_b, r_c), (r_d, r_e, r_f))
    (a, b, c) ← Refresh((a, b, c), (r_a, r_b, r_c))
    (d, e, f) ← Refresh((d, e, f), (r_d, r_e, r_f))
    m_a ← bf ⊕ r_c e
    m_d ← ce ⊕ r_f b
    x ← ae ⊕ r_f
    y ← bd ⊕ r_c
    z ← a m_a ⊕ d m_d ⊕ r_c r_f ⊕ cf
    return (x, y, z)

function Refresh((a, b, c), (r_a, r_b, r_c))
    m_a ← r_a · (b ⊕ r_c)
    m_b ← r_b · (a ⊕ r_c)
    r_c ← m_a ⊕ m_b ⊕ (r_a ⊕ r_c)(r_b ⊕ r_c) ⊕ r_c
    a ← a ⊕ r_a
    b ← b ⊕ r_b
    c ← c ⊕ r_c
    return (a, b, c)
```

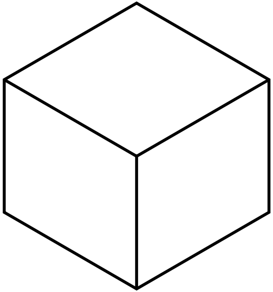
Proof-of-concept masked AES-128

1. MQMS + 1-st order Boolean masking
2. 31,783 $\rightarrow$ 2,588,743 gates expansion (x81)
3. 16 Mb code / 1 Kb RAM / 0.05s per block on a laptop
4. (unoptimized)

github.com/cryptolu/whitebox

Conclusions

1. new attack methods $\Rightarrow$ new constraints on a white-box impl.
2. new results on provable security for white-box model
3. new links with side-channel research



Design and Cryptanalysis of Symmetric-Key Algorithms in Black and White-box Models

Aleksei Udovenko
aleksei.udovenko@uni.lu

- Design of Symmetric-key Algorithms
- Structural and Decomposition Cryptanalysis
- Nonlinear Invariant Cryptanalysis
- White-box Cryptography